Java Script

What is JavaScript?

* JavaScript was designed to add interactivity to HTML pages
* JavaScript is a scripting language
* A scripting language is a lightweight programming language
* JavaScript is usually embedded directly into HTML pages
* JavaScript is an interpreted language (means that scripts execute without preliminary compilation)
* Everyone can use JavaScript without purchasing a license

What Can JavaScript do?

* JavaScript gives HTML designers a programming tool - HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages
* JavaScript can react to events - A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element
* JavaScript can read and write HTML elements - A JavaScript can read and change the content of an HTML element
* JavaScript can be used to validate data - A JavaScript can be used to validate form data before it is submitted to a server. This saves the server from extra processing
* JavaScript can be used to detect the visitor's browser - A JavaScript can be used to detect the visitor's browser, and - depending on the browser - load another page specifically designed for that browser
* JavaScript can be used to create cookies - A JavaScript can be used to store and retrieve information on the visitor's computer

## JavaScript Functions and Events

## It is a common practice to put all functions in the head section, or at the bottom of the page.

*<html><head>*

*<script type="text/javascript">  
function displayDate()  
{  
document.getElementById("demo").innerHTML=Date();  
}*

*</script>*

*</head>*

*<body><h1>My First Web Page</h1>*

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

*<button type="button" onclick="displayDate()">Display Date</button>*

*</body>  
</html>*

## Using an External JavaScript

## Note: External script cannot contain the <script></script> tags!

## *<script type="text/javascript" src="xxx.js"></script>*

# JavaScript Statements

# JavaScript is Case Sensitive

* 1. A JavaScript statement is a command to a browser. The purpose of the command is to tell the browser what to do.
  2. This JavaScript statement tells the browser to write "Hello Dolly" to the web page:

document.write("Hello Dolly");

## JavaScript Blocks

## JavaScript statements can be grouped together in blocks.

## {

## document.write("<h1>This is a heading</h1>"); document.write("<p>This is a paragraph.</p>"); document.write("<p>This is another paragraph.</p>");

## }

## JavaScript Variables

## var x; var carname;

Note: When you assign a text value to a variable, use quotes around the value.

Note: If you redeclare a JavaScript variable, it will not lose its value.

# JavaScript Popup Boxes alert("sometext");

## Prompt Box prompt("sometext","defaultvalue");

# JavaScript For...In Statement

## var person={fname:"John",lname:"Doe",age:25}; for (x in person) { document.write(person[x] + " "); }

## Some JS String API:

## var str = 'blahblahblah\_123456789';

## // Get first part ("blahblahblah")

## str.slice(0, str.indexOf('\_'));

## // Get second part ("123456789")

## str.slice(str.indexOf('\_') + 1);

## // Get first part ("blahblahblah")

## str.substr(0, str.indexOf('\_'));

## // Get second part ("123456789")

## str.substr(str.indexOf('\_') + 1);

## // Get first part ("blahblahblah")

## str.substring(0, str.indexOf('\_'));

## // Get second part ("123456789")

## str.substring(str.indexOf('\_') + 1);

## // Get first part ("blahblahblah")

## str.split('\_')[0];

## // Get second part ("123456789")

## str.split('\_')[1];

## What is undefined and null?

## The value of a variable with no value is undefined (i.e., it has not been initialized). Variables can be emptied by setting their value to null

if (nullExample === null) { // executes this block only if null }

if (undExample ===Undefined) { // executes this block only if Undefined }

if (bothExampe == null) { // executes this block if Undefined or null }

## What are Cookies?

Cookies are data, stored in small text files, on your computer.

**Create Cookies (Change cookies same as create cookies):**

document.cookie="username=John Doe;";

// Default , cookies is deleted when the browser is closed.

**Create cookies with expiry date:**

document.cookie="username=John Doe; expires=Fri, 12 Jan 2014 20:00:00 GMT";

**Read cookies:**

var data = document.cookie;

To **Delete cookies** set the expires parameter to a passed date

If you **set a new cookie**, older cookies are not overwritten. The new cookie is added to document.cookie, so if you read docoment.cookie again you will get someting like:

cookie1=value; cookie2=value;

## What is Closure?

In JavaScript, if you use the function keyword inside another function, you are creating a closure.

e.g.

*function sayHello2(name) {*

*var text = 'Hello ' + name; // Local variable*

*var sayAlert = function() { alert(text); } // This is closure*

*return sayAlert;*

*}*

*say2 = sayHello2('Bob');*

*say2(); // alerts "Hello Bob"*

Closure summaries:

A closure is the local variables for a function - kept alive after the function has returned, or A closure is a stack-frame which is not deallocated when the function returns (as if a 'stack-frame' were malloc'ed instead of being on the stack!).

## What is Prototype:

## Prototype is used for inheritance:

## Example:

## var employee = function(fname,lname){

## this.fname = fname;

## this.lname = lname;

## this.fullName = function(){ return this.fname+" "+this.lname}

## }

## var manager = function(task){

## this.task = task;

## }

## manager.prototype = new employee("Pravin",'Sonawane');

## var m = new manager('take interview');

## alert(m.fname);

## alert(m.fullName());

***JavaScript does not support multiple inheritance.***

**Hoisting** is JavaScript's default behavior of moving all declarations to the top of the current scope (to the top of the current script or the current function).

In JavaScript, a variable can be declared after it has been used.

In other words; a variable can be used before it has been declared.

**Scenarios:**

1. **Find whether image is present on server**

*function checkImage(src) {*

*var img = new Image();*

*img.onload = function() {*

*// code to set the src on success*

*};*

*img.onerror = function() {*

*// doesn't exist or error loading*

*};*

*img.src = src; // fires off loading of image*

*}*

1. **Communicate between parent & child window**

From child window script select div using selector e.g. #myDiv

Width the help of **window.opener**

e.g. window.opener.document.getElementById(‘myDiv’);

From parent window access child window:

e.g. var winobj=window.open(..) // window.open() returns child window instance

winobj.document.getElementById(‘mychilddiv’);

1. **T**