Unit 12 Java Script Programming – 2

Structure:

12.1 Introduction

In previous unit, you have learnt core JavaScript programming concepts and JavaScript objects. JavaScript supports programming with objects. Objects are a way of organizing the variables that will help you to develop a dynamic web page. In this unit you will learn implementation of arrays, server side programming, cookies and finally event handling.

Objectives:

After studying this unit, you should be able to:

- describe external JavaScript files
- define properties of arrays
- describe role of server side JavaScript
- describe properties and methods of form
- list out different event handling object

12.1.1 Inline JavaScript

JavaScript program run from within a webpage (either HTML or XHTML document). That is, you type the code directly into web page code as a separate section. JavaScript programs contained within a webpage areoften referred as scripts. The <script> element tells the browser that the scripting engine must interpret the commands it contains. The type attribute of the <script> element tells the browser which scripting language is being used. You can assign the value "text/JavaScript" to the type attribute to indicate that the script is written with JavaScript.

Example:

```
<script type="text/JavaScript">
Statements
</script>
```

12.1.2 External Java script files

By using <script> tag you can add JavaScript to single page. But many times you'll create script that you want to share with all of the pages on your site. For example, you might use a JavaScript program to add drop down navigation menu to a web page. You want to use same fancy navigation bar on every page of your site, but copying and pasting the same JavaScript code into each page of your site is really difficult.

A better approach is to use external JavaScript file. It is similar like adding external CSS file to your webpage. An external JavaScript file is simply a text file that ends with file extension .js (ex: navigation.js).

The file only includes JavaScript code and is linked to webpages using the <script> tag.

Example: Adding JavaScript file navigation.js to your web browser, uses following code

```
<html>
<head>
<Script type=text/javascript" src=navigator.js"></script>
</head>
```

12.1.3 Implementing Arrays

An array is a collection of values called elements, such as an array of colors, an array of numbers, or an array of strings. Each element of array is accessed with an index value enclosed in square brackets.

There are three ways to construct an array.

- Through array literal.
- By directly generating an instance of an Array (using new keyword).
- Through the use of an Array constructor (using new keyword).

Array literal: An array literal in Javascript is a list of expressions that are each surrounded in a pair of square brackets, or'[] ', and each of which represents an array element. When a n array is constructed using an array literal, its length is set to the number of parameters supplied and its components are initialised with the specified values. A zero-length empty array is created if no value is provided.

Example:

```
var fruit = [ ];
var fruit = ["Orange", "Apple", "Banana", "Mango"];
var fruit = ["Orange", , "Mango"];
```

The first example creates an empty array. The second creates an array with four items. The array has two entries with values and one element that is empty in the centre. Orange, undefined, and mango are the values for fruits[0], fruits[1,] and fruits[2,] respectively.

Array constructor: The Array constructor's new keyword can also be used to create arrays. The length of the new array will be determined by a number parameter given within the parentheses.

Example:

```
var emp = new Array();

emp[0] = "Arun";

emp[1] = "Varun";

emp[2] = "John";
```

By providing the number of elements we need the array to hold as an input to the Array() object function , we can also determine the initial size of the array when we build. It is not required to mention at creation time the amount of elements you estimate the array will need because an array will grow automatically as new items are added.

Example:

```
var emp=new Array("Jai","Vijay","Smith");
```

In order to avoid explicitly providing a value, you must make an instance of the array by sending arguments to the constructor.

Like variables, arrays must be declared before they can be used. The new keyword is used to dynamically create the array object. It calls the array object constructor, Array (), to create a new array object. The size of the array can be passed as an argument to the constructor, but it is not necessary.

Any element of any array can be accessed by referring the index into that array.

Var mycolors = new array {“red”, “green”, “blue”, “violet”};

JavaScript does not directly support multi-dimensional arrays, but you can store any sort of data inside array elements, including other arrays. So you can get the behavior of multi-dimensional arrays by storing arrays within the elements of another array.

For example, the following code reads the values of table using arrays

```
   Var Maxcount =5;
   Var I, j;
  Var table = new Array (Maxcount +1);
For ( i=1; I <=Maxcount; i++)
{
table[i]= new table(Maxcount + 1 ); // creates columns in the table
For (j= 1; j <= Maxcount; j++) //creates rows in the table
{
 Table[j]= new table(Maxcount + 1);
}
}
```

12.1.4 Array Methods and properties

Whether you have an array of colors, names, or numbers, there are many ways you might want to manipulate the array elements. JavaScript provides a whole set of methods for adding name or color to beginning or end of the array, remove a number from the end of the array, or sort the elements.

Table 12.1: Methods of array

Method	Description
Concat()	Concatenates the elements passed into an array
Join(separator)	Joins together all the elements in an array to produce a string. If no separator argument is passed by default it takes comma as separator.
Pop()	Deletes the last element added to the array
Push()	Adds a new element to the end of the array
Reverse()	Reverse the order of elements in the array
Shift	Deletes the first element in the array and shifts the reaming elements down to fill the empty slot
Sort(0	Sorts the elements of an array into alpha or numerical order
Splice()	Inserts and removes the elements from an array
Unshift()	Adds an element to the front of the array

The main properties of the array objects are described in below table:

Table 12.2: Properties of array

Property	Description
Length	Contains the length of the array
Prototype	Provides a mechanism for JavaScript developers to add their properties to the array object.

The length property of array used to specify how many items are in the array and is automatically updated when you add or remove items to the array.

JavaScript arrays are a type of object used for storing multiple values in a single variable. Each value gets numeric index and may be any data type. Array like objects look like arrays. They have various numbered elements and a length property. But that's where the similarity stops. Array like objects do not have any of Array's functions, and for-in loops don't even work.

As you can create your own object or array by using the keyword new. You can also destroy a property or element in an array using the keyword delete.

Example: var name = new array (5);

Name= "Thomas";

Delete name (0);

In above example, an array element is destroyed by using delete keyword**.**

Self Assessment Questions

1. The___________attribute of the <script> element tells the browserwhich scripting language is being used.
2. JavaScript does not directly support multi-dimensional arrays, but you can store any sort of data inside array elements, including other arrays. (True/False)
3. You can destroy a property or element in an array using the keyword ________________.

12.2 Server side Java Scripting

A server side JavaScript language performs various types of processing or access databases. When a client requests a server side script, the script interacted and executed by the scripting engine with the web server software. After the script finishes executing, the web server software translates the results of script (such as the result of a calculation or the records returned from a database) into HTML, which it then returns to the client.

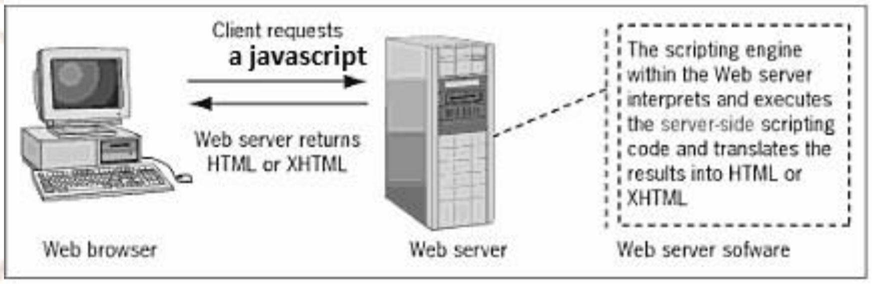
Client requests
a javascript
Web server returns HTML or XHTML
The scripting engine within the Web server interprets and executes the server-side scripting code and translates the results into HTML or XHTML
Web browser
Web server
Web server sofware

Figure 12.1: A web server process a server side script

It is the server-based application responsible for creating the content of web pages.

1) Using a database query
2) Database-related operations
3) Read/Write to a server file.
4) Communicate with other servers.
5) Define web application structures.
6) Deal with user feedback. Run a search algorithm on the data

stored on the server, for instance, if the user enters text in the search box, and send the results.

The benefits of server-side programming

- By executing scripts on the server, server side scripting lightens the load on the user's computer.
- Server side scripting can be used to construct database web applications.
- Server side scripting is used to hide scripts from users even if they have access to the source code; only client side scripts are displayed.
- With server side scripting, dynamic websites with content that can be modified at any time by the site administrator can be made fast.
- We don't need to care about browser versions because server side scripts are not browser-reliant.
- Server side scripting allows for the completion of complicated tasks in several steps.
- It is easy to use and comprehend.

The drawbacks of Server side programming

- It can be difficult to debug web server code.
- Scripting on web servers exposes users to hacker attacks.
- Running a web server can be difficult on a computer because it requires a significant amount of Memory, which slows down the system's performance.

12.2.1 Server Side Objects

The following four built-in objects can be used to access information from the server side in any server side JavaScript application.

1. Request
2. Client
3. Project
4. Server

Server side objects are also called session management objects. Each of the session management objects has different lifetimes and availability.Server object functions at the processing tier between the client and the data storage tier. They are used to access specific information from the processing tier and store the state information on the server.

Request object

The request object represents the current URL, request form a client. Each time a client request a particular URL, a new request object created. In addition, a request object is also created when client-side JavaScript uses the document. location (), history. go (), or the built-in redirect () method is used to forward a client request to different web page, which is quite similar to href property of the location of client side JavaScript.

Client Object

The user defined property of a client object can be used to store information regarding the items of a shopping cart or to preserve the field values of a form with multiple pages.

Project object

Often, the information contained in an application needs to be available globally so that it can be shared by all the clients accessing the application. Server side JavaScript uses a project object to globally store application specific information. A project object is created when an application is started by using the start button of the application manger. Thereafter, the project object for application is available to all clients accessing the application until the application is stopped by using the stop built in application manager.

Server Object

The server object created at server startup and is destroyed at shutdown. An environment contains multiple servers, each server is assigned an individual server object whose predefined properties can be accessed by all the applications on the server. Below table describes properties of server objects.

Table 12.3: Properties of server object

Property	Used to specify
Hostname	The full name of the host
Host	The name, subdomain, and domain name.
Protocol	The communications protocol of the server
Port	The port number being used to run the server
JsVersion	The version and platform used by the server.

12.2.2 Server-Side Program

In every computer language, dynamic web pages can be produced as long as it can run on the web server. As a result, there are many more options available than for client-side programming, which requires the use of JavaScript. Python, Ruby, PHP, or C# are frequently used to create server-side software. The majority of web developers are familiar with JavaScript, thus it is also a popular choice.

Example:

```
var http = require('http');
var server1 = http.createServer(function(req, res) {
  res.setHeader('Content-type', 'text/html')
  res.writeHead(200);
  res.write('<!DOCTYPE html>\n');
  res.write('<html><head><meta charset="UTF-8" />\n');
  res.write('<title>Node.js application</title></head><body>\n');
  res.write('<p>This is a page created by Node.js.</p>\n');
  for ( i=1; i<=10; i+=1 ) {
    res.write('<p>Another paragraph created.</p>\n');
  }
  res.write('</body></html>\n');
  res.end();
});
server1.listen(8080);
```

The HTML code is generated by the res.write() function. The server will transmit this HTML to the browser. A database or other source may include data that server-side code may access that a user's web browser cannot. Due to the availability of the whole spectrum of programming tools, dynamically produced web pages are significantly more powerful than static ones.

12.2.3 JavaScript Framework

A web development framework is an abstraction that enables extra user-written code to selectively alter software that provides generic functionality. A JavaScript framework is a JavaScript-written application framework that allows programmers to customise the functions and use them as needed.

frameworks are more flexible for online design, they are preferred by the majority of website developers. By providing options like making apps device responsive, frameworks mean working with JavaScript simpler and more streamlined.

The popular JavaScript frameworks are:

- Angular: An open-source framework for creating Single Page Applications, Angular is one of the most potent and effective JavaScript frameworks (SPA). Data binding is accomplished by extending HTML into the programme and interpreting the attributes.

- React: The React framework, was developed by Facebook, has become very well known. It is employed to create and manage the dynamic user interface of heavily visited web pages. As a result of the use of a virtual DOM, integration with any application is simpler.

- Ember.js**:** it was debuted in 2015, and thanks to its numerous applications, it has grown in popularity since then. Ember.js's features enable two-way data binding, which makes it a dependable platform for working with complex User Interfaces. Ember.js is used by well-known websites including LinkedIn, Netflix, and Nordstrom.

- Vue.js: This JavaScript framework was created in 2016, but it has already received attention in the market and established its value by providing a wide range of capabilities. One of its most alluring aspects for building a high-end SPA is its dual integration option. The platform is dependable for cross-platform application.

- Node.js: A cross-platform, open-source server-side JavaScript run-time environment is known as Node.js. The framework's event-driven architecture can be used to drive asynchronous Input/Output. It functions within the JavaScript Runtime environment and demonstrates Java's threading, packing, and looping capabilities.

- Polymer: The Google-developed open-source Polymer JavaScript toolkit.It can easily generate the basic elements of a webpage. Moreover, it allows one-way and two-way data binding, giving it a wider range of applications.

- Aurelia**:** The Aurelia framework is great for creating far more robust websites, while not being as popular as it once was. This Javascript framework may add features to HTML, such as data binding, for a variety of uses. Due to its cutting-edge architecture, tolls are always used for simultaneous client- and server-side interpretation.

- Backbone.js: Single Page Apps are made using Backbone, which is a very simple language to learn. This framework's development is based on the principle that all server-side operations must pass through an Application programming interface in order to achieve sophisticated functionality with minimal code.

Self Assessment Questions

4. Server side objects are also known as____________objects.
5. Which language is used to write Node.js?

12.3 Cookies and Event Handling

12.3.1 Cookies

Cookies are bit of information a site leaves on the hard drive of visitors. Because the information ends up on the hard drive, it remains after the user leaves the current page and even after the computer turned off. In this unit you will see basic cookie operations: setting cookie, reading cookie and setting expire date.

The escape () and unescape () function are used to code and decode the cookies.

Setting cookie

Setting cookie is simple. Create a string in the form cookie name,

cookie_name= value

and then set the *document. Cookie* property to the string. The only trick is that cookie values can't include spaces, commas, or semicolons.

Syntax:

```
Function setcookie()
{
Char the_cookiename = “username” +escape(the_name);
Document.cookie= the_cookie;
Alert(“thanks”);
```

}

Reading cookie:

Reading cookie is nothing but reading cookie that some where you saved in the hard disk.

Syntax:

```
Function readcookie()
{
```

```
Var the_cookie =document.cookie;
Var broken_cookie1 = the-cookie.split('=');
Var the_name= broken_cookie(1);
Var the_name= unescape(the_name);
Alert ("the cookie name "+ the_cookie);
}
```

Whenever your browser opens a web page, the browser reads whatever cookie that site has stored on your machine and loads into document.cookie property. The main purpose of reading cookie is getting just the information of cookie, which you want form that cookie.

Setting the duration of a cookie

Once cookie is created that disappear when user exits the browser. Sometimes this is for best. Since each domain can have only 20 cookies on a user machine, you don't want to waste space by saving unnecessary cookies between browser sessions. However, if you want your cookies to remain on a user's hard drive after user quits the browser, you have to set an expire date in UTC format.

To set your cookie to expire time, you have to add the expiration date to the cookie. (expire=date).

Example:

```
Var the_date = new Date("December 31,2013");
Var the_cookie_date = the_date.toUTCString ();
Var the_cokie+ the_cookie + "; expires = "+the_cookie_date;
```

The above code will help you to set the expire date of cookie.

12.3.2 Properties and methods of Form

The heart of web is the form. It is used to pass information from the browser to the server. Anytime you go online and order a book, trade an action, fill out a survey or sending an email using a web browser, you are working witha form.

An HTML offers number of ways to accept input, such as radio button, checkbox and textbox; these are called input valid controls. Once the form has been filled out by user, it normally is sent to server where the input is processed by server side program. The information normally sent from the browser to a server in a URL encoded format. Here, the major benefit of JavaScript comes into a picture, before sending the form off to the server, JavaScript can check to see if the form was filled out properly. If we like, every input field can be validated by JavaScript. It can check for emptyfields, valid credit card numbers, zip codes, and so on.

The JavaScript form object will help you when you need to access certain elements or attributes of the form in a script.

Properties

The form object's property provides information you might need when working with form in your script. Table12.4 describes the properties of the form object and their values.

Table 12.4: Properties of from object

Property	**Value**
Action	The value of the action attribute in the HTML form tag
Element	An array that includes an array element for each form element In HTML form
Encoding	The value of the enctype attribute, which varies with different browser
Length	The value of the total number of elements in an HTML form
Method	The value of the method attribute in an HTML form
Name	The value of the name attribute in an HTML form
Target	The value of the target attribute in an HTML form

Methods

The form has only two methods: reset () and submit().

Reset() : resets the form to its initial state, which means that all the entries and choices the user made get undone, and form shows the initial values defined in the value, selected, or checked attributes of the individual elements. Be aware of the difference reset() does not clear the form, but restores it to its initial state.

Submit (): submits the form.

12.3.3 Event Handling

Events are represented by an event object as a member variable of the window object, such as window. Event handlers are JavaScript code that are not include inside the <script> tags, but rather, inside the html tags, that execute JavaScript when some events fire, such as pressing a button,moving your mouse over a link, submitting a form, etc.

With an event handler you can do something with an element when an event occurs: when the user clicks an element, when the page loads, whena form is submitted, etc.

The basic syntax of these event handlers is:

name_of_handler="JavaScript code here"

Every element on an HTML page has events which can trigger a JavaScript.

For example, we can use the onClick event of a button element to indicate that a function will run when a user clicks on the button.

For example 1:

In the following example, the content of the h1 element will change when a user clicks on Distance Education:

```
<Html>
<body>
        <h1 onclick="this.innerHTML='SMUDE!'">
                Distance Education
        </h1>
</body>
</html>
```

There are various events that will be helpful for us to understand event handling in JavaScript.

1. **Onclick Event:** If the user clicks the left mouse button, the onclick event for an object occurs only if the mouse pointer is over the object, but if the user clicks the mouse on the object but moves the mouse pointer away from the object before releasing, no onclick event occurs.

   Syntax: <ELEMENT onclick = "handler(event);" >

   Event Property: object.onclick = handler

2. **Onabort Event:** This event is defined in HTML5. This event fires when the user aborts the download.

syntax: <ELEMENT onabort = "handler(event);" >

Event Property : object.onabort = handler;

3. **Onblur Event:** This event occurs when a user leaves an object or when a object loses the input focus. The onblur event fires on the original object before the onclick event fires on the object that is receiving focus. Where applicable, the onblur event fires after the onchange event.

   syntax: <ELEMENT onblur = "handler(event);" >

   Event Property: object.onblur = handler;

4. **Onchange Event:** This event fires when the contents of the object or selection have changed. For example if a user is typing in a textbox and after typing he/she commits, but if he/she wants to do some changes for the same this event will occur that time.

   Syntax: <ELEMENT onchange = "handler(event);" >

   Event Property: object.onchange = handler;

5. **Ondblclick Event:** This event fires when the user double-clicks the object.

   Syntax: <ELEMENT ondblclick = "handler (event);" >

   Event Property: object.ondblclick = handler;

   For example: if a user write something in the text box and wants to add that text in the given list then he/she can double click on the written text in the textbox to add that text in the given list.

6. **Onfocus Event:** This event fires when the object receives focus. When one object loses activation and another object becomes the active element, the onfocus event fires on the object becoming the active element only after the onblur event fires on the object losing activation. Use the focus events to determine when to prepare an object to receive input from the user.

   Syntax: <ELEMENT onfocus = "handler (event);" >

   Event Property:object.onfocus = handler;

7. **Onload Event:** This event fires immediately after the client loads the object. Attributes onload and onunload can only be used in <body> or <frameset>. In this event script to be run when a document load.

Syntax: <ELEMENT onload = "handler (event);" >

Event Property: object.onload = handler;

8. **Onunload Event:** This event fires immediately before the object is unloaded.

   Syntax :<ELEMENT onunload = "handler (event);" >

   Event Property: object.Onunload = handler;

9. **Onreset Event:** This event fires when a user resets a form.

   Syntax: <ELEMENT onreset = "handler (event);" >

   Event Property: object.onreset = handler;

   To invoke onreset event we can use two methods;
   (i) Click an input type=reset button.
   (ii) Invoke the reset method of the form object like onclick="form.reset()

10. **Onselect Event:** This event fires when a user selects content on a page.

    Syntax: <ELEMENT onselect = "handler (event);" >

    Event Property: object.onselect = handler;

11. **Onsubmit Event:** This event fires when a user submits a form. You can override this event by returning false in the event handler. Use this capability to validate data on the client side to prevent invalid data from being submitted to the server.

    Syntax: <ELEMENT onsubmit = "handler (event);" >

    Event Property: object.onsubmit = handler;

Following script demonstrates usage of these events in JavaScript.

```
<Html>
<Head>
<Title> script to demonstrate the usage of frequently used events </title>
</head>
<body onload= ‘alert (“example of JavaScript events”)’>
<h1 align=“centre“> JavaScript events </h1>
<ul>
<li> onblur event handler :
```

```
<input type="text" value="click here and see the result" onblur= 'alert ("NOT CLEAR")'>
<br>
<li> onclick event handler :
<input type="button" value="click here and see the  result" onclick='alert ("clicked")'>
<br>
<li> onChange event handler :
<input type="text" value="click here and see the result" onchange='alert ("Changed")'>
<br>
</ul>
</body>
 </html>
```

Keyboard events

Keyboard related events occur when a user presses a key while a web page is loaded. In addition to capturing the overall key Press event, you can separately capture the user's pressing the key and then releasing the key.

The following code links a tradition JavaScript function named disallowInput () to the onkeypress event handler associated with the document object.

<body onkeypress= "disallowInput ();" >

Below table specifies event handler that support keyboard events

Table 12.5: Keyboard related events

Event Handler	**Event(event handler triggered when** ….)
OnKeyDown	when user presses a key
OnKeyPress	The user presses and releases a key (which combines the onkeydown and onkeyup event handlers)
OnKeyUP	The user releases a previously pressed key.

Window events

One window event that most web surfers are familiar with is the pop-up advertisement a tiny window that appears automatically when you load certain web pages into your browser. Pop-up ads are attached to the onload event handler (and sometimes the OnUnload event handler, too).In addition to the OnLoad and OnUnload event handlers, windows and frames,which are a special type of window support event handlers including OnBlur,OnFocus, OnMove and OnResize.

Table 12.6: Window and Frame related events

Event handler	**Event (event handler triggered when…)**
OnBlur	The element loses input focus
OnFocus	The element gains input focus
OnLoad	The element loads successfully
OnMove	The user moves or resizes the window or frame
OnResize	The user resizes the window or frame
OnUnload	The user unloads a document (by closing the browser or by loading another document)

Mouse events

Mouse events make cool interactive effects such as rollovers and tooltips. No such object as mouse exists. Rather, mouse events occur when mouse pointer moves or is clicked over some other object.

Table 12.7 describes objects and event handlers associated with mouse events.

Table 12.7: Mouse related events

Event Handler	**Event (event handler triggered when…..)**
OnMouseDown	The user presses a mouse button(but don’t release it).
OnMouseOut	The mouse moves off the element.
OnMouseOver	The mouse moves onto the element.
OnMouseup	The user releases a previously clicked mouse button.

Handling a Form Event

A form event includes things like selecting a check box or submitting the form itself. Reacting to form events is one of the most things that JavaScript programmers do. Some form events are quite practical. The user must have a means of submitting the form, so handling the onsubmit event is essential. Other form events are nice, but not absolutely essential. For example onblur event lets you perform tasks such as asking the user about saving form data need. Table 12.8 shows the built in form events.

Table 12.8: Event handlers for the form elements

Object	**Event Handler**
Button	OnClick, OnBlur, OnFocus.
Password	OnBlur, OnFocus, OnSelect.
Radio	Onclick, OnBlur, OnFocus
Reset	OnReset
Select	OnFocus, OnBlur, OnChange
Submit	OnSubmit
Text/Textarea	OnClick, OnBlur, OnChange

Self Assessment Questions

6. The________and____________function are used to code and decode the cookies.
7. ________event occurs when a user leaves an object or object loses the input focus.

12.4 Summary

- JavaScript programs contained within a webpage are often referred as scripts. By using <script> tag you can add JavaScript to single page.
- JavaScript does not directly support multi-dimensional arrays, but you can store any sort of data inside array elements, including other arrays.
- A server side JavaScript language performs various types of processing or access databases.
- Server side objects are also called session management objects.
- Cookies are bit of information a site leaves on the hard drive of visitors.
- The JavaScript form object will help you when you need to access certain elements or attributes of the form in a script.
- Events are represented by an event object as a member variable of the window object, such as window.
- Keyboard related events occur when a user presses a key while a web page is loaded.

12.5 Terminal Questions

1. Describe array properties and methods?
2. What is server side JavaScript? Explain with a diagram.
3. Explain the concept of cookie?
4. Write a short note on following events
   A) Keyboard events B) Window events C) Form events

12.6 Answers

Self Assessment Questions

1. Type
2. True
3. Delete
4. session management
5. Javascript
6. escape (),unescape ()
7. onblur

Terminal Questions

1. JavaScript provides a whole set of methods for adding name or color to beginning or end of the array, remove a number from the end of the array, or sort the elements. For more detail refer section 12.1.4.
2. A server side JavaScript language performs various types of processing or access databases. For more details refer section 12.2.
3. Cookies are bit of information a site leaves on the hard drive of visitors. The basic operations of cookie are: setting cookie, reading cookie and setting expire date. For more details refer section 12.3.1.
4. Keyboard related events occur when a user presses a key while a web page is loaded. For more details refer section 12.3.3. Onblur(),Onfocus(), there are many more events for window. For more details refer section 12.3.3. Onsubmit event is more essential form event. For more form events refer table 12.8.

12.7 References

- Jennifer Niederest (2003). Learning web design: A beginning guide to HTML and JavaScript.