فصل پنجم :
بررسی و تعیین اعتبار داده های وارد شده از طرف کاربر و موارد تکمیلی کنترل های وب

مقدمه :

این فصل در حقیقت قسمت دوم فصل پیشین می باشد. در فصل جاری مروری خواهیم داشت بر باقیمانده ی کنترل های مهم سرور وب .

ارزیابی داده های ورودی کاربر:

یکی از مهمترین مراحل دریافت داده ها از کاربر این است که اطمینان حاصل کنیم آیا داده های وارد شده از طرف او معتبر هستند یا خیر؟ اصول تعیین اعتبار بدین شرح می باشند: آیا کاربر چیزی را وارد کرده است ؟ آیا نوع صحیحی از داده را وارد کرده است (برای مثال آدرس ایمیل) . آیا داده ورودی در یک بازه خاص قرار دارد؟ و امثال اینها.

ASP.NET یک سری از کنترل های ارزیابی داده های ورودی را قبل از اینکه داده ها به سرور فرستاده شوند ، در سمت کلاینت مهیا کرده است و به این صورت بدون درگیر شدن سرور و تحمیل بار اضافی به آن اینکار صورت می گیرد.

تعیین اعتبار داده های ورودی در سمت کلاینت توسط کتابخانه ای که در فایل WebUIValidation.JS قرار دارد و به صورت مجزا بر روی کامپیوتر های کلاینت دریافت خواهد شد ، صورت می گیرد. نکته جالب اینجا است که حتی اگر بدلیل پشتیبانی نکردن مرورگر وب کاربر از Jscript (نگارش های پایین تر از ٤ اینترنت اکسپلورر) تعیین اعتبار سمت کلاینت مهیا نبود، به صورت خودکار تعیین اعتبار سمت سرور مهیا می گردد و در هر حال تعیین اعتبار داده های ورودی صورت خواهد گرفت.

در جدول زیرشش کنترل موجود برای تعیین اعتبارداده های ورودی کاربر، توضیح داده شده اند. این نوع کنترل ها مقدار کنترلی را که در خاصیت ControlToValidate آنها مشخص شده است را بررسی می نمایند.

جدول ۱- کنترل های تعیین اعتبار در ASP.NET .

کنترل	کاربرد
RequiredFieldValidator	بررسی می کند که آیا کنترل حاوی داده است یا خیر. (آیا کاربر چیزی را وارد کرده است ؟)
CompareValidator	بررسی می کند که آیا داده ی وارد شده با داده ی موجود در کنترل دیگر تطابق دارد یا خیر.
RangeValidator	بررسی می کند که آیا آیتم وارد شده بین دو مقدار تعریف شده قرار دارد یا خیر.
RegularExpressionValidator	بررسی می کند که آیا داده وارد شده با فرمت مشخص شده مطابقت دارد یا خیر.
CustomValidator	اعتبار داده ی ورودی را توسط اسکریپتی کلاینت ساید یا سمت سرور و یا هر دو انجام می دهد.
validationSummary	تمام موارد بررسی شده را در یک مکان نمایش می دهد یا به صورت کلی فقط یک پیغام را نمایش می دهد.

برای استفاده از کنترل های تعیین اعتبار باید مراحل زیر طی شود :

۱- ترسیم و یا قرار دادن یک کنترل اعتبار ورودی روی فرم و تنظیم کردن خاصیت ControlToValidate آن به کنترلی که می خواهید تعیین اعتبار شود. اگر شما از کنترل CompareValidator استفاده می کنید ، باید خاصیت ControlToCompare را نیز تنظیم کنید.

۲- خاصیت ErrorMessage را به پیغامی که می خواهید هنگامیکه داده ی ورودی معتبر نیست نمایش دهد تنظیم کنید.

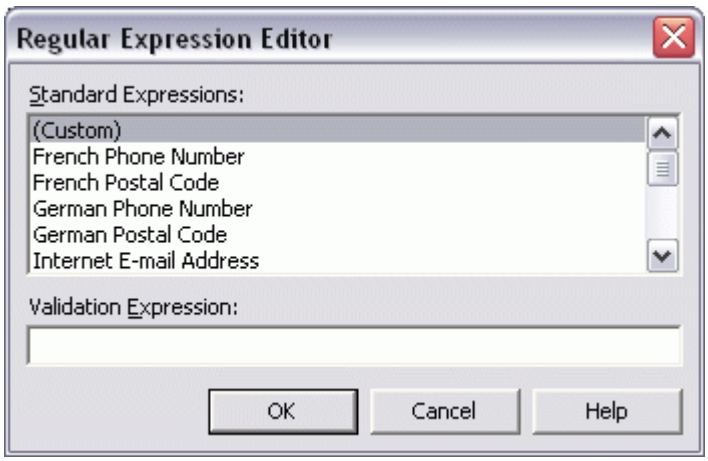
۳- خاصیت Text آنرا برای نمایش دادن پیغامی هنگامیکه خطا رخ می دهد ، تنظیم کنید. از این مورد برای نمایش دادن توضیحات طولانی تر از خاصیت ErrorMessage استفاده می شود.

٤- در صورت نیاز یک کنترل ValidationSummary را روی فرم وب برای نمایش تمام پیغام های خطای حاصل از کنترل های تعیین اعتبار ، ترسیم کنید.

٥- تنها وجود کنترلی که سبب فرستاده شدن یک رخداد Post-Back می شود ، سبب انجام بررسی تعیین اعتبار می گردد. پس وجود یک چنین کنترلی (مانند یک دکمه) روی فرم در این حالت ضروری است.

برای نمایش خطاهای تعیین اعتبار به صورت یک MessageBox خاصیت کنترل ValidationSummary به نام ShowMessage را True کنید.

اگر یک کنترل RegularExprssionValidator را روی صفحه قرار دهید و خاصیت ValidationExpression آنرا انتخاب نمایید ، دیالوگ باکس شکل -۱ ظاهر می شود که در اغلب موارد کافی می باشد.



شکل ۱- صفحه ی ادیتور مربوط به کنترل RegularExprssionValidator .

این کنترل از زبان Pattern-matching برای تعیین اعتبار داده ی ورودی استفاده می کند (طلاعات وسیعتر را در این زمینه می توان در MSDN بدست آورد.)

ترکیب کنترل های تعیین اعتبار :

یک کنترل روی صفحه می تواند از چندین کنترل تعیین اعتبار استفاده کند. برای مثال TextBox ایی که ایمیل کاربر را دریافت می کند می تواند به کنترل RequiredFieldValidator و کنترل RegularExpressionValidator متصل باشد.

در مورد کنترل CompareValidator باید به نکات زیر توجه داشت :

- اگر کنترل مشخص شده در خاصیت ControlToValidate نتواند به یک نوع داده مناسب تبدیل شود، **نتیجه** Invalid خواهد بود.
- اگر کنترل مشخص شده در خاصیت ControlToComare نتواند به یک نوع داده مناسب تبدیل شود، نتیجه Valid خواهد بود. در این حالت باید از یک کنترل دیگر برای تعیین اعتبار بهره جست.

هنگامیکه می خواهیم از کنترل CompareValidator استفاده کنیم با تنظیم کردن خاصیت Operator آن می توان نوع مقایسه را انجام داد. برای مثال گاهی از اوقات ورودی یک فیلد باید کمتر یا مساوی فیلد دیگری باشد و امثال اینها. این خاصیت در پنجره خواص کنترل ذکر شده به سادگی قابل تنظیم است.

مثال اول :

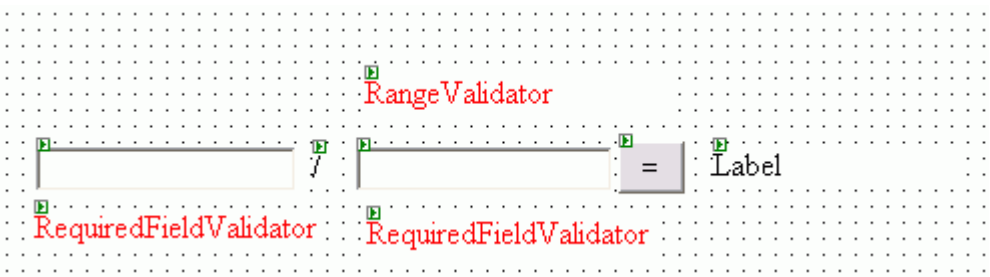
مطابق شکل۲- دو TextBox ، دو Label و یک عدد Button روی صفحه قرار دهید. می خواهیم مقدار عددی TextBox اول را به مقدار عددی TextBox دوم تقسیم کنیم و حاصل را در Label نهایی نمایش دهیم.

شکل ۲ – نمای ابتدایی مثال اول.

نام TextBox ها را (از سمت چپ) به txtVal1 و txtVal2 و دکمه را به btnCalc و Label نهایی را به lblResult تغییر دهید.

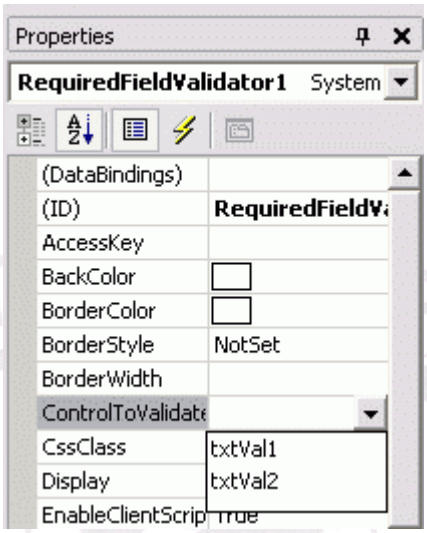
می خواهیم چک کنیم که آیا کاربر در هر دو TextBox چیزی وارد کرده است یا خیر و آیا محتویات TextBox دوم از صفر بزرگتر و کوچکتر از ۱۰۰۰ می باشد؟

برای این منظور دو کنترل RequiredFieldValidator و یک کنترل RangeValidator را روی فرم قرار دهید (شکل ۳).



شکل ۳- قرار دادن کنترل های تعیین اعتبار روی فرم وب.

خاصیت ControlToValidate کنترل RequiredFieldValidator مربوط به تکست باکس اول را به txtVal1 تنظیم کنید (شکل ٤) و این خاصیت را برای کنترل RequiredFieldValidator مربوط به TextBox دوم به txtVal2 تنظیم نمایید.

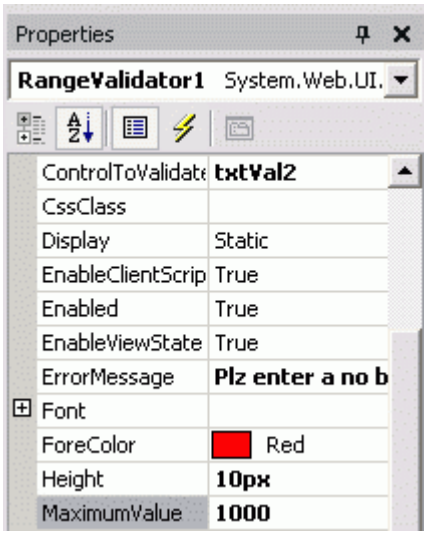


شکل ٤- تنظیم کردن خاصیت ControlToValidate کنترل RequiredFieldValidator .

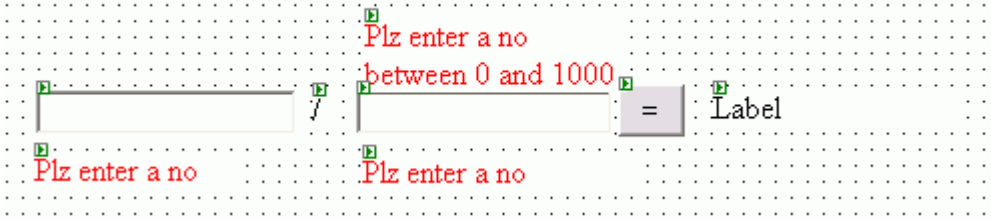
و در مورد کنترل RangeValidator ابتدا خاصیت ControlToValidate را به txtVal2 تنظیم کنید و سپس min و ***max*** آنرا به ۱ و ۱۰۰۰ تغییر دهید (شکل ٥).

اکنون نوبت به تنظیم کردن خاصیت ErrorMessage تک تک کنترل های تعیین اعتبار می باشد. برای هرکدام یک عبارت معنا دار بنویسید (شکل ٦).

روی دکمه دوبار کلیک کنید و کدی ساده برای تقسیم کردن دو عدد بر هم را بنویسید (لطفا به سورس همراه مراجعه کنید) .



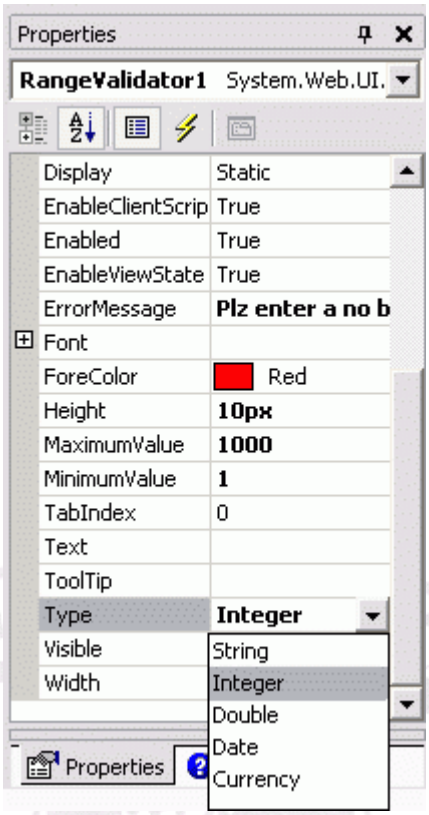
شکل ٥- تنظیم خواص کنترل RangeValidator .



شکل ٦- نمای فرم پس از تنظیم خاصیت ErrorMessage کنترل های تعیین اعتبار داده ها.

حالا برنامه را اجرا کنید . قبل از هرکاری روی دکمه انجام محاسبه کلیک نمایید تا نتیجه مطلوب را ببینید!

اگر در TextBox دوم عددی بزرگتر از یک را وارد نماییم یک پیغام خطا از طرف کنترل RangeValidator ظاهر می شود. چرا؟! چون نوع داده ی ورودی را مشخص نکرده ایم! خاصیت Type این کنترل را به int تغییر دهید (شکل ٧) .

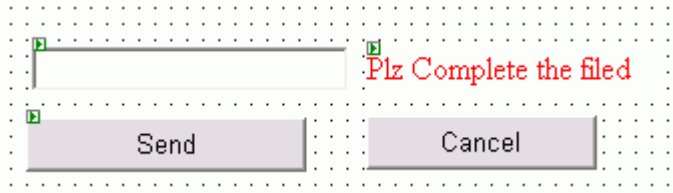


شکل ۷- قبل از هر کاری باید نوع داده ی ورودی کنترل RangeValidator را مشخص کرد.

کنسل کردن تعیین اعتبار :

چون تعیین اعتبار قبل از اینکه سرور صفحه ای را پردازش کند اتفاق می افتد ، ممکن است کاربر در بین انبوهی از پیغام های خطا گیر بیفتد و نه راه پس داشته باشد و نه پیش! برای کنسل کردن این کنترل ها می توان از یک کنترل که خاصیت Post-Back نداشته باشد مانند Submit HTML Control استفاده کرد.
در این حالت می توان کاربر را به یک صفحه ی دیگر هدایت کرد (با بررسی کردن خاصیت IsValid شیء Page)

مثال دوم:



شکل ۸- تصویری از مثال دوم در حالت طراحی.

در این مثال می خواهیم اگر کاربر بر روی دکمه ی کنسل کلیک کرد بدون انجام Validation به صفحه ی دیگری راهنمایی شود.

یک TextBox به نام txtID ، یک RequiredFieldValidator که این TextBox را کنترل می کند روی فرم قرار دهید. یک دکمه سرور وب به نام btnSend و یک دکمه HTML معمولی از نوع Submit به نام btnCancel را روی فرم قرار دهید (شکل ۸).

یک فرم وب جدید به برنامه از طریق منوی Project (آیتم Add web form) اضافه نمایید و نام پیش فرض آنرا بپذیرید. روی این فرم یک Label قرار دهید و داخل آن بنویسید : **Sorry!** .

در سورس HTML صفحه ، خواص کنترل Submit را به صورت زیر باید تغییر داد:

```
<INPUT style="Z-INDEX: 101; LEFT: 274px; WIDTH: 115px; POSITION: absolute; TOP: 142px; HEIGHT: 27px"
type="submit" value="Cancel" id="btnCancel" language="javascript" onclick="Page_ValidationActive=false;">
```

و در فرم وب کد زیر را می توان اضافه کرد :

```
private void Page_Load(object sender, System.EventArgs e)
{
        if ( Page.IsPostBack )
        {
                Page.Validate();
                //user cancelled the validation
                if (! Page.IsValid )
                        Response.Redirect("WebForm2.aspx") ;
        }
}
```

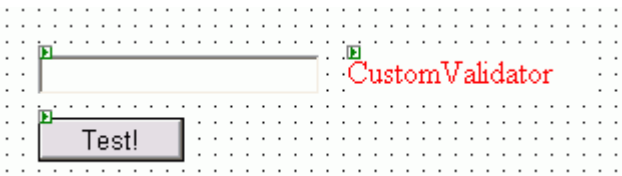
در این مثال اگر کاربر روی دکمه کنسل کلیک کند و در صورتیکه چیزی را وارد نکرده باشد به راحتی به صفحه ی Sorry! راهنمایی می شود! (این برنامه را بیشتر می توان توسعه داد...)

تعیین اعتبار سفارشی :

اگر هیچکدام از کنترل های تعیین اعتبار نیاز شما را برآورده نمی کنند می توانید از کنترل CustomValidator استفاده نمایید. اگر لازم است پردازش سمت سرور انجام شود ، کد تعیین اعتبار را در رخداد ServerValidate قرار دهید. برای تعیین اعتبار سمت کلاینت خاصیت ClientValidationFunction این کنترل را باید تنظیم کرد.

مثال سوم :

در این مثال قصد داریم بررسی کنیم آیا ورودی کاربر یک عدد اول است یا خیر.
برای این منظور یک دکمه به نام btnTest و یک TextBox به نام txtPrime و یک کنترل CustomValidator به نام vldtxtPrime را روی فرم قرار دهید (شکل ۹) .

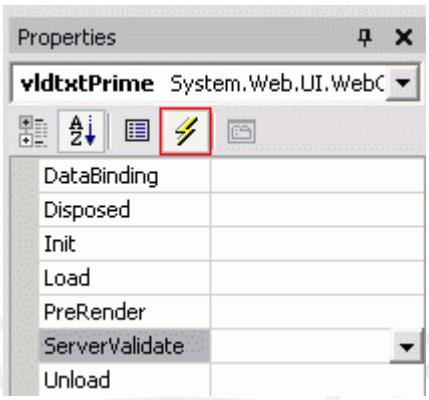


شکل ۹- فرم وب مثال سوم در حالت طراحی

خاصیت ControlToValidate مربوط به vldtxtPrime را به btnTest تنظیم کنید. سپس در برگه ی خواص ، کنترل vldtxtPrime را انتخاب نموده و روی آیکونی به شکل رعد و برق کلیک نمایید . صفحه

ی رخدادهای این کنترل ظاهر می شود (شکل ۱۰) . اکنون در قسمت ServerValidate دوبار کلیک کنید. رخداد زیر به صورت اتوماتیک به برنامه اضافه می شود:

```
private void vldtxtPrime_ServerValidate(object source,
                                        System.Web.UI.WebControls.ServerValidateEventArgs args)
```



شکل ۱۰– نحوه ی اضافه کردن یک رخداد به کنترل CustomValidator .

در این رخداد عملیات چک کردن ورودی کاربر صورت می گیرد. برای تکمیل کد لطفا به کد همراه مراجعه نمایید.

(بهتر است خاصیت ErrorMessage این کنترل را به یک عبارت معنا دار تبدیل کنید.)

موارد تکمیلی کنترل های وب :

طریقه ی حرکت بین صفحات مختلف در ASP.NET :

برای حرکت بین صفحات مختلف ، ASP.NET روش های مختلفی را ارائه داده است که در ادامه بررسی خواهند شد.

جدول ۲- حرکت بین صفحات در ASP.NET

روش هدایت و حرکت به صفحه ای دیگر	کاربرد
کنترل HyperLink	حرکت به صفحه ای دیگر
تابع ;(...)Response.Redirect	معادل کلیک بر روی یک کنترل HyperLink می باشد.
تابع ;(...)Server.Transfer	به فرم وب جاری خاتمه بخشیده و اجرای صفحه ای دیگر را آغاز می کند. این روش تنها برای حرکت بین فرم های وب (.aspx) کاربرد دارد.
تابع ;(...)Server.Execute	درحالیکه فرم وب جاری درحال نمایش است ، اجرای یک فرم وب جدید را آغاز می کند. محتویات هر دو فرم ترکیب خواهند شد. این روش نیز تنها برای فرم های وب کاربرد دارد.
تابع اسکریپتی window.open(...);	یک صفحه را در یک پنجره جدید مرورگر نمایش می دهد. اگر کاربر از برنامه هایی مانند pop-up stopper استفاده کند این متد کارآیی نخواهد داشت.

استفاده از HyperLink و Redirection :

با تنظیم کردن خاصیت NavigateURL کنترل HyperLink ، با کلیک کاربر بر روی این کنترل به صفحه ی مشخص شده ، هدایت می گردد. این کنترل سبب انجام هیچگونه رخدادی در سمت سرور نمی گردد. در صورت نیاز به پردازش رخداد کلیک می توان از کنترل های LinkButton و یا ImageButton استفاده کرد. در این حالت می توان از تابع Response.Redirect برای هدایت کاربر به صفحه ای دیگر استفاده کرد.

استفاده از متد Transfer :

استفاده از این تابع یا متد بسیار شبیه به استفاده از HyperLink و یا استفاده از تابع Redirect می باشد با یک تفاوت : Transfer می تواند بعضی از اطلاعات مربوط به صفحه ی اصلی را دربین درخواست ها ، حفظ و نگهداری کند. تنظیم کردن آرگومان تابع Transfer به True سبب می شود که QueryString و ViewState و پروسیجر رخداد در فرم مقصد نیز مهیا باشند. برای استفاده از این حالت ابتدا باید خاصیت EnableViewStateMac فرم وب را false کنید. به صورت پیش فرض ASP.NET اطلاعات ViewState را Hash می کند. با False کردن آن ، این اطلاعات در صفحه ی دیگر نیز قابل خواندن خواهند شد.

برای دریافت این اطلاعات در صفحه ای دیگر می توان از Request.Form استفاده کرد. اگر با ASP قدیمی آشنایی داشته باشید این نوع روش ها فقط برای حفظ سازگاری با آن در ASP.NET قرار داده شده است (پس زیاد نگران نباشید!).

تذکر : متدهای Transfer و Execute تنها با فرم های وب کار می کنند. هرگونه سعی در استفاده از یک صفحه ی HTML معمولی با یک خطای زمان اجرا پاسخ داده خواهد شد.

مثال ۴ :

یک پروژه جدید باز کنید . از منوی پروژه یک فرم وب دیگر به برنامه اضافه نمایید. در این برنامه می خواهیم اطلاعات موجود در فرم اول را در فرم دوم نمایش دهیم.

روی فرم اول یک TextBox و یک دکمه قرار دهید و نام آنها را به btnSend و txtSend تغییر دهید. روی دکمه دوبار کلیک کنید و کد زیر را در رخداد کلیک آن بنویسید.

```
Server.Transfer("WebForm2.aspx",true) ;
```

روی فرم دوم یک Label به نام lblReceive قرار دهید و بر روی صفحه دوبار کلیک نموده و در رخداد Page_Load آن بنویسید :

```
lblReceive.Text = "Received from WebForm1: "+ Request.Form["txtSend"].ToString()  ;
```

(فراموش نکنید که EnableViewStateMac فرم اول را باید False نمایید. به سورس HTML صفحه رجوع کنید. اگر به صورت خودکار این مورد به تگ بالای صفحه اضافه نشده ، یک بار این خاصیت را در پنجره خواص خواص True کنید و سپس آنرا False نمایید تا به صورت خودکار به سورس HTML صفحه اضافه گردد. در غیراینصورت برنامه اجرا نخواهد شد.)

```
<%@ Page language="c#" Codebehind="WebForm1.aspx.cs" AutoEventWireup="false"
 Inherits="ex04.WebForm1" enableViewState="True" enableViewStateMac="False"%>
```

استفاده از متد Execute :

با استفاده از متد Execute می توان فرم وب دوم را بدون ترک اولین فرم وب ، پردازش کرد. این مورد اجازه می دهد نتایج را از یک فرم وب به ناحیه ای در همین صفحه جاری هدایت کنیم. همانند متد Transfer ، باید EnableViewStateMac صفحه False شود.

هنگامیکه فرم های وب را با استفاده از متد Execute ترکیب می کنید، هر گونه رخداد Post-Back استفاده شده در فرم دوم سبب پاک شدن فرم اول می گردد. برای این منظور استفاده از این روش تنها هنگامی مفید است که فرم وب دوم حاوی کنترلی نباشد که رخداد Post-Back ایی را سبب شود.

نمایش صفحه در یک صفحه مرورگر جدید :

با استفاده از متد window.open کلاینت ساید می توان یک صفحه ی جدید در مرورگر بازکرد. برای مثال :

```
Window.Open( "http://www.wrox.com/", "myWindowOne",
             "toolbar=no, menubar=no, location=no, directories=no" );
```

البته اگر خاصیت Target را در کنترل HyperLink عوض کنید می توان یک چنین کاری را با کنترل های وب هم انجام داد.

تمرین :

۱- از آنجائیکه تابع window.open آرگومانهای زیادی را می پذیرد می توان آنرا به صورت یک کلاس encapsulate کرد. به عنوان تمرین اینکار را انجام دهید.
۲- یک کنترل ValidationSummary را روی فرم قرار داده و مثال اول فصل را توسعه دهید.
۳- با استفاده از کنترل CompareValidator فیلدهای مثال اول فصل را طوری تنظیم کنید که مقدار فیلد دوم همواره کمتر از مقدار فیلد اول باشد.
٤- حالت کنترل اعتبار سمت کلاینت مثال سوم فصل را توسعه دهید.
٥- مثال چهارم فصل را با Server.Execute دوباره نویسی نمایید و نتیجه را مقایسه کنید.
٦- برای اینکه URL متد window.open به صورت متغیر باشد و بتوان با برنامه نویسی آنرا تنظیم کرد چه کاری را باید انجام داد؟