Tema 10. Capa de Interfaz II

Controles de Lista Sencillos

listBox
dropDownUst
radioButtonList
checkBoxList
dropDownList; Lista desplegable
RepeatDirection
RadioButtonList: lista de botones de radio
CheckBoxList: lista de opciones múltiple

ListItem

Text:Contenido visualizado
Value:Valor oculto del código HTML
Selected:true o false (seleccionado o no)

SelectedIndex

Indica la fila seleccionada como un índice que empieza en cero

SelectedItem

Permite que el código recupere un objeto ListItem que representa el elemento seleccionado

Controles de lista con selección múltiple

ListBox

Pueden seleccionarse varios elementos: propiedad

CheckBoxList

Siempre pueden seleccionarse varios elementos
Para encontrar todos los elementos seleccionados necesitamos: recorrer la colección Items del control lista y comprobar la propiedad ListItem.Selected de cada elemento

Control de navegación menu

Se puede utilizar para crear un menú que colocaremos en la página maestra y otras ayudas de navegación. También podemos añadir submenús y definir menús dinámicos.
Los elementos pueden añadirse directamente en el control o enlazarlos con una fuente de datos.
En las propiedades podemos especificar la apariencia, orientación y contenido.

Static Display y Dynamic Display

El control tiene 2 modos de "Display":
Estático: El control Menu está expandido completamente todo el tiempo.
Dinámico: En este caso solo son estáticas las porciones especificadas, los demás elementos se muestran al hacer click sobre el padre.

StaticDisplayLevels

El número de niveles estáticos que queremos mostrar como raíz del menú (el mínimo es 1).

MaximumDynamicDisplayLevels

Cuantos niveles de nodos que aparecen de forma dinámica se mostraran después del nivel estático.

Dynamic: cantidad de tiempo

Podemos especificar la cantidad de tiempo que queremos que tarde la parte dinámica de un menú en desaparecer.
Lo podemos especificar en milisegundos mediante la propiedad DisappearAfter del menú, el valor por defecto son 500 ms.

Definición del contenido de menu

Añadir objetos individuales MenuItem
se le puede enlazar un archivo XML.
Propiedades del control propiedad Items (colección de objetos MenuItem)

Validación de datos

Debemos asegurar que los usuarios introducen datos correctamente
ASP.Net proporciona un conjunto de controles de validación predefinidos
2 tipos de validación:
Lado del cliente: Utilización de código JavaScript que valida los datos introducidos por el usuario, directamente en el navegador
Lado del servidor: Utilización de código (C#) para validar los datos de formularios una vez han sido enviados al servidor

Controles de validación

ASP.Net detecta si el navegador soporta validación del lado del cliente
Generan el código JavaScript necesario para validar los Datos, en otro caso, los datos del formulario se validan en el servidor
Para mostrar el mensaje de error ErrorMessage, para indicar el control a validar ControlToValidate

Entrada requerida: RequiredFieldValidator: La validación es OK cuando el control de entrada no contiene una cadena vacía.
Coincidencia de modelos: RegularExpressionValidator:. La validación es OK si el valor de un control de entrada se corresponde con una expresión regular especificada.
Comparación con un valor: CompareValidator: La validación es OK si el control contiene un valor que se corresponde con el valor de otro control especificado.
Comprobación de intervalo: RangeValidator: La validación es OK cuando el control de entrada contiene un valor dentro de un intervalo numérico, alfabético o temporal especificado.
CustomValidaton: La validación la realiza una función definida por el usuario.
ValidationSummarv: Este control muestra un resumen con todos los mensajes de error de cada control de validación.
Propiedad display: Sirve para comprobar si el control de validación muestra mensajes de error.

Sintaxis Expresiones Regulares

* cero o más ocurrencias del carácter o subexpresión anterior.
+ una o más ocurrencias del carácter o subexpresión anterior
() agrupa una subexpresión que se trata como un único elemento
| Cualquiera de las dos partes (OR)
[ ] se corresponde con un carácter en un intervalo de caracteres válidos [a-c]
{n} exactamente n de los caracteres o subexpresiones anteriores

. cualquier carácter excepto el salto de línea
? el carácter anterior o la subexpresión anterior es opcional
^ comienzo de una cadena
$ fin de una cadena
\s carácter de espacio en blanco \S cualquier carácter no espacio
\d cualquier carácter numérico
\D cualquier carácter no dígito
\w cualquier carácter alfanumèrico (letra, número o carácter de subrayado)

Grupos de validación

Los controles de validación se pueden asociar en grupos de validación para cuando sean comunes se validen juntos
se pueden usar para habilitar o deshabilitar de forma selectiva la validación
hay que definir el nombre del grupo en los controles de validación y en el botón u otros controles de envío que causan validación.

SetFocusOnError

Se establece en controles de validación que causan que el primer control no válido reciba el foco

Objetos Session Application

Los objetos **Session** están asociados a un usuario particular y sirven como manera de transportar y mantener los datos del usuario.
Los objetos **Application** son compartidos por todos los usuarios y permiten almacenar información compartida por toda la aplicación web.
En ASP.NET los objetos Session y Application están implementados como colecciones o conjuntos de pares nombre-valor.

Qué es una sesión?

Una sesión es el período de tiempo en el que un usuario particular interactúa con una aplicación web. Durante esta la identidad única se mantiene internamente.
Los datos se almacenan temporalmente en el servidor.
Una sesión finaliza si hay un timeout o si tú la finalizas.

Cuál es el uso de una sesión?

Las sesiones ayudan a preservar los datos entre accesos sucesivos.
Se hace gracias a los objetos de sesión.
Los objetos de Sesión nos permiten preservar las preferencias del usuario y otra información del usuario al navegar por la aplicación web.

Objeto Session

sirve para almacenar datos pertenecientes a un único usuario .
En asp.net: Las sesiones son tablas Hash en memoria con un timeout especificado.están identificadas usando enteros 32-bit long conocidos como Session IDs.
El motor ASP genera estos session ID’s que son únicos
Al crear la parte privada de la Web utilizaremos variables de sesión para controlar si el usuario ha entrado logueandose o ha entrado poniendo directamente la URL, en cuyo caso la variable de sesión estará vacía y no deberíamos permitir el acceso
¿Inicializar variables que estén disponibles en una sesión y sean las mismas para todos los usuarios? Esto supone que un cambio en el valor de una variable de aplicación se refleja en las sesiones actuales de todos los usuarios.

Session.Abandon l	Abandona (cancela) la sesión actua
Session.Remove	Borra un elemento de la colección de estado de la sesión.
Session. RemoveAII	Borra todos los elementos de estado de la sesión.
Session.Timeout	Establece el the timeout (en minutos) para una sesión
SessionID	Recupera el ID de la sesión (propiedad de sólo lectura de una sesión) para la sesión.
Session. IsNewSession	Es para comprobar que la sesión del usuario se creó con la petición actual

Objeto Application

Application: proporciona una manera sencilla de almacenar en el servidor datos comunes a todos los visitantes de nuestro sitio web.

Global.asax

Permite escribir código de aplicación global
no contiene etiquetas HTML ni ASP.NET
se utiliza para definir variables globales y reaccionar a eventos globales
contiene código de tratamiento de eventos que reacciona a los eventos de aplicación o sesión
se añade a la aplicación web como un nuevo elemento
Clase de aplicación global
Cualquier cambio en el archivo global.asax reiniciará la aplicación ,Sólo puede haber un archivo global.asax, debe estar en la raíz.