JAVASCRIPT

<script>
{ Java code.
</script>

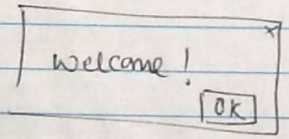
(or.)
<script src = "mycode.js">
</script>

file
Java code. { No need of script/script.

comment
//This is a comment in one line
or
/* Comment in any no. of lines */

semicolon
can omit in general
But NOT a good idea

alert( ) ⇒ alert ("welcome!");

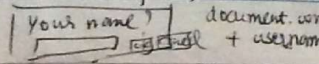


variable types
var( )
typeOf( )
Data types
→ number
→ string
→ Boolean
→ others eg. Object.

confirm( ) ⇒ if(confirm("want to go?"))
document.location.href
= "http://park.hongkd.com";

variable
var( ) ~ variables.
(its like a box.
make a var. & can put something in it.
⇒ var totalcost = 7000;
using variable.

prompt( ) ⇒ var user_name;
user_name = prompt("welcome", "your name");



document.write("Welc" + username + "!!!");

• Number
Java has only one type of #
→ 34.289
→ 34
→ 123e3; //123000000
all are number

⇒ var num1 = 256

JAVA SCRIPT

• string
(means text)
can use
'single' or "double speech"
speech marks marks
☆ if want to use (') inside use (" ") outside.
(wow!!)
⇒ var name = "David";
⇒ var title = 'professor';
⇒ var msg = "it's alright";

• Boolean
(true or false)
⇒ var ans = true; ← Boolean
⇒ var string = "true"; ← string

variable type can change
→ var storage = "David";
→ storage = 98;
it's ok

typeof( ) alert (typeof "John");
→ string OK

\n → next line

JAVASCRIPT

common changes

a = a + 1	a++
a = a - 1	a - -
a = a + 10	a += 10
hello = hello + "!"	hello += "!"
pigs = pigs * 5	pigs *= 5
cakes = cakes / students	cakes /= students

From 1 type to another

parseInt( )	to integer
parseFloat( )	to float point no
string( )	to string

EVENTS & FUNCTIONS

⇒ Onload event
• Onload is triggered when object has loaded
eg:-
<body onload = "alert('Hello')">
.....
</body>

⇒ Functions

- A function is a group of code

```
function do_something(){
    ... code goes here...
}
```

to run
do_something();

↳ You can also pass something to function

⇒ eg. function purchase(cats)
{ - - - }

- Run like this
  purchase(10);

↳ You can also get answer from function

⇒ function do_s()
{ ... return answer; }
. result = do_s();

⇒ A recursive function

```
function do_something(){
    ... code here...
    do_something()
}
```

HANDLE BUGS

open browser.
(Developer options)
(console)
(shows error

You can use console also for trying functions

if you do not want to show on main browser screen.
instead of alert()
use console.log()
↓
only on console screen you can see that.

Making decisions

if
if.. else
if.. else.. if

Comparing things
< , > <= >=
== equal to
!= not equal to

using braces
{if you have more than 2 lines of code}

⇒ switch(variable_name)
{ case "option_1":
  break;
...
case "option_n": do something
  alert("- - - - ");
}

↳ last you can also ~~add~~ use
→ default:

↳ if you don't use break it won't break

LOOPS

- while
- do.. while

⇒ string.indexOf("text")
gives location of first "text" in string

eg:-
var text = "The Cat's hat was wet";
0 1 2 3 4 5
result = text.indexOf("at");
- result will be 5

How Variables Work?

→ if defined within fncn can be used only within function

Hence local only to the function.
local variable

→ Global variable
→ works anywhere
→ defined in main function

→ Local & Global var. sharing name
priority to ~~global~~ local var

→ creating global variables inside fnc^n

⇒ don't declare any var NOT declared

→ will automatically become Global variable.

Logical Operators

→ How to use & manipulate Boolean values.

→ Boolean values ↔ true / false

→ Logical operators → && logical & ; || → logical or ; ! → logical NOT

Array

→ Linear continuous storage

→ Each box - unique identity - called index

1st box index = 0

⇒ eg: var pets = ["Dog", "Cat", "Rat"];

or

⇒ var pets = new Array(10);

defined but NOT assigned

⇒ var, pets = ["D", "C", "R"];

alert(pets.join(" and "));

// This shows "D and C and R

→ separator is by default ","

alert(pets.join());

// —— D, C, R

⇒ alert pets[2]; // "R"

⇒ alert(pets.length); // This shows 3

⇒ pets.push("H")

puts/adds an element to the end.

// D, C, R, H is new array.

→ Index are automatically added.

⇒ pets.unshift("H")

// puts new element at 1st.

H, D, C, R

⇒ pets.pop();

removes/pops the last item

⇒ pets.shift();

removes 1st item.

↳ combining 2 arrays.

array1.concat(array2)

Generating Random Numbers

⇒ var ran = Math.random();

Resulting range is [0, 1)

↳ multiply in order to get range you want

⇒ var ran = Math.random() * max value;

Resulting range [0, max value)

⇒ Math.floor()

↳ dumps the decimal place

2.82248 → 2

→ data-structure.length()

tells you how many data items data structure contains

For Loops

for(var index=0; index<x; index++)

{

}

for... in Loop

eg: var continents = ["Australia", "Af...

...]

for(var index in continents)

{

}

↳ Automatically runs for all indexes in continents.

- going through an array ↑
- going through an object ↓

[Object: it is a series of left hand side(&): Right hand side]

eg:

var one person = { initals: "Dr" age: 40, job: "Professor" };

for(var property in one person)

{ alert(property + "=" + one person[property]) }

for... of Loop

for(var continents of continents)

{

}

assigns the actual element of array 1 by 1

→ Omitting Parts

for (; ;)
{
alert ("welcome")
}

It is an infinite loop

→ or its ok if

for (; number<=12; number++)
{ (if it is already assigned earlier)
}

→ You can increase 2 values of variables at a time.

like

for (var rabbits=2; gen=1; gen<=12; gen++, rabbit*=2)
{ ... }

LOOP CONTROL

- break
  - ↳ Totally stops the loop
- continue
  - ↳ Stops the current iteration.

Advanced Array Functions

sort() indexOf() slice()
reverse() lastIndexOf()
splice()

sort() → sort from smaller to largest
↳ numbers / words

reverse()
↳ in reverse order of sort

eg:-
var pets=["D", "C", "R", "H"]
pets.sort().reverse
//result is R.H.D.C
↳ both fncn's applied in same fncn.

pets.indexOf("C")
will give, 1
if it can't find it will give (-1)

We can also SPECIFY, from where to start search

eg:-
foundat = pets.indexOf("Rabit", startsearch)

search will start from index = value of startsearch

→ slice() → is a copying parameter
It copies an array into a new variable.
(DOES NOT affect original array)

1st way: array = ["A", "B", "C", "D", "E"]
result = pets.slice (2)
it takes all element from 2 in result
//result → ["C", "D", "E"]

2nd way: array = pets.slice (1, 3)
from 1 to 2 (NOT 3)
// result → ["B", "C"]

→ splice()
(1) ↳ To remove element from an array
array.splice(position, quantity)

result = pets.splice(1,1);
// pets is ["D", "R", "H"]
// & result is ["C"]

(2) To add an element to an array used.
pet (1, 2, 3)
result = pet.splice(2,0,"R");
// result = []
pet = [1, 2, "R", 3]

(or)

pet (1, 2, 3, 4)
result = pet.splice(2, 1, "R", "S")
// result = [2]
pet : [1, 2, R, S, 4]

Advanced Array Operation

forEach()
map()

pets = ["D", "C", "H"];
for(var i=0; i< pets.length; i++)
{ alert (pets[i]); }

Same can be done by forEach()

pets.forEach(alert);

// This shows 3 separate alert

How

What does forEach() fnc work

```
function forEach(theArray, fn)
{ for(var i=0; i<theArray.length; i++)
  { fn(theArray[i], i, theArray);
  }
}
```

This is to show

for each takes. ~~the array~~, individual element, index & the array

eg:

```
var n = [1,2,3,4,5];
num.forEach(function(elem, idx, arr)
  { arr[idx] = elem * elem; });
alert(numbers);
// This shows [1,4,9,16,25);
```

It changed original array.

MAP()

similar to forEach but stores value in a diff^t array (it returns)

How it works?

```
function map(thearray, fn)
{ var results = [];
  for(var i=0; i<the Array.length; i++)
  results.push
  (fn(theArray[i], i, theArray);
  }
  return results;
}
```

eg:-

```
var square = function(ee)
  {return ee*ee;}
var numbers = [1,2,3,4,5]
var results = numbers.map(square);
alert(results);
```

This shows [1,4,9,16,25]

DOM

DOM means Document Object Model

When you load something into a browser it is converted into a DOM structure

White space nodes

Anything you can't see

eg: <body>
Hello // here is a white space that says go to next line

: <body> Hello

// No white space

NODE RELATIONSHIP

parent, child, sibling

```
function handleClick(event){
  event.stopPropagation();
  var node = event.target
  var thisPath = node.nodeName;
```

→ till parentNode exists

```
while (node.parentNode)
{ node = node.parentNode;
  thisPath = node.nodeName
    + ">" + thisPath;
}
alert(thisPath);
}
```

How to Trigger the code?

To trigger the code, the user clicks on a node. To enable this, event handlers are added to the nodes.

2 examples: HTML & SVG

They use the same code. Event handlers are added to every element

```
// Register the click event
handler for all node.
function attachHandler(node){
  if (node == null) return;
  node.onclick = handleClick;
  for(var i=0; i<node.childNodes.length;
      i++)
  { attachHandler(node.childNodes
      [i]);
  }
```

Different ways of reaching a node:

- getElementsByTagName()
- getElementById()
- setAttribute()
  - common way to change something.

```
the_node = getElementById("thisNode");
the_node.setAttribute("style", "color:red");
```

Creating a node

createElement(), createTextNode()

Adding a node → insertBefore(), appendChild()
- ends after

eg: horizontal line

```
newItem = document.createElement("hr");
destParent = document.getElementsByTagName("body")[0];
destParent.insertBefore(newItem, destParent.firstChild);
```

Put a hr on top of web page

Deleting Nodes

removeChild()

refer the parent to remove it.

```
this_node = getElementById("myPara");
this_node.parentNode.removeChild(this_node);
```

If want to remove all child.

```
var theNode = document.getElementById("theBody");
while (theNode.firstChild) // While there is a child to the Node.
    theNode.removeChild(theNode.firstChild);
```

Cloning Nodes

copying a node	the_node.cloneNode() → copies node without actual text within
copy a branch	the_node.cloneNode(true)
Adding node(s)	dest.appendChild(the_node)

Mouse Events

- onclick
- onmousedown
- onmouseup
- onmouseover
- onmouseout

<body onmouse=" ">

Timer Events

- setTimeout
- setInterval
- clearTimeout
- clearInterval

```
var the_timer;
the_timer = setTimeout(do_something, 1000)
```

↳ define a variable & use that to point to the timer.
- do_something will be executed 1000 msec (1 sec) later.

→ TO MOVE an image, a repeatative timer can be used

↳ To stop timer ⇓ use

<button onclick="clearTimeout(the_timer)">stop</button>

(or)

⇒ setInterval(do_something, 2000)

↳ does the thing again & again in every 2 sec)

How to end timer ?

again <button onclick="clearInterval(thetimer)">
stop</button>

Adding Events

addEventListener()

removeEventListener()

⇒ Instead of using
<body onload="">
you can use
window.onload=""

or

window.addEventListener("load", do_something)

⟹ More than one eventshandler can be added.

↳ Event Handlers are stored in array

↳ when event happens all handlers are executed

↳ Executed in order they are added.

To stop.

↳ var thebody = document.getElementById("theBody");
thebody.removeEventListener("load", do_something);

↳ this removes only event added by JavaScript

NOT something like
<button id: "btn0" onclick="alert("Hello")">

Advanced Use of functions

→ assign a function to a variable
→ pass a function to a function
→ return a function from a function

Events onload
functions function
retur

A new meta tag element in <head>

<meta http-equiv="refresh" content="1">

↳ it refreshes the html page every 1 second

HTML, JAVA, CSS

HTML, JAVA & CSS.

<html> </html>
<head> </head>
<body> </body>
<!-- comments -->
<h1> heading 1 </h1>
<h2> heading 2 (smaller than 1) upto 6 </h2>
 italic   emphasis
 bold   strong
<u> underline </u>
<big> bigger than rest slightly </big>
<small> smaller than rest slightly </small>
may not work later (NOT part of standard HTML)
<mark> Highlights the text </mark>
_{subscript}
^{superscript}
<ins> inserts a text & shows underlined </ins>
<del> strikes the text </del>
mainly for editors
 paragraph

Images & Attributes

<img src="mohit.jpg">
<!-- image source -->
<img src="mohit.jpg" width="300" height="300"> → pixels
<img src="mohit.jpg" width="300">
<!-- since height NOT mentioned automatically Aspect ratio is maintained -->

<img src="mohit.jpg" width="50%">
<!-- 50% is in respect to body of html page -->

ends line

Handling Sound

<audio src="beats.mp3"></audio>
but this will bring the audio & NOT play, so
<audio src="beats.mp3" autoplay></audio>
plays as soon as page opens
<audio src="beats.mp3" controls></audio>
brings some basic audio controls like sound volume & position.
<audio src="beats.mp3" autoplay loop></audio>
repeats the song after finishing
<audio src="beats.mp3" autoplay alt="A splashy 3 beat drum piece">
<!-- alt="description" to give textual description. Helpful for search engines & people who can't hear -->

Playing Video (similar to sound)

<video src="clip.mp4" autoplay>
<video src="clip.mp4" controls></video>
<!-- GUI controls (Graphic user interface) -->
<video src="clip.mp4" controls>
 sorry! your browser doesn't support the video
</video>

outer part gets picked up by new browsers

inner part gets picked up by old browsers

LINKS

<a> means anchor

Twitter

instead an image can be used or a video (anything)

position within a page

Add id="here" to the element u'll link to.

<a href name="#here">Go here

on another web page (at particular spot)

Go here

Lorem ipsum

↳ a random latin text

use it while practising

↳ so you can focus on attributes leaving the text UNcaistubig

lipsum.com

Void Elements

→ they do not have an end tag.

<meta name="author" content="David Rositer">

→ may or may not have attributes

- Handling media <img>
- Handling forms <input>
- Handling breaks
 <wbr> <hr>

 <!-- a forced break to next line -->

<wbr> <!-- used b/w word if breaks only if necessary i.e. (word not fit) -->

<hr> <!-- breaks with a horizontal rule -->

STYLES

General concept

(Information) + (style) → (visual output)

1 CSS (cascading style sheet) can be used by multiple pages.

css file: main.css

web page 1 → main.css ← web page n

<link href="main.css">

<head>
<title> -- </title>
<link href="cssfile.css" rel="stylesheet" type="text">
</head>

simple cssfile.css

```
h1 {color: purple}
p {color: blue}
```

(OR) instead of using different file. we can do as follows

<head>
<style>
style rules.
</style>
</head>

<style>
h1 {color: purple}
p {color: blue}
</style>

(OR) You can use id attribute

use provide id="xyz" to the part you want to style

then in css file have a mention of

#xyz {color: red}

(OR) use class

<head>
fallback → .zappy {color: purple; background: pink}
.woo {color: blue; background: grey}
</head>
<body>
<h1 class="zappy"> I'm floating </h1>
 ...
 ...
</body>

(OR) Multiple class

HEAD

```
.zappy {color: blue}
.spicy {background: red}
```

BODY

 ...

(OR) INLINE STYLE

 ...

higher priority than general styles

context style

<head>
<style>
uo li { color: yellow }
</style>
</head>

color yellow will be applied only to unordered list

pseudo classes

- some kind of intelligence

h1: hover {color: red}

→ when mouse moves over any h1, text temporarily changes to red

→ There are many such pseudo classes. Some are

*) a: link { color: red }

*) a: visited { color: red }
↳ link that has been visited

*) a: active { color: red }
↳ link currently being followed

*) empty means an empty element
p empty { color: red }

TABLES

<table>
<thead>
<tr> <th> skills <th> <th> Difficulty </th> </tr>
</thead>
<tbody>
<tr> <td> HTML </td> <td> easy </td> </tr>
<tr> <td> CSS </td> <td> medium </td> </tr>
</tbody>
</table>
(fills row-by-row) (td) - table data.

SKILLS	Difficulty
HTML	easy
CSS	medium

some attributes / styles.
↳ color (- text color)
↳ text-align (- horizontal text alignt)
↳ background (- background color)
↳ vertical-align: (top / bottom / middle)
↳ width
↳ height

<head>
<style>
table, td, th { border: 1px solid black; padding: 15px }
↑ gap b/w tables columns

td { color: purple }

(OR) USE HTML EDITOR

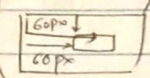
DIV & SPAN

For large area - <div>
For a few words - <span>

div has NO particular style (or) NO particular meaning
CAN be used for any purpose

↳ Absolute position

<div style="position:absolute; top: 60px; left: 60px">



↳ Relative position

<div style = position relative; top:-20px; left:-20px">

→ moves relative to what it's actual position would have been

span is similar to div

HTML FORMS

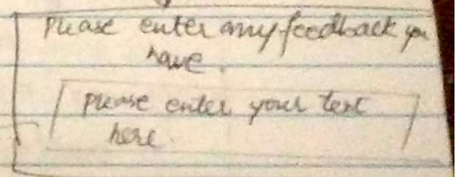
Basic structure
<form action="destination" method="get or post">
.... form elements
<input type="submit">
</form>

method "get"
you can see what U sent (maybe in URL)
but you can send only some upto 100 characters

method = "post"
- you canNOT see data (better for keeping secrets)
- can handle big transmission eg:- files.

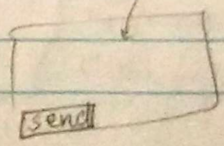
<form>
<p> Please enter any feedback you have. </p>
< textarea rows="5" cols="60" name="feedback">
Please enter your text here
</text area>
</form>

⇓



<form action="http://inane ... php" method=get>

<input type="submit" value="send"
</form>



Basics

Form input elements

Submit button <input type="submit">
Plain text "text"
Checkbox "checkbox"
radio button "radio"
password field "password"
File selector "file"

New HTML 5 input elements

<input type = "number"
"date"
color picker → "color"
slider → "range"
"time" >

<input type="checkbox" name="items"
value="car" > car
↑ ☐ car

USEFUL ATTRIBUTES

value = "something" fixes what is shown at the start
placeholder = "something" shows useful text which disappears when user enters something
autofocus sets which field is given focus at the start
required means this field is must.

Group elements

Grouping things <fieldset>
Giving a title <legend>

<form>
<fieldset>
<legend> Personal Info </legend>
First name <input type="text">
Last name
</fieldset>
<fieldset>
<legend> fav. things </legend>
fav. cartoon?
-u- pizza
</fieldset>
<input type "submit"
value = "Send" />
</form>

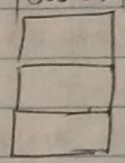
→ NOT necessary for forms
anywhere eg:- (div)

loading files → HTML side, server side

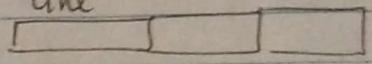
form action = "destination" method = "post"
enc type = "multipart/form-data">
↑ purpose format type

inline block

block



line



inline-block