دسته‌بندی Property های کنترلها

Property های هر کنترل به 5 دسته تقسیم می‌شود.

1 - دسته اول همه کنترلها دارند ولی نباید به عنوان آدم حرفه‌ای آنها را تغییر دهیم.

2 - دسته دوم بین دو دسته 1 و 3 هستند.

3 - دسته سوم همه دارند اگر لازم شد تغییر می‌دهیم.

4 - دسته چهارم بین دو دسته 3 و 5 است.

5 - دسته پنجم خاص یک کنترل است و به صورت تخصصی مربوط به همان کنترل است.

تکلیف 4 دسته از صفات این کنترل را مشخص می‌کنیم.

Property های نوع اول

همه کنترلها این Property ها را دارند و این Property ها بین تمام کنترلها مشترک است و ما نباید تحت هیچ شرایطی به عنوان یک آدم حرفه‌ای، مقدار آنها را عوض کنیم. (البته نه اینکه هیچ وقت عوض نشود، بلکه باید در CSS تنظیم شود)

1 - Back Color

2 - Border Color

3 - Border Style

4 - Border Width

5 - Font

6 - Fore Color

Property های نوع سوم

همه کنترلها آن را دارند و اگر بخواهیم می‌توانیم مقدار آن را تغییر دهیم:

1 - ID
2 - Access Key
3 - Css Class
4 - Enabled (با false کردن این property کنترل دیده می‌شود ولی کار نمی‌کند)
5 - EnabledTheming
6 - SkinID (این Property به property شماره 5 مرتبط است)
7 - Enabled View State (هنوز دلیلی پیدا نشده است که این Property به صورت False تنظیم شود. چون تنها مزیت کنترلهای Server Side داشتن View State است که اگر با false کردن این property آن را از بین ببریم، دیگر چه لزومی به استفاده از کنترل server side وجود داشت. در ضمن این صفت کار نمی‌کند مگر اینکه View State صفحه را هم false بگذارید که این هم اصلاً به درد نمی‌خورد.
8 - Tab index (به صورت پیش فرض Tabindex همه کنترلها 0 است، بنابراین ترتیب اجرای آنها مطابق HTML آن است و اگر بخواهید شماره می‌دهید. اگر tab index یک کنترل را 1- بگذارید، دیگر با زدن Tab داخل آن نمی‌رود. در زبانهای قبلی یک صفت به نام Tab Stop داشتیم که همین کار را می‌کرد و معادل 1- کردن Tab index است.
9 - Tooltip
10 - Visible (اگر false کنید در asp اصلاً render هم نمی‌شود. در Windows وقتی این صفت را false می‌کنید کنترل وجود دارد ولی در web اصلاً کد HTML آن Generate نمی‌شود.

کنترل Validator

در مبحث قبلی CMS بسیاری از Validation ها را در سمت سرور انجام دادیم. Validation یعنی که یک سری کنترلها را قبل از انجام عمل مورد نظر انجام دهیم. به عنوان نمونه قبل از اینکه یک فایل را کپی کنیم، کنترل اینکه آن فایل خالی نباشد و موجود باشد یک validation است.

می‌توان به صورت دستی عمل Validation را انجام داد با کد نویسی در سمت Sever بگوییم اگر این شد، این کار را بکن و ... ولی می‌توان بسیاری از Validation ها را به صورت javascript در سمت Client Side انجام دهم، با این کار بار سایت را کم کردیم.

JavaScript بسیار زبان سخت و غیر استانداردی است، بنابراین ما به سراغ JQuery می‌رویم البته خود Visual Studio یک زحمتی کشیده و یک سری Validator تولید نموده است.

Property های نوع دوم

دسته دوم بین دو دسته اول و سوم است، باید با درایت تشخیص دهید که باید تغییر یابد یا خیر.

1 - width

2 - Hight (گاهی با CSS به همه دکمه ها طول و عرض می‌دهید ولی گاهی در یک صفحه یک عکس دارید که می‌خواهید به آن طول و عرض دهید، دیگر لازم نیست که در Css این کار را انجام دهید و در خود صفحه Property کنترل را تغییر می‌دهیم. پس تشخیص آن با توجه به درایت برنامه‌نویس می‌باشد.

Property های نوع چهارم

این Property حدوداً بین 98٪ کنترلها مشترک است و این 98٪ کنترل این Property را دارند.

1 - Text (اکثراً دارند بنابراین نه داخل 5 می‌رود و نه 3 چون 98٪ کنترلها دارند)

Property های نوع پنجم

هر چیزی که باقی مانده است، دسته پنجم است، پس ما هر کنترلی که درس می‌دهیم فقط دسته پنجم را می‌گویم.

کنترل Required Field Validator

یک Lable در صفحه می‌گذاریم به نام lblMessage با runat سمت Server

یک Lable و یک Textbox هم با عنوان SomeField می‌گذارم.

یک دکمه Submit هم می‌گذاریم.

به طور سنتی بر روی دکمه دستور می‌نوشتید که خالی بودن را چک کن و ... را چک کن و ...

در این روش فقط پشت دکمه کارهای قطعی را انجام می‌دهید.

lblMessage.text = “Your information update successfully”

اولین مورد اینکه می‌خواهیم textbox خالی نباشد. که ماکروسافت کنترل Required Field Validator را برای این کار پیشنهاد داده است.

Some Field

(S)ubmit

```
<form id="frmMain" runat="server">
            <asp:label id="lblMessage" runat="server" />
            <br />
            <br />
            <asp:label id="lblSomeField" runat="server" text="Some Field" />
            <asp:textbox id="txtSomeField" runat="server" />
            <asp:requiredfieldvalidator runat="server" id="rfvSomeField"
                  controltovalidate="txtSomeField" display="Dynamic"
                  enableclientscript="False" setfocusonerror="True" >
                  Some field is required!</asp:requiredfieldvalidator>
            <br />
            <br />
            <asp:button id="btnSubmit" runat="server" text="(S)ubmit"
                        accesskey="S" onclick="btnSubmit_Click" />
</form>
```

حال صفتهای این کنترل را معرفی می‌کنیم.

1 - Controltovalidate (این صفت باید حتماً دارای یک مقدار باشد و حتماً هم نام یک کنترل در صفحه باشد. در غیر این صورت error می‌دهد. خودش یک DropDown list دارد که می‌توانید نام کنترل را انتخاب کنید. هر Validator ای باید حتماً یک و فقط یک کنترل را Validate کند. هر Validator ای ممکن است قوانین متفاوتی از custom Validator داشته باشد و قوانین اینچنینی که در مورد هر validator بیان می‌کنیم، ممکن است در مورد custom Validator صادق نباشد. به عنوان مثال اگر 50 تا فیلد داریم و بخواهیم هر 50 تا را کنترل کنیم که پر شده باشد، باید 50 تا Validator control بگذاریم. البته برعکس این قانون صادق نیست و هر کنترلی می‌تواند چندین Validator control داشته باشد که هر کدام یک کاری بر روی آن کنترل انجام دهد.

2 - ID (هر Validator مربوط به یک کنترل است و هر کنترل هم که یک ID داشت، Required Field Validator به صورت مخفف rfv است، پس ID این کنترل با rfv شروع می‌شود و سپس نام کنترل بدون گذاشتن abbrevation خود کنترل (مثلاً بدون txt و btn) قرارمی‌گیرد).

3 - دو تا صفت دیگر هم validator ها دارند که بعداً می‌گوییم. یکی error message و دیگری Validation group می‌باشد.

کار این Validation کنترل کردن خالی نبودن textbox بود، اینکه در صورت خالی بودن چه خطایی بدهد، باید خطا را در Text بنویسید.

پس برای کنترل خالی بودن یا نه، کد server side و c# ای نوشته نشد و فقط از java script استفاده شد و اصلاً هیچ کدی نوشته نشده است. اگر کنترل خالی باشد اصلاً صفحه Post back هم نمی‌شود.

البته این کنترل یک باگ دارد. این کنترل به طور طبیعی عمل نمی‌کند. یعنی همان اول هر چی Tab بزنید و بالا و پایین کنید، خطا نمی‌دهد. وقتی عمل می‌کند که یک بار بر روی دکمه submit کلیک کنید و یا اینکه یک بار داخل آن یک چیزی بنویسید و پاک کنید و خارج شوید، تازه با این کار این کنترل راه می‌افتد.

الان focus روی textbox است، وقتی دکمه submit را می‌زنم، focus بر روی دکمه منتقل می‌شود. اگر صفت set focus on error property را true کنید، focus بر روی textbox باقی می‌ماند.

از اول پیدایش net. یک سری تغییرات ایجاد شد.

1 - یک سری کنترلها کنار گذاشته شد و کنترلهای جدید آمد. قبلاً DataGrid داشتیم و الان DataGridView داریم.

2 - یک سری کنترلها نام آنها عوض شد مثل Menu Strip .

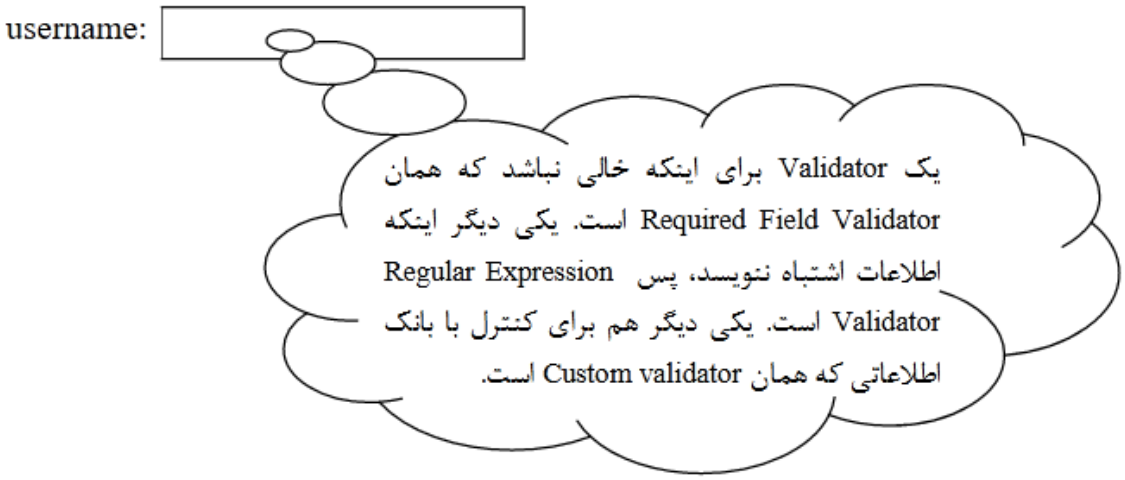
3 - یک سری کنترلها یک سری Property به آن اضافه شد. مثل set focus on error Property این از 2003 به بعد اضافه شد.

صفت InitialValue : یک صفت است برای ست کردن مقدار اولیه. البته این را استفاده نکنید، چون به صورت ثابت می‌نویسد، اگر خواستید در صورت خالی بودن چیزی بنویسد ، حتماً آن را با javascript ست کنید.

صفت Display است که اگر none بگذارید، متن خطا را نشان نمی‌دهد، یعنی Validation کار می‌کند ولی خطا را نمایش نمی‌دهد. غیر none آن یا static است یا Dynamic . یک کنترل می‌تواند چندین Validator داشته باشد. اگر به صورت استاتیک باشد، خطاها پشت سر هم گذاشته می‌شود و بین آنها بر اساس طول خطا فاصله می‌گذارد و قشنگ نیست. ولی اگر dynamic بگذارید، روی هم می‌اندازد و این درست است.
سوال: اگر چند تا خطا بر بخورد، چه کار باید کرد. این هم اشتباه است. Validator ها باید به صورت جامع، کامل و مانع استفاده شود.

property بعدی، EnableClientScript است. یعنی می‌خواهم سمت Client side و script ای عمل کنم. اگر false باشد، یعنی فقط می‌خواهم Post back شود و کنترل در سمت سرور و با کد #C ای انجام گیرد. اگر true باشد، هم در Client و هم در Server صورت می‌گیرد.
یعنی اگر true باشد و صد بار خطا داشته باشد، صد بار هم سمت client و هم سمت server اجرا می‌شود؟ نه اینطور نیست، تا خطا دارد فقط سمت Client اجرا می‌شود و بعد که درست شد، برای check نهایی سمت سرور می‌رود.
False کردن این property هیچ فایده‌ای ندارد. اینکه این صفت true باشد و javascript را disable کنید، معادل این است که این صفت را false کنید و javascript را بگذارید enable باشد. پس برای بررسی نکات امنیتی است.

حالا این صفت را false کنید تا فقط سمت سرور، اجرا شود. پس با زدن submit صفحه Post Back می‌شود.



سه نکته مهم در .NET

تا به حال 3 چیز مهم یاد گرفته اید

1. View State
2. Post Back
3. دکمه submit همیشه اتفاق می‌افتد. تابع Submit اگر نوشته شود، همیشه اتفاق می‌افتد.

If page.isvalid این در واقع دستوری است که همیشه در تابع submit بنویسید.

Server side validation بعد از Page load اتفاق می‌افتد.

پس فرآیندی که اتفاق می‌افتد.

1. Client Validation
2. Post Back
3. Page load
4. Server side Validation
5. function (آن چیزی که باعث Post Back شده است)

PageValid یعنی سمت سرور هیچ خطایی را detect نکند.
تمام چیزهایی که باعث Post back می‌شود، با چشم بسته بنویسید. هیچ فرقی نمی‌کند. پس همیشه در توابعی که بعد از PostBack صورت می‌گیرد، این if را بنویسید.
لزوماً دکمه و event click باعث Post Back نمی‌شود، چیزهای دیگری هم هستند که باعث Post Back می‌شوند.

چیزهایی که باعث Post Back می‌شوند

1. وقتی که بر روی Button رویداد click صورت گیرد.
2. وقتی که بر روی LinkButton رویداد Click صورت گیرد. (در مورد LinkButton عملیات open in New Window معنی ندارد)
3. وقتی که بر روی Image Button رویداد click صورت گیرد.
4. وقتی که بر روی DropdownList ، ListBox ، CheckBox List و Radio Button List ، رویداد Selectedindex Change اتفاق بیافتد.

یک ListBox دارید وقتی item آن را تغییر می‌دهد که نباید Post Back شود چون اتفاقی نیافتاده است. ولی اگر IspostBack آن را True کنیم، با تغییر item ، Post Back می‌شود.
Selectedindex Change به شرط true بودن AutoPostBack عمل می‌کند.

TextBox اگر autoPostBack آن true باشد با TextChanged عمل PostBack بر روی آن صورت می‌گیرد.

در خصوص check Box نیز با ValueChange آن به شرط درست بودن AutoPostBack ، PostBack انجام می‌شود.

صفحه Login و Registeration با کنترل Field Required Validator

برنامه نمونه: صفحه login را با کنترلهای مربوطه بنویسید.

```
public partial class Login : System.Web.UI.Page
{
	protected void Page_Load(object sender, System.EventArgs e)
	{
		divMessage.Visible = false;

		if (Page.IsPostBack == false)
		{
			Initialize();
		}
	}

	private void Initialize()
	{
	}

	protected void btnSubmit_Click(object sender, System.EventArgs e)
	{
		if (Page.IsValid)
		{

		}
	}

	private void DisplayInformationMessage(string message)
	{
		divMessage.Visible = true;

		lblMessage.Text = message;
		lblMessage.CssClass = "information";
	}
```

کنترل Regular Expression Validator

وظیفه این کنترل

این validator تست می‌کند که آیا متنی یا عبارتی که وارد نموده‌اید از فرمت درستی طبعیت می‌کند یا خیر. این validator فقط از نظر شکلی تست می‌کند و نه از نظر مفهومی.

چند مورد مثالی مختلف را تست می‌کنیم:

فرض کنید که می‌پرسید شماره موبایل شما چنده؟ می‌گوید 123 آیا قبول می‌کنید؟ خیر. ولی 09122788745 را قبول می‌کنید؟ بله

ولی اصلاً کاری نداریم که آیا این شماره موبایل واقعاً وجود دارد یا خیر و متعلق به آن فرد است یا خیر.

در خصوص آدرس سایت و ایمیل و غیره هم همینطور است. یعنی مثلاً x@y.z یک آدرس پست الکترونیک معتبر است، ولی ممکن است این آدرس ایمیل متعلق به فرد مورد نظر نباشد و یا اصلاً وجود نداشته باشد، اصلاً مهم نیست از نظر این validator این یک آدرس ایمیل معتبر است.

Property های این کنترل

- تمام کنترلهای validator یک صفت به نام Control ToValidate دارند و مشخص می‌کند که کدام کنترل را validate می‌کنند.
- Text شامل پیغام خطا : مثلاً You did not specify Valid Format
- Validation Group و ErrorMessage : مهم هستند و بعداً می‌گوییم.
- برای ID (اسم کنترل) یک مخفف rev را به نام کنترلی که می‌خواهیم Validate کنیم بدون txt, btn و ... اضافه می‌کنیم.

دو کار در برنامه‌نویسی تحت وب معادل ریاضت کشیدن است،

1 - استفاده از javascript

2 - نوشتن یا تولید Regular Expression

در خصوص نوشتن Requlr Expression خیلی کم مواردی پیش می‌آید که بخواهید بنویسید، بخاطر اینکه خود محیط Visual Studio بسیاری از موارد را به صورت template برای ما گذاشته است و در اینترنت هم پیدا می‌شود و بعضی از سایتها هم هستند که requlr Expression را ارائه می‌کنند. یا در موارد ساده می‌توانید با ایجاد تغییر در template های موجود template مربوط به خود را بسازید.

در یک پروژه بزرگ تعداد Requlr Expression بیش از 15 تا نیست.

راهکارهای پیشنهادی جهت نوشتن Requlr Expression

1- استفاده از خود templateهای از پیش تعریف شده microsoft

2- جستجو در اینترنت

3- استفاده از سایت ها یا برنامه هایی در اینترنت که قدم به قدم می پرسد و در نهایت expressionرا تحویل می دهند.

4- استفاده از expressionهای موجود و ایده گرفتن از آنها

نکته : دانستن یا داشتن یک سری از expression برای سرچ کردن در یک متن 5000 صفحه ای مثلا یافتن هر چی ایمیل است مفید است.

نکته: خیلی از این expression ها ساده است. مثلاً \d{5} یعنی یک عدد 5 رقمی.

فرض کنید که یک فایل word دارید که 5 صفحه است. اگر در فایل word بگوییم هر چه email است را پیدا کنید، چگونه اینکار را انجام می‌دهید. مفهوم expression یک مفهوم استاندارد است. می‌پرسد می‌خواهید فلان چیز را پیدا کنید و یا expression را پیدا کنید، اگر در قسمت Expression این خط را بنویسید، می‌رود و آن را پیدا می‌کند.

اگر بخواهم در متن آدرس هر چی سایت است را پیدا کنید یا هر چی عدد 5 رقمی است را پیدا کنید، می‌توانید از همین عبارات برای search در محیط های استاندارد Microsoft استفاده کنید.

/d{3,5} عدد 3 یا 4 یا 5 رقمی را می‌پذیرد.

کنترل Compare Validator

وظیفه این کنترل

برای مقایسه محتویات دو Text Box است.

Property های این کنترل

- ID :برای نام گزاری مخفف آن cmv است.
- controltocompare نام کنترلی را که می خواهیم با controltovalidate مقایسه کنیم.
- یک صفت اضافه به نام ValueToControl دارد، که می‌توان بجای کنترل با Value ثابت مقایسه کرد.
- یک صفت دیگر به نام type وجود دارد. در مقایسه string ای 11 کوچکتر از 2 است ولی در مقایسه عددی برعکس است. پس باید type تعیین شود.
- صفت operator دارد و می‌پرسد که می‌خواهد عمل مقایسه چه باشد، Greater ، equal و ...

هر کدام validator مستقل هستند و نیاز به حضور دیگری ندارد.

در خصوص این کنترل نیز هم اگر در کنترلی که به عنوان controlToValidate مشخص نموده‌اید، چیزی ننویسید دیگر خطا نمی‌دهد یعنی در خصوص ConfirmPassword ، password اگر Password را پر کنید و confirm آن را خالی بگذارید، خطا نمی‌دهد . چون قرار است confirm را اعتبار سنجی کند و اگر این کنترل متفاوت از pass بود خطا دهد ولی وقتی چیزی ننوشتید که دیگر خطا نمی‌دهد.

کنترل Range Validator

وظیفه این کنترل

فرض کنید که می‌خواهید سن فرد را برای استخدام بگیرید.و محدودیت سنی داشته باشید

Property های این کنترل

- ID: rgv مخفف این کنترل است.
- Text پیام خطای آن است. مثلاً Range Check Error .
- چند صفت خاص دارد. Min آن مثلاً 25 و Max آن 35 باشد.

در مورد اعداد می‌تواند double , integer باشد. مثلاً فرد می‌تواند 25.5 بزند.

- typeیاچگونگی مقایسه
- CultureInvarientvalue در windows Application وقتی که کنترل را right to left می‌کنیم. و 25 می‌نویسیم، 25 رافارسی نشان می‌دهد. این صفت را وقتیکه true می‌کنید، یعنی اینکه اگر کد حروف متفاوت است، اول آن را به انگلیسی تبدیل کند و بعد آن را در range مناسب مقایسه کند.

برای Compare validator هم بهتر است این Property را ست نمایید به true.

Validator ها را جامع و مانع و کامل استفاده کنید. برای Confirm ها باید Regular Expression ها را حذف کنید. چون در صورت نا معتبر بودن مثلاً Password ، دو بار error می‌دهد.

برای Confirm ها Regular Expression را ننویسید. اصولاً سعی کنید، به گونه‌ای از validator ها استفاده کنید، که هیچوقت دو خطا با هم ظاهر نشود.

- در خصوص validator custom می‌توانید control To Validate را بدهید یا ندهید. عرف این است که اگر به یک کنترل مربوط می‌شود، نام ببرید و اگر به بیش از یک کنترل مربوط می‌شود، نام نبرید.

پس اگر خواستید password , username را با هم چک کنید ، چون دو تا است، نباید Control to validate را مقدار دهید ولی اگر فقط username را کنترل می‌کنید، بنویسید. این یک تاکتیک است ولی اجبار نیست.

- در خصوص custom ها شما باید event سمت سرور آن را raised کنید. onservervalidate نام این event است.

```
<asp:customvalidator id="csvEmail" runat="server" cssclass="validator"
controltovalidate="txtEmail" display="Dynamic" setfocusonerror="true"
text="Email is already exist! Please choose another one..."
validateemptytext="false" onservervalidate="csvEmail_ServerValidate" />
```

در وب یک سری احمق داریم. شما در دوره سوم Xml Web Service خلق Event را یاد می‌گیرید. در دوره visual studio فقط استفاده از آن را یاد می‌گیرید.

کنترل Custom Validator

وظیفه این کنترل

فرض کنید می‌خواهید Registeration بنویسید و با database مقایسه کنید که درست است یا خیر. کلیه validator هایی که Microsoft نوشته را الان درس دادیم. هر کدام را نتوان با این Validator ها انجام داد باید با custom Validator بنویسیم.

99٪ اوقات Custom validator های ما چون با database کار می‌کنیم، Server Side است.

Property های این کنترل

- ID را با مخفف csv نام‌گذاری کنید.
- Set focus on error را به true مقدار دهیم و ...
- ترجیحاً صفت ValidateEmptyText را false نگاه دارید تا سیاستهای ماکروسافت را ادامه دهید.
- قسمت EnableClientScript را هم False بگذارید. چون اولاً javascript بلد نیستیم و ثانیاً بیشتر کار دیتابیس است و باید سمت سرور باشد. ولی اگر بخواهید true بگذارید باید یک تابع javascript بنویسید و نام آن را در اینجا بنویسید.

آنجایی که خلق event را یاد می‌گیرید، می‌بینید که لازم نیست که برای هر Event دو پارامتر بگیرید، می‌توانید یک پارامتره، شش پارامتره و 7 پارامتره

البته همانجا یاد می‌گیرید که اگر دو پارامتره باشد، خیلی استاندارد است.

اگر یک نفر از شما پرسید که signature این تابع چیست؟ غلط است که نام متغییرها را بکار ببرید، آنها فقط دکور هستند. مثلاً اگر کسی از شما پرسید که signature پشت submit چیه؟ باید بگویید object , system نه sender , e .

غلط:

```
protected void csvPassword_ServerValidate(object source,
System.Web.UI.WebControls.ServerValidateEventArgs args)
```

درست:

```
protected void csvPassword_ServerValidate (object sender,
System.Web.UI.WebControls.ServerValidateEventArgs e)
```

پس از عرفیات خارج نشوید، اینها عرف است. به هر دوره وب و یا ویندوز برگردید. اسم این دو متغییر sender و e است.

استاندارد نویسی و بحث تاکتیکی گاهاً از تکنیک بهتر است.

آیا این e به درد بخور است یا خیر؟ در اینجا به هیچ درد نمی‌خورد. خداییش دکور است و به هیچ درد نمی‌خورد. اگر از نوع eventArgs باشد به هیچ درد نمی‌خورد و اگر پیشوند و پسوند داشت به یک دردهایی می‌خورد و .e بزنید یک چیزهایی در ارتباط با آن می‌آورد.

اگر control to validate را معادل نام کنترل مورد نظر گذاشته باشید. e.Value
معادل نام(محتوی) کنترل می‌شود.

```
String strUsername = e.value;
```

ولی اگر نگذاشته باشید هم مهم نیست. به این صورت بنویسید

```
String strUsername = txtUsername.text;
```

پس مقدار را از خود Textbox خوندم و دیگر نیازی به E ندارم. پس برای همین بود که مهم نبود.

مهم: به هیچ عنوان e.isvalid را نباید true کنید، فقط صرفاً اگر مشکلی وجود داشت آن را False کنید. صفحه‌ای را در نظر بگیرید که 5 تا validator دارد. از 5 تا 2 تا آن به مشکل خورده است و آخرین validator یک custom است. و در if آن نوشته بودم که اگر این validator به مشکل نخورد true کن و به این ترتیب کل validator های صفحه و کارهایی که کرده بودند را از بین بردیم و همه چیز کشک شد. پس با true کردن آن نتیجه کلیه validator ها را True می‌کنید.

یک تور بر روی مباحث مطرح شده:
Custom یک صفت داشت که تا خالی است کنترل نکنم. و ما آن را انتخاب نمودیم.

صفت validation group

- اگر در صفحه ای دو سری validation group داشته باشید، به هم گیر می‌دهد. در Visual Studio 2005 به بعد این باگ بر طرف شد. در قبل از آن اگر قرار بود صفحه ارسال یا reset پسورد را بگذارند آن را در یک صفحه دیگر می‌گذاشتند.

زمانی که بیش از یک ناحیه در صفحه داشته باشید، validation group به ما کمک می‌کند. برای این کار برای validation ها و دکمه ها validation group تعریف کنید. حواستان جمع باشد که textbox ها هم Validation group دارند ولی آن را به هیچ عنوان set نکنید چون دچار مشکل می‌ترکد.

اول در ناحیه validation group عملیات client side validation صورت می‌گیرد و بعد Post back و بعد Page Load و بعد server side validation در ناحیه validation group صورت می‌گیرد و بعد خود تابعی که باعث post back شده است.

اگر نخواهید پروژه را چند لایه بنویسید، شیک‌تر است که Validation را به این صورت که امروز درس داده شده است، عمل کنید. ولی در مدل چند لایه که business جداست اصلاً معنی Custom Validation حذف می‌شود. اگر لایه business ندارید بهتر است که یک کاسه کنید.

کنترل Validation Summery

وظیفه این کنترل

Validation Summery پیامهای خطاهای (errormessage)مربوط به Validation را در هنگام بروز خطا را دسته‌بندی و به همراه یک متن نمایش می‌دهد.

می‌توان به جای text برای خطا ها تصویر گذاشت.

Error messageپراپرتی در validation summery استفاده می‌شود و کاربرد آن این است که خطاهایی که در کل اتفاق افتاده را نشان می‌دهد.

بهتر است validation summary را در ابتدا و بالای صفحه بگذارید.

Property های این کنترل

- **DisplayMode property**

یک propertyدارد که نوع نمایش خطا ها است که سه نوع است :

- **list**
- **singleparagraph**
- **bulletlist**

که bulletخوب است که هر خطا را در یک پاراگراف نشان می دهد وبا یک شکل زیبا نیز نمایش می دهد.

- **Header Text**

معمولاً برای validation summery یک صفت به نام title وجود دارد. که آن را ست می‌کنند.

- **show message box**

showmessagebox را false کنید.

نکته: Tooltipو Alert java scrip را فارسی ننویسید

دو چیز را هیچ وقت فارسی نکنید،alert مربوط به java script و Tooltip فارسی.

می‌توان یک div گذاشت و با position و ... کاری نمود که به صورت مجازی درست نمایید.

Tooltip فقط روی windows فارسی جواب می‌دهد. ولی tooltip و alert مربوط به javascript در سایر سیستم عاملها فارسی دیده نمی‌شود و به صورت مربع و علامت سوال می‌شود.

```
<asp:ValidationSummary ID="RegisterUserValidationSummary" runat="server"
CssClass="failureNotification"   ValidationGroup="RegisterUserValidationGroup"
headertext="notice" showmessagebox="False" />
```

مفهوم User Control

از قرار گرفتن چند کنترل در کنار هم که به اصطلاح مفهوم جدیدی را تولید کند به آن user control می‌گویند. مثلاً می‌خواهید تاریخ شمسی داشته باشید. پس می‌توانید یک textbox و بعد از آن یک lable اسلش و بعد دوباره textbox و بعد lable و در انتها combobox و .. اگر بلد نباشید user control درست کنید هر بار که بخواهید تاریخ شمسی داشته باشید باید تمام این کارها را تکرار کنید. ولی اگر همه این کارها را در کنار هم بگذارید و یک موجود جدید درست می‌کنید و آن را صدا می‌کنید.

یک سایت بورس درست می‌کنید و یک کنترل به نام Currency Control

یک بحث به نام inheret control ها دارید و به یک کنترل یک سری ویژگی می‌دهید و در این درس کاری با آن نداریم در اینجا با کنار هم گذاشتن کنترلها یک کنترل جدید یا مفهوم جدید ایجاد می‌کنیم.

عرف بر این است که اول یک فولدر به نام UserControls درست کنید و داخل آن یک عدد صفحه از نوع web user control اضافه کنید نام آن را CurrencyConvertor می‌گذاریم.

User control تحت وب بسیار شبیه بهUser control در windows application است و خالی است و اجازه می‌دهد که هر چه بخواهید در آن بگذارید.

در این صفحه یک textbox به نام txtDollars با طول 10و بعد از آن یک textbox دیگر با نام txtRials درست کنید و سپس یک دکمه با نام btnSubmit اضافه کنید.
در txtbox دوم نباید چیزی تایپ شود، پس یا باید read only آن را ست کنید و یا enabled آن را false کنید، ولی به دلیل زیبایی الان enabled آن را false می‌کنیم.

❖ container

در ویندوز based 5 نوع container بیشتر نداشتیم یعنی کنترلهایی که درونشان کنترلهای دیگری می شود انداخت و 5 تا بیشتر نیستند:

- ❖ Form
- ❖ panel
- ❖ user Control
- ❖ tab control
- ❖ group box

❖ span و div

❖ span و div

در وب ما container زیاد داریم. تگی که در آن می‌توان تگهای دیگر نوشت، container است، هم div , p و h1..h6, span.. container هستند ولی دو تگ هستند که اصالت container آن بیشتر است و این دو تا div و span هستند که معمولا وقتی چیزهایی را بخواهیم مدیریت کنیم یا زیبا کنیم از این دو استفاده می کنیم.

- فرق div , spam چیست؟

تگها از نگاه دیگر به دو دسته تقسیم می‌شوند یا **in line** هستند این تگها شما را سر خط نمی‌برد و پشت سر هم هستند، مثل b ,i,u و span هم همینطور است.

تگهای **block** شما را به سر خط می‌برد. مثل div

در خصوص user control ها بهتر است از span استفاده کنید اگر در یک خط چند تا تاریخ شمسی داشته باشید، با div نمی‌توانید ولی با span می‌توانید. برعکس آن می‌توانید یک span را در یک خط نگاه دارید و دنبال آن چیزی بنویسید.

```
<span style="background-color: Yellow; color: Blue; border: 1px outset Gray; margin:
5px; padding: 5px;">
       <asp:textbox id="txtDollars" runat="server" maxlength="10" />
       <asp:textbox id="txtRials" runat="server" enabled="false" />
       <asp:button id="btnSubmit" runat="server" text="..." onclick="btnSubmit_Click"
validationgroup="CurrnecyConvertor" />
</span>
```

چند style مفید برای قسمت Main صفحه

Style چپ به راست و راست به چپ

Style ای که در آن یک ناحیه را فارسی یا انگلیسی کنیم. البته بهتر است بگوییم Ltr , Rtl کنیم و نگوییم فارس و انگلیسی.
چون می‌خواهیم به درد هر زبانی که از راست به چپ و یا چپ به راست است، بخورد به آن rtl. , ltr. می‌گوییم.
طبق آخرین استاندارد ماکروسافت وقتی که نام namespace و folder بنویسیم تمام حروف آن را با حروف بزرگ می‌نویسیم.
اگر اسم کلاس یا متغییر abbreviation است فقط حرف اول را بزرگ می‌نویسیم.
اگر استاندارد Microsoft بود، باید L مربوط به ltr را بزرگ می‌نوشت.

CSS برای چپ به راست به صورت ذیل تنظیم می‌کنیم:
Font family : verdana باشد و font size 8 یا 10 باشد. text align را Left استفاده کنید. برای زبانهایی که حالت چسبیده ندارند به هیچ عنوان Justified استفاده نکنید و direction را Ltr بگیرید.

CSS برای راست به چپ به صورت ذیل تنظیم می‌کنیم:
direction:rtl; و text-align: justify; و font-size :10 و font family : Tahoma .

```
.ltr
{
	direction: ltr;
	text-align: left;
	font-size: 10px;
	font-family: Verdana;
}

.rtl
{
	direction: rtl;
	text-align: justify;
	font-size: 10px;
	font-family: Tahoma;
}
```

دو فونت arial , Tahoma به صورت multi language است و وقتی آن را باز می‌کنیم تمام زبانهای دنیا حتی چینی و ژاپنی را هم support می‌کند. Tahoma قشنگتر است.

اگر یک ناحیه می‌خواهید هم right to left باشد و هم border باشد، اینها را در یک style sheet نگذارید، چند style sheet تعریف کنید و بعد به صورت ترکیبی استفاده کنید. اصلاً در هم آن قشنگ نیست. پس سعی کنید که کلاس‌بندی مرتبی داشته باشید و خصوصیات شبیه به هم را در کنار هم قرار دهید.

```
.withBorder
{
	margin: 10px;
	padding: 10px;
	border: outset thin;
}
```

پس یک کلاس به نام withboarder ایجاد نمایید. و در آن بگویید که padding , margin آن 4px باشد و boarder آن outset باشد و thin و ...

Style مربوط به Hyperlink

در خصوص hyperlink: در stylesheet چیزی به نام **pseudo** داریم که برای مرتبط کردن یک سری مفاهیم به یک سری event است. تلفظ آن zoodo است.

اینکه حرف اول را بکشد و بقیه حروف جملات بعدی را با آن پر کند. همه اینها pseudo است ولی مهمترین آنها

1. **link**
2. **visited**
3. **hover**
4. **active**

است که ترتیب آنها موقع نوشتن مهم است.

در Tahoma وقتی که از Underline استفاده می‌کنید، یک خط روی نقطه ها می‌اندازد، بنابراین در هنگام گذاشتن link، خوانایی متن کم می‌شود. برای از بین بردن آن باید چه کار کرد؟

Visited ها در سایت ما بهتر است که همان آبی باشد، در خصوص سایتهایی که خیلی لینک زیاد است، باید لینک ویزیت شده رنگ دیگری داشته باشد. آخرین لینکی هم که زدیم بهتر است قرمز باشد، تا بدانیم آخرین لینکی که کلیک کردیم کدام است. (active) .

در بعضی از سایتها لینکها خط ندارد و وقتی رفتین روش خط می‌گذارد . صفتی که زیر متنها خط underline می‌کشد، اسم آن **text- decoration** است.

در تمام قسمتها **text-decoration را none** می‌زنم و بعد **Border** تعریف می‌کنم و می‌توان مکان border را تعیین کرد و فقط boarder-bottom را تعریف می‌کنیم و به رنگ آبی #0000FF .

کنترل‌های Server Side

هر چیزی که در صفحه aspx به صورت HTML بنویسید عیناً به سمت client منتقل می‌شود. یک سری موجودات Server side داریم که آنها ابتدا تبدیل به HTML می‌شوند و بعد به سمت client می‌روند که به این عملیات تبدیل شدن، render شدن می‌گویند.

به ارسال داده از صفحه‌ای به سمت سرور Post back می‌گویند. در صفحات web based بعد از کلیک بر روی دکمه submit بلافاصله textbox ها پاک می‌شود که این از بزرگترین معضلات وب است. ولی net. آن را حل کرده است. در تکنولوژی و محیط غیر net. اگر تعدادی textbox داشتید و دکمه submit می‌زدید کل textbox ها پاک می‌شد و اگر می‌خواستید که textbox ها مقدار خود را حفظ نمایند، شما باید می‌رفتید کلی کد می‌نوشتید، تا این اطلاعات را در جایی نگهداری و مجدداً textbox ها را پرکنید. بنابراین net. موجودات server side را ایجاد نمود. این موجودات server side مخصوص net. می‌باشد.