Licenciatura en Gestión Tecnológica

Programación Avanzada II

Formación Continua

Validación de Controles ASP.NET

Ing. Mariano Juiz

Agenda

1. Introducción
2. Controles de Validación

Validación de Controles: Introducción

Teniendo en cuenta que...

- Los usuarios son impredecibles en el uso de una aplicación
- Hay que evitar los fallos de las aplicaciones
- La aplicación, entre otras cosas, debe ser robusta.

Ejemplos:
- No se completa un campo obligatorio
- Se ingresan 2 contraseñas diferentes
- El formato del dni es incorrecto
- Etc.

Validación de Controles: Introducción

- Se añaden a un Formulario Web de la misma forma que el resto de los controles

- Hay controles para tipos específicos de validación

- Se puede asignar más de un control de validación a un control de entrada

- Son "invisibles" pero muestran Mensajes de Error

Controles de Validación

RequiredFieldValidator	CompareValidator
RangeValidator	RegularExpressionValidator
CustomValidator	ValidationSummary

Validación de Controles: Introducción

- Los controles de validación tienen propiedades en común:
  - runat="server" (control de servidor)
  - ControlToValidate (control a validar)
  - ValidationGroup (se utiliza para tener escenarios de validación diferentes en la misma página)
  - SetFocusOnError (hace que el primer control inválido reciba el foco)
  - ErrorMessage (mensaje a mostrar cuando ocurre el error dentro de un ValidationSummary o si Text no está definida)
  - Text (texto a mostrar en el lugar donde se encuentra el validador cuando ocurre el error, ejemplo "*")
  - Display (como se muestra el mensaje de error)
    - None (El mensaje no se muestra)
    - Static (Se asigna un espacio en el layout)
    - Dynamic (Se asigna un espacio dinámicamente si ocurre el error)
  - EnableClientScript (Habilita/Deshabilita validación en el cliente)
  - Enabled (Habilita/Deshabilita validación en el cliente y servidor)

Pero también tienen sus propias propiedades

Validación de Controles: Introducción

Validación de una Página

- Page.Validate (Evento que dispara la validación en el servidor)

- Page.IsValid (Propiedad que contiene si una página es válida o no)

- CausesValidation (si se setea en “true” no se ejecuta el evento si la página es inválida, si se setea en “false” no hay que olvidarse de disparar el método “Page.Validate” para validar la página)

Validación de Controles: Controles de Validación

RequiredFieldValidator

- Valida que se completen valores en un control (campo obligatorio)
- Si todos los controles son válidos (tienen valores) la página es válida

Validación de Controles: Controles de Validación

CompareValidator

- Compara los valores de dos controles
- Utiliza tres propiedades clave para realizar su validación
  - **ControlToValidate** (contiene el valor a comparar)
  - **ControlToCompare** (contiene el valor a comparar)
  - **Operator** (define el tipo de comparación a realizar)
  - **Type** (define el tipo de dato a comparar, String, Integer, Double, Date, Currency)
- Opcionalmente se puede utilizar **ValueToCompare** (reemplazando a ControlToCompare)
- Se puede utilizar para realizar la validación de Tipos de Datos (y su formato)

Validación de Controles: Controles de Validación

RangeValidator

- Valida que una entrada se encuentre dentro de un determinado rango
- Puede ser numérico o por cantidad de caracteres
- Utiliza tres propiedades clave para realizar su validación
    - **ControlToValidate** (contiene el valor a comparar)
    - **MinimumValue** (define el valor mínimo a comparar)
    - **MaximumValue** (define el valor máximo a comparar)
    - **Type** (define el tipo de dato a comparar, String, Integer, Double, Date, Currency)

Validación de Controles: Controles de Validación

RegularExpressionValidator

- Valida que una entrada coincida con un determinado patrón definido por una expresión regular

- Este tipo de validación nos permite comprobar secuencias predecibles de carácteres, como direcciones de e-mail, códigos postales, etc.

- Utiliza dos propiedades clave para realizar su validación

    - **ControlToValidate** (contiene el valor a comparar)

    - **ValidationExpression** (contiene la expresión regular con la que tiene que coincidir)

Validación de Controles: Controles de Validación

CustomValidator

• Llama a una función definida por el usuario para realizar validaciones que los validadores estándar no pueden llevar a cabo

• Se puede ejecutar en el servidor
  • Se setea la propiedad **OnServerValidate**

• Se puede ejecutar con un script del lado del cliente
  ○ Se setea la propiedad **ClientValidationFunction**
    ▪ La función debe tener dos argumentos (*source* que es el objeto CustomValidator y *arguments* que tiene dos propiedades: IsValid y Value

• Pueden utilizarse ambas al mismo tiempo

• Debe setearse la propiedad "**ValidateEmptyText**" (true) para que la validación personalizada se ejecute si el control a validar se encuentra vacío

Validación de Controles: Controles de Validación

CustomValidator – Ejemplo

```
<asp:CustomValidator ID="CustomValidator1" runat="server" ControlToValidate="txtExperiencia"
OnServerValidate="CustomValidator1_ServerValidate" ErrorMessage="CustomValidator"
ClientValidationFunction="valExperiencia" ValidateEmptyText="True">No califica</asp:CustomValidator>
```

Funcion de Validación en Cliente

```
function valExperiencia(sender, args) {
        var experience = args.Value.toLowerCase();

        if (args.Value == "") {
            sender.innerText = sender.errormessage = "*";
            args.IsValid = false;
        }
        else {
            //si el postulante fue despedido, no califica.
            if (experience.indexOf("despedido") != -1) {
                sender.innerText = sender.errormessage = "No califica";
                args.IsValid = false;
            }
            else {
                args.IsValid = true;
            }
        }
    }
```

Validación de Controles: Controles de Validación

Funcion de Validación en Servidor

```
protected void CustomValidator1_ServerValidate(object sender, ServerValidateEventArgs e)
{
    string experience = e.Value.ToLower();
    if (string.IsNullOrEmpty(experience))
    {
        CustomValidator1.Text = "*";
        e.IsValid = false;
    }
    else
    {
        CustomValidator1.Text = "No califica";
        if (experience.IndexOf("despedido") != -1)
        {
            e.IsValid = false;
        }
        else
        {
            e.IsValid = true;
        }
    }
}
```

Validación de Controles: Controles de Validación

ValidationSummary

- Se muestra cuando la propiedad IsValid de la página está establecida a “false”

- Verifica cada control de validación de la página y agrega el texto de error (ErrorMessage) que cada uno muestra

Licenciatura en Gestión Tecnológica

Programación Avanzada II

Formación Continua

Muchas Gracias

Ing. Mariano Juiz