CODES

Capacitación Web – JavaScript

codes
CONSULTORIA
Y DESARROLLO DE SISTEMAS

CMMI

ÍNDICE

1. JAVASCRIPT – Conceptos a tener en cuenta

Request: Contiene los valores HTTP enviados por un cliente durante una solicitud Web

Response: Encapsula la información de la respuesta HTTP de una operación ASP.NET.

Context: Encapsula toda la información específica de HTTP acerca de una solicitud HTTP individual.

Session: El estado de sesión de ASP.NET permite almacenar y recuperar los valores de un usuario cuando el usuario explora diferentes páginas ASP.NET que conforman una aplicación Web. HTTP es un protocolo sin estado, es decir, el servidor Web trata cada solicitud HTTP de página como solicitud independiente; de forma predeterminada, el servidor no retiene información alguna sobre los valores de las variables que se utilizan durante las solicitudes anteriores. En consecuencia, la creación de aplicaciones Web que necesitan mantener la información de estado entre las solicitudes (aplicaciones que implementan carros de la compra, desplazamiento de datos, etc.) puede resultar complicada. El estado de sesión de ASP.NET identifica las solicitudes recibidas desde el mismo explorador durante un período limitado de tiempo como una sesión y proporciona la capacidad de conservar los valores de las variables durante la duración de esa sesión.

Cookies: Las cookies son pequeños archivos de texto que el servidor y el explorador intercambian en cada solicitud de página, y que se pueden usar para almacenar información que le ayudará a personalizar la aplicación para cada usuario.

CONCEPTOS

- Parámetros del request
- Javascript
- Acercamiento a JQuery

1. JAVASCRIPT - Introducción

Se trata de un lenguaje de tipo script compacto, basado en objetos y guiado por eventos diseñado específicamente para el desarrollo de aplicaciones cliente-servidor dentro del ámbito de Internet.

Los programas JavaScript van incrustados en los documentos HMTL, y se encargan de realizar acciones en el cliente, como pueden ser pedir datos, confirmaciones, mostrar mensajes, crear animaciones, comprobar campos.

Existen distintos modos de incluir lenguaje JavaScript en una página.
La forma mas frecuente de hacerlo es utilizando la directiva <script> en un documento HTML (se pueden incluir tantas directivas <script> como se quiera en un documento). El formato es el siguiente:

```
<script language="Javascript">
```

Puede incluirse también código JavaScript como respuesta a algún evento:

```
<input type="submit" onclick="alert('Acabas de hacer click');return false;" value="Click">
```

importante: JavaScript es sensible a mayúsculas y minúsculas, todos los elementos de JavaScript deben referenciarse cómo se definieron, no es lo mismo "Salto" que "salto“

En la mayoría de los casos, siempre trataremos de agrupar toda la funcionalidad JS en un archivo separado del HTML. La ventaja es que la página quedaría más organizada, además de la posibilidad de la reutilización de código JS. Invocaremos al archivo JS de la siguiente manera:

```
<script src="/Scripts/jquery.validate.min.js" type="text/javascript"></script>
```

1. JAVASCRIPT – Uso de variables

JavaScript reconoce los siguientes valores:

-Valores numéricos.
-Valores lógicos (true, false)
-Cadenas de caracteres
-Valor null
-Valor no definido (undefined)

Además de estos valores, existen otros elementos propios de JavaScript como los objetos y las funciones.

En JavaScript, los datos son tratados en forma dinámica, por lo que se pueden realizar las siguientes operaciones sin inconvenientes:

```
var unValor=50;
unValor="pepe";
```

Cuando a una variable no se le asigna un valor, tiene valor indefinido (undefined). Si se le pone un valor, pueden ocurrir dos cosas:

-Si fue declarada sin "var", se produce un error en tiempo de ejecución.
-Si fue declarada con "var", devuelve el valor NaN (Not a Number).

1. JAVASCRIPT – Uso de variables (II)

Ejemplos de uso de variables sin defínción:

```
function f1() { return y-2; }
f1() // Esta llamada a f1 provoca un error en tiempo de ejecución

function f2() { return var y-2; }
f2() // devuelve el valor NaN
```

Podemos utilizar el valor "undefined" para ponerle valor a una expresión:

```
var miVar;
if(miVar==undefined) {
hazunacosa(); }
else {
hazotracosa(); }
```

El valor "undefined" se comporta como falso cuando se usa como tipo booleano.

Las variables pueden ser:

- Globales: cuando se le asigna un valor fuera de una función. El uso de "var" es opcional.
- Locales: Se realiza la operación de asignación dentro de una función. El uso de "var" es obligatorio.

Por último, es bueno saber que se puede acceder a una variable de un documento HTML de un FRAME desde otro:

```
parent.miVar
```

La línea de código parent.miVar se refiere a la forma de acceder a una variable en un documento HTML cargado en un marco (frame) desde otro documento HTML que contiene ese marco.

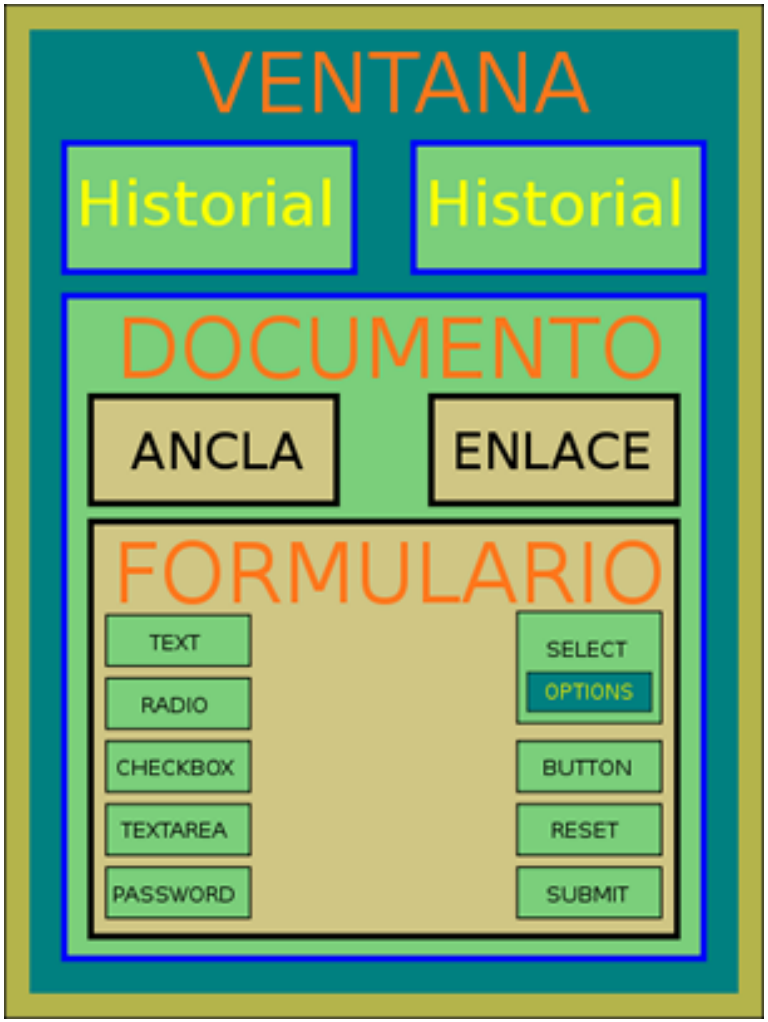
parent hace referencia al documento HTML "padre" que contiene el marco.
miVar es el nombre de la variable que se está tratando de acceder.
Por lo tanto, parent.miVar se utiliza para acceder y obtener el valor de una variable llamada "miVar" que está definida en el documento HTML del marco (frame) contenido dentro del documento HTML padre. Esto es útil cuando tienes una estructura de marcos (frameset) en tu página web y deseas interactuar o compartir datos entre los diferentes marcos.

Es importante destacar que para que esto funcione correctamente, la política de seguridad del navegador debe permitir el acceso entre marcos (frames), ya que, en algunos casos, por razones de seguridad, los navegadores pueden restringir estas interacciones.

DOM

Es una API que proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

Jerarquía de DOM se muestra en la siguiente imagen:



1. JAVASCRIPT – JQuery

jQuery es una biblioteca de JavaScript,que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web.

jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

Dentro del Visual Studio ya se encuentra disponible el uso de la librería jQuery.

Características

- Selección de elementos DOM.
- Interactividad y modificaciones del árbol DOM.
- Manejo de eventos.
- Manipulación de la hoja de estilos CSS.
- Efectos y animaciones.
- Animaciones personalizadas.
- AJAX.
- Soporta extensiones.
- Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
- Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari 3+, Opera 10.6+ y Google Chrome 8+.5

1. JAVASCRIPT – Uso de jQuery

jQuery consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM, eventos, efectos y AJAX. La característica principal de la biblioteca es que permite cambiar el contenido de una página web sin necesidad de recargarla, mediante la manipulación del árbol DOM y peticiones AJAX. Para ello utiliza las funciones $() o jQuery().

Función $()

La forma de interactuar con la página es mediante la función **$()**, un alias de **jQuery(),** que recibe como parámetro una expresión CSS o el nombre de una etiqueta HTML y devuelve todos los nodos (elementos) que concuerden con la expresión. Esta expresión es denominada selector en la terminología de jQuery.

Ejemplos:

```
$("#tablaAlumnos"); // Devolverá el elemento con id="tablaAlumnos"
$(".activo");       // Devolverá una matriz de elementos con class="activo"
```

Una vez obtenidos los nodos, se les puede aplicar cualquiera de las funciones que facilita la biblioteca.

Ejemplos:

```
// Se elimina el estilo (con removeClass()) y se aplica uno nuevo (con addClass()) a todos los nodos con class="activo"
$(".activo").removeClass("activo").addClass("inactivo");
```

O por ejemplo, efectos gráficos:

```
// Anima todos los componentes con class="activo"
$(".activo").slideToggle("slow");
```

Ejemplos de cómo obtener valores de elementos HTML:

```
// Obtiene el valor de un SELECT
$( "select.foo option:selected").val();

// Obtiene el valor de un checkbox seleccionado
$( "input:checkbox:checked" ).val();

// Obtiene el valor de un conjunto de radio buttons
$( "input:radio[name=bar]:checked" ).val();

//Establece un valor a un input
$( “input" ).val( ‘valor prueba’ );
```

Ejemplos de cómo manipular los estilos de elementos:

```
//Cambiamos el color del BODY a rojo
$( “body” ).css( "color", "red" );
alert($("body").css("background-color"));
```

Ejemplo de cómo modificar el HTML de un documento:

```
<div class="demo-container">
  <div class="demo-box">Viejo contenido</div>
</div>
$( "div.demo-container" ).html( "<p>Nuevo contenido HTML. <em>Con jQuery!</em></p>" );
```

NOTA: El tag define una división o a sección en un documento HTML. El elemento es muy usado junto con CSS para dar forma a una página y estructurar su contenido.

FIN

Gracias

codes
CONSULTORIA
Y DESARROLLO DE SISTEMAS

CMMI