BASIC JAVASCRIPT CHEAT SHEET

Author: Saif ur Rehman

Saifr7493@gmail.com

1) alert() = > Use for Window-popup;

Window is a global object of your Page

2) var a="hello", variable cant be declare again,it can be used only a variable_name to redeclare or edit,like var a="Hello",a =”World.

BUT,var a = “Saif” , var a =”Rehman(This is Wrong).

3) var a =99 ,You can also assign a number to a variable not only stirng.

4) If we add a string with a number , Javascript convert the number into string and concatenate it.

Eg=> var a = 10 + "1"

Result will be 101(Concate the String with Number)

5) Varibale cant contain any space and cant be start with a number or cant be defined with a Keyword.If variable name is long use CamelCase convention (var helloWorldPak;)

6) var a = 10 % 3 It will return the remainder .

Which is a = 1

7) var num=1,

num++Pre increment

++num post increment

8) num=1;

num++ = 1

now in next line it will increment it by 1 which is 2

*9) var a = 1+3*5/10;*

Highest precendence will sort out first

For this amibiguity use ().

**and/ First Precedence.*

+and- Second Precedence.

10) prompt(ask,default Answer) = >

It will take two arguments and ask user for the input,one argument for the input and the second for the default if user doesn't enter anything it will provide the default argument value.

11).var a = prompt("Enter Your Name","Saif");

In a Result will be Saif if the user Enter Nothing in the pop-up Window.

If else and else if statements.

12) == checks the value and === checks for the datatype

, < ,> =< , => ,&& AND || OR

>>1 == "1" //true.

>>1 === "1" //false.

Arrays

13) To declare Arrays , var a = [], to call its content use a[indexNumber] ,always start a index with 0.

14) push() ,Using the keyword, push , you can add one or more elements to the end of an array.

15) pop () ,Using the keyword, pop , you can remove the last element of an array.

16) shift() ,Use the shift method to remove an element from the beginning of an array.

17) unshift() ,To add one or more elements to the beginning of an array, use the unshift method.

18) splice() ,Use the splice method to insert one or more elements anywhere in an array, while optionally removing one or more elements that come after it

19) SPLICE ALWAYS USE ON ARRAY WHILE SLICE USE ON STRING AS WELL AS ON ARRAYS.

20) SLICE COPY AND DOESNT EFFECT THE ORIGNAL BUT SPLICE EFFECTS THE ORIGINAL ARRAY

21) splice(2,5) 2=>start form here 5=>remove 5 elements after 2

22) For loop for(; ;)
start; end; inc/dec

23) flag in loops for example, inside for there is a if , set a flag = true /flase,if condition mets change the flag value and test after the loop for the if condition for the flag value.

24) Nested loops (inside loop will iterate complete then inc the outerloop.

25) toLowerCase() => Convert to lowercase.

26) toUpperCase() => Covert to Uppercase.

27) slice() = > use to copy the array and strings.

28) indexOf() => use to return the index of the word inside the paranth,if text doesnt available it returns -1.

28) lastIndexOf() => To find the last instance of a segment in a string, use lastIndexOf.

29) charAt() => Use to find the index number of a string.

30) replace() => use to replace a word or a sentence in a string .If a word appear more than one time you can use gloably to change the all instance..In a global replace, you enclose the segment to be replaced by slashes instead of quotation marks, and follow the closing slash with "g" for "global." The segment to be inserted is enclosed by quotation marks, as in a one-time replace. like, replace(/okay/g,"ACHA").

31) Math.round() => Round the Number to the nearest number,3.9 to 4 3.2 to 3.

32) Math.ceil() => Round the Number to the upper value , 3.2 to 4 3.9 to 4.

33) Math.floor() => Round the Number to the lower value , 3.9 to 3.

34) Math.random() => Gives a 16 decimal random number greater than 0 and less than 1.

0.0000000000000000 -0.9999999999999999.

35) Multiply by 17 digits number to convet the number to integer number.

36) We can also make a dice game using these random numbers.

37) If you perform arithematic operation with string and number other than number , JS convert string to number and perform maths

38) pasrseInt () => use to convert the string to number

39) parseFloat() => use to convert the sting float number to float number

40) if we use parseint in floating numbers it loops off the decimals.

41) To convert the string into Number we use Number()

to convert.

42) To convert the number to string we use .toSting().

43) To covert the big decimal to limited use toFixed().it returns string always.

44) To get the current date of the computer you just type the built in function var tDate = new Date(),its a objct which passes the objects and get the current date from the computer.

45) Day Start from the index 0 to 6. 0 for Sunday 6 for Saturday.

46) getDay() days of week 0-6 0 is sunday.

47) getMonth() month 0-11 0 is January.

48) getDate() day of Month 1-31.

49) getFullYear return the 4 digits year 2018.

50) getHours() 0-23, 0 is midnight 12 is noon and 23 is 11pm.

51) get Minutes(),getSeconds() they both gives 0-59 ,getMiliseconds() gives 0-999.

52) getTime(), Miliseconds since 1st Jan 1970.

53)setFullYear,setMonth,setDate,setHours,setMinutes,setSeconds,setMilliseconds =>They all are use to set the date by your own .

54) function(), its a right way to define a function.

55) function(arguments) , to pass a arguments this is

a right way to pass a arguments.

56) You can pass as many arguments as you want.

57) You can return the function and set it to any variable.
"funtion(){return something;} var a = function()."

58) Global vs Local Function = >

a) A global variable is one that's declared in the main body of your code—that is, not inside
a function.

b) A local variable is one that's declared inside a function. It can be either a parameter of
the function, which is declared implicitly by being named as a parameter, or a variable
declared explicitly in the function with the var keyword

59) What makes a global variable global is that it is meaningful in every section of your code,
whether that code is in the main body or in any of the functions. Global scope is like global
fame. Wherever you go in the world, they've heard of Bill Clinton.

60) A local variable is one that's meaningful only within the function that declares it. Local
scope is like local fame.

61) switch statement use to test conditions.Its a alternate of If/Else.

62) While loop is :intialize is done before the declaration and increment the condition inside the paranthesis.

```
var i = 0;
while (i <= 3){
alert(i);
i++;s
}
```

63) DO WHILE LOOP :: Since a do...while loop executes the code inside the curly brackets before it gets to the loop-limiter at the bottom, it always executes that code at least once, no matter what the loop-limiter says.

```
var i = 0;
do{
alert(i);
i++;
}while(i < 0);
```

64) External and internal javaScript Files.Generally, the best place for scripts, though, is at the end of

the body section.This guarantees that CSS styling and image display won't get held up while scripts are loading.

65) All of these user actions—clicking a button, moving the mouse, moving from one field to another in a form, selecting a different option—are known as events. JavaScript code that responds to an event—for example, displaying a guarantee or swapping an image when the pointer hovers over it—is called an event handler.

66) href="JavaScript:void(0)" dont reload the page from the top of the browser.

67) Inline event handler vs Event Trigger::

A) <img src="button-greet.png" onClick="alert('Hello world!');">

B)<imgsrc="button-greet.png" onClick="greetTheUser();">

68) onMouseOver : This is the markup that replaces the before picture with the after picture.

```
function swapPic() {
document.getElementById("before").src="after-pic.jpg";}

<imgsrc="before-pic.jpg"id="before"
onMouseover="swapPic(id,'after-pic.jpg');"
```

The markup has to pass both the id and the name of the replacement image to the function as arguments

function swapPic(eId, newPic) {
document.getElementById(eId).src = newPic;}

69) onMouseOut : Revert the picture when user mouse away from the picture.

70) onFocus : Field Turns dark when the user click the field.

71) onBlur : Its opposite of onFocus,when we move to another
field the has no longer the focus.

72) this.style.background='White';
//Change the background Colour

73) onSubmit: Use to sublit the form or to get the valus from the form.

74) .innerHTML used to read or write elements and

text .

75) document.getElementById().className = "". Its use to add the class in the element.

76) If you want to add a class to an element, preserving its ex83)isting classes, you can do it.

```
document.getElementById("p1").className += " big";
```

77) document.getElementById("link1").href, Uses to get the address of the element.

78) This statement left-floats an image.

```
document.getElementById("pic99").style.cssFloat = "left";
```

This statement makes an element invisible.

```
document.getElementById("div9").style.visibility = "hidden";
```

This statement gives an element left and right margins of 10 pix.

```
document.getElementById("mainPic").style.margin = "0 10px 0 10px;";
```

79) If you want to change al the things inside the tag we use : document.getElementsByTagName("p");

80) Now the variable contains the array like collections par[0] is the first element par[2] is the second element in the tag.

81) You can also change the text inside the tag by .innerHTML.

82) You can also use loop to get through all the elements Collections.

83) It makes the Collection of img divs ul and many more.

document.getElementsByTagName("img");

document.getElementsByTagName("div");

document.getElementsByTagName("ul");

84) If you dont want all the collection off element make the id in particular div then set it inside.

DOM

85)Tree like structure of HTML is called as DOM or Document

86)All tag inside are objects.

87)The way we define the structure is called as Model.

88)console.log(document) returns the complete Document off HTML.

89)console.log(document.head) returns the head elements.

90)console.log(document.body) returns the body elements.

90)We can change or edit inside the document through DOM.

91)console.log(refrence) use to returns the sepecific id or class.

92)refrence.innerText will use to add the text between the tag.

93)Id is unique and classname not.classname will store lie an array if there is more than one class.

94)To access the class array, ref[number].innerText = "SAIF";

95)Text and Element Node.

95)Every Element is Called as Node.

96)Every Text inside tags are called as TextNode.

97)createElement used to create Elements.

98)createTextNode used to create text inside the Elements.

99)appendChild is used to append the textnode in the elements.

100) childNodes[] is use to access the tags and if there is a gap it will called the junk artifacts.

101) Remove the spaces between the tags to remove the juck artifcats

102) childNodes[].nextSibling to target the next sibling of the parent element.

103) childNodes[].previousSibling to target the previous sibling of the parent element.

104) childNodes[].parentNode to target the next sibling of the parent element.

105) getElementById and tag name,Both approaches can change things on your web page, but neither is able to deal with all the things on the page, to create new

things, to move existing things, or to delete them with dom you can do it,

106) getElementById or getElementByClassName :: Both approaches can change things on your web page, but neither is able to deal with all the things on the page, to create new things, to move existing things, or to delete them

107) When a node is enclosed within another node, we say that the enclosed node is a child of
the node that encloses it. So, for example, the node is a child of the <body> node.
Conversely, the <body> node is the parent of the node

108) Nodes with the same parent are known as siblings. So, <head> and
<body> are siblings because < html> is the parent of both. The two <p>'s are siblings because
* is the parent of both.*

109) Every node that is enclosed by another node is the

child of the node that
encloses it. Since the text node "important" is enclosed by the element node , this
particular text node is the child of , not <p>. The text nodes "This is " and "!" as well as the element node are siblings, because they're all enclosed by <p>. They're all children of <p>.

110) Document Object Model (DOM) is a hierarchy of parents and children

111) <
div
*>... is the same as *

112) nodeType:: If the node is an element like or <p> , the number is 1. If it's a text node,the number is 3.

113) parentNode.childNode[0] is same as parentNode.firstChild

114) And if there are, for example, 3 child nodes total , parentNode.childNode[2] is same as parentNode.lastChild.

115) nextSilbling and previousSibling used to target the next child and the previous Child.

116) If there is no child it will return null.

117) nodeType is used to check type of node.

118) In the example above, if the node is an element, the variable nType will be assigned the number 1. If the node is
text, it'll be assigned the number 3.

119) You can get additional information about a node by using nodeName .

120) if the node is an element, the variable nName will be assigned a string, like P or DIV.

121) if the node is a text node, the name of the node is always #text—in lower-case.

122) An element node like P or IMG has a name but no value. If you try to assign an element node value to a variable, the variable will be assigned null
123) innerHTML includes all the descendants of the element, including any inner element nodes
like as well as text nodes. The node value includes only the characters thatcomprise the single text node.

124) You can make a collection of all the child nodes of a targeted node using .childNodes;

125) The word on the left side of the equal sign is the attribute name. The word on the right side of the equal sign is the attribute value.

126) var hasClass = target.hasAttribute("class"), If it does, the variable hasClass is assigned true. If not, it is assigned false.

127) You can read the value of an attribute with getAttribute .

128) You can set the value of an attribute with setAttribute,
vartarget=document.getElementById("div1");
target.setAttribute("class, "special");

129) Similarly, you can make a collection of all the attributes of an element, with a statement
like this...
var list = document.getElementById("p1").attributes;

130) Node name is called as the element inside the parameter inside the tag of the element.

131) When you add a node to your web page by appending it as a child to a parent, as I did in the example above, the limitation is that you have no control over where the new element lands among the parent's children. JavaScript always places it at the end. So for example, if the div in the example above already has three paragraphs, the new paragraph will become the fourth one, even if you'd like to place it in a higher position.
The solution is INSERTBEFORE.

parentDiv.insertBefore(newParagraph, paragraph1);

132) To remove a node, use removeChild, .removeChild(nodeToRemove);

OBJECTS

134) var obj = { name:"Saif"};

135) You can change the Value of the objects, and you can also save a string value to it, You can also assign array value to it.

136) You can also assign a Boolean value, true or False.

137) You can also use an assignment statement to define a new property for an object.

138) You can also use an assignment statement to define a new property for an object,Just as you can create an undefined variable by not assigning it a value, you can create an object without any properties.

var deal4 = {};

139) If you want to create a property now and assign it a value later, you can create it with a
value of undefined, prop = undefined

140) You can delete property of an object, delete obj.prop.

141) You can check to see if a property of an object exists and return true and false , propertyExists = "market" in deal3;

142) You add variable to the object it becomes the

property of an object, Similarly you can attach the function to the object it will become the object's method .

143) name:funtion(){} , var a = name();

144) You can use this instead of the object name to referred the object,When JavaScript sees this keyword, it knows you're referring to the object that's being defined.

145) When you write this.whatever , JavaScript is smart enough to understand that you're referring to a property of the object that's being defined.

146) JavaScript lets you build such a factory. It's called a constructor function

147) The function name is capitalized. JavaScript doesn't care whether you do this or not, but it's conventional to do it to distinguish constructor functions from regular functions.

148) Each of the parameter values is assigned to a variable. But the variable is a property attached to some object whose name hasn't been specified yet. But don't worry. Just as the parameter values will be filled in by the calling code, so will the name of the object.

149) var plan1 = new Plan("Basic", 3.99, 100, 1000, 10);
This would be just a regular function call if it weren't for that new . It's the keyword that tells JavaScript to create a new object. The name of the new object is plan1 . Its properties are enumerated inside the parentheses,Now it's easy to mass-produce as many objects as you want, using the same pattern.

150) When you pass the value inside the constructor functions it will become the value of the obj's Property.

151) If you're creating more than one object with the same pattern of properties and methods, it's a convenience to build the method as well as the properties into the constructor function.

152) In Constructor Fucntions you declare with an assignment (something = somethingelse).

153) JSON.stringyfy is used to convert the objects into readable or string form.

154) Properties of object are always different for each objects,but The constructor function keeps duplicating the same method for each object, object after object.

155) If we were to create 100 objects using the constructor, JavaScript would duplicate the same method 100 times, once for each object.

156) We want only one copy of the method, shared by all objects created with the constructor, no matter how many objects are created,with the help of prototype statement we can do this.

157) First, we don't include the method in the constructor function, because that creates a copy of the method for every single object that's created with the constructor.

158) constructorFuntionName.prototype.Method-name = funtion(){},This is the syntax.

159) Objects can have prototype properties as well as prototype methods.

160) It's possible to override a prototype for any

individual object,In the same way, you could override a shared prototype method for any individual objects as well.

161) You can check to see if an object has a particular property by writing a simple statement like this = = var gotTheProperty = "price" in plan1;

162) If you want a list of an object's properties, there's a spiffy way to do it:
Note also that instead of prop , you could use any other legal variable name.

```
var listOfProperties = [];
for (var prop in plan1) {
listOfProperties.push(prop) }
```

163) Note that, unlike a regular for loop, this one doesn't establish a limit on the number of

loops or increment a counter. After each iteration, JavaScript automatically moves to the next property of the object and stops iterating when there are no more properties to enumerate.

164) If a method or property wasnt included in the original definations of object and was added by the prototype of constructor it will became a property of the object by inheritance.

165) hasOwnProperty is a function used to test the original method or property of the objects, it the method ot property was included through the prototype it will return false.

166) You can also test a literal property name by this syntax :::

```
abc=objname.hasOwnProperty("propertyName");
```

NOTE = for loop to check all the properties of the objects,the syntax is : for (var a in obj).

BROWSER CONTROL

167) In order to make things happen you have to use javascript to control your Web page and it is called as browser controls.

168) you can get your browser current location by using window.location.href,href will return complete address

169) window.location.hostname will return only domain note. no https no path only domain.

170) window.location.pathname will return you the path address means /home/add.html.

171) window.location.hash will give you the string "#humidifier" is assigned to the variable If there is no anchor in the URL, the variable is assigned an empty

string, "".

172) You can also edit the web page and make it to go to the url you want by window.location.href="http://....."

href pura address dega https kay sath complete.
hostname aur host srf pehla address dega https hata k jese www.web.com.
hash #something yahn jo huga wo dega ye

173) The statement above tells the browser to navigate to the page.

174) It would be nice if you could use window.pathname = ... to move to a different page on the current site or window.hash = ... to move to an anchor on the current page, but you can't.

175) What you can do, though, is query the browser for the domain name and combine that with the page where you want to go, like this::

```
var currentSite = window.location.hostname;
var destination = "http://" + currentSite + "/wow.html";
window.location.href = destination;
```

176) You can omit window. It's legal to use location.href , location.hostname , location.pathname , and location.hash . It's more common to include window.

177) You can omit href when you're detecting the URL. It's legal to use window.location , or simply location . (See above.) Including href is preferred for esoteric reasons.You can use document.URL as an alternative to window.location.href

178) There is a alternate of assigning the browser to mve to the required page is : window.location.assign("http://www.me.com");

179) Here's another alternate of this with a slighty different changes is window.location.replace(),when you use replace , the original page doesn't make it into the history. If the user presses Backspace after being taken to the new page, she's taken to the page that displayed before the original page since the original page is no longer in the history.

180) window.location.reload(true);

window.location.reload(false);

window.location.reload();

All three statements reload the current page. If the argument is true (example 1 above), the statement forces the browser to load the page from the server. If the argument is false (example 2) or if there is no argument (example 3), the browser will load the page from the cache if the page has been cached.

181) You can use window.location.href = window.location.href or any of the abbreviated alternatives to reload the current page. The reload is faster, but it doesn't allow you to specify whether the browser reloads from the server or the cache.
document.URL = document.URL doesn't work.

182) history.back(); and history.go(negative-number)is used to make the user go back the history.

183) history.forward();and history.go(positive-number) is used to make the user go forward the history.

In both cases, if there is no URL in the history that would make the move possible, the browser does nothing.

184) Using positive number to take forward and using the negative number take you to the back history.

185) If a negative number inside the parentheses is greater than the number of previous URLs in the history, the browser will do nothing. If a positive number inside the parentheses is greater than the number of forward URLs in the history, the browser will do nothing.

186) If the user clicked a link to get to the current page, you can get the URL of the page where the link was clicked,by using the document.referrer.

187) However, this works only if a link was clicked, including a link in a search result. If the user got to your page through a bookmark or by entering your URL in the address bar, the result of document.referrer will be an empty string, "".

188) window.open() use to open a short new blank window in the browser with a maximum size.

189) If you want to fill the window you can do this by some line of code.

```
var a =  window.open();
var content =  "something";
a.document.write(content).
```

This is the way you can write the content to the browser by your own.

190) You can also use to give a html page instead of writing the string in the document.write ,assign the simple html but it will overwirte the content.

191) The second way to fill the window with content is to assign a document to it,

```
monkeyWindow.location.assign("http://www.animals.com/capuchin.html");
monkeyWindow.location.href = "http://www.animals.com/capuchin.html";
```

192) The third and most common way to fill the window with content is to include the document assignment in the statement that opens the window.

```
var monkeyWindow = window.open("http://www.animals.com/capuchin.html");
```

193) This is how you close a window.

```
monkeyWindow.close();
```

194) In a windown.open("url","name"), first parameter is url and the second is the name of the window.

195) You can sepcify the open with no name but without url you can write this.

196) You can also give width and size of the new window by giving it the width and height in the third parameter.

197) In one string you can write height and widht both.

198) The number in the height and width will always in pixels and a minimum of 100 pixels.

199) left and top bh askta h isme parameter usko manage krnay k liye.

200) You can specify window size without specifying window position, but if you specify window position without size, it will be ignored since it will be a full-size window that fills the whole screen.

201) As usual, some or all of the parameters can be assigned to a variable, and the variable can be used in the statement that opens the window. Since the whole thing has to be a quoted string, the quotes within the string have to be changed to single quotes.

```
var windowSpecs = "'faq.html', 'faq', 'width=420,height=380,left=200,top=100'";
var faqPage = window.open(windowSpecs);
```

202) For the popup blocker you will use the following way to identify the popup.

```
function checkForPopBlocker() {
var testPop = window.open("",
"","width=100,height=100");
if (testPop === null) {
alert("Please disable your popup blocker.");}
testPop.close(); }
```

203) Normally, you'd run the test when the page loads.

```
<body onLoad ="checkForPopBlocker();">
```