ASP.NET Validation Controls İşlemleri

Kullanıcı bir **web form'u** doldururken veya doldurduktan sonra girdiği bilgilerin doğru olup olmadığını kontrol etmeliyiz. Aksi taktirde girilen değerleri hemen database vs gibi ortamlara aktarırsak hatalı mesajlar alabiliriz. Mesela email adresi istenen bir **TextBox'ın** içerisine geçerli olmayan adresin girilip girilmediği ile ilgili kontrol'ü bir validation control sayesinde yapabiliriz. Kullanıcıların yaptığı hatalardan birkaçını ifade edersek niçin validation control'lere ihtiyacımız olduğunu daha iyi anlarız :

1. Önemli bir alanın boş bırakılması
2. Geçerli olmayan email adresin girilmesi
3. Sayısal değerler girilmesi gereken alana sayısal olmayan değerlerin girilmesi

NOT : Hacker'lar sayfalarımızı crash ettirebilmek için özenle seçilmiş değerler girebilirler. Bu tarz bir durum meydana geldiğinde bir de database'e girilen verilerin kaydedilmesi varsa, mesela yeni üyelik formu gibi, telafisi olmayan hatalar ortaya çıkabilir. Bu yüzden form'larda çok iyi bir şekilde validation/kontrol etme işlemi yapmalıyız...

The Validator Controls

ASP.NET bize beş tane validator control sağlar. Bu control'ler sayesinde her tür kontrolü yapabiliriz. Bu control'leri **TollBox'ın Validation** grubu altında bulabiliriz:

RequiredFieldValidator : Input control **(TextBox)'ün,** boş olup olmadığını kontrol etmeyi sağlar.

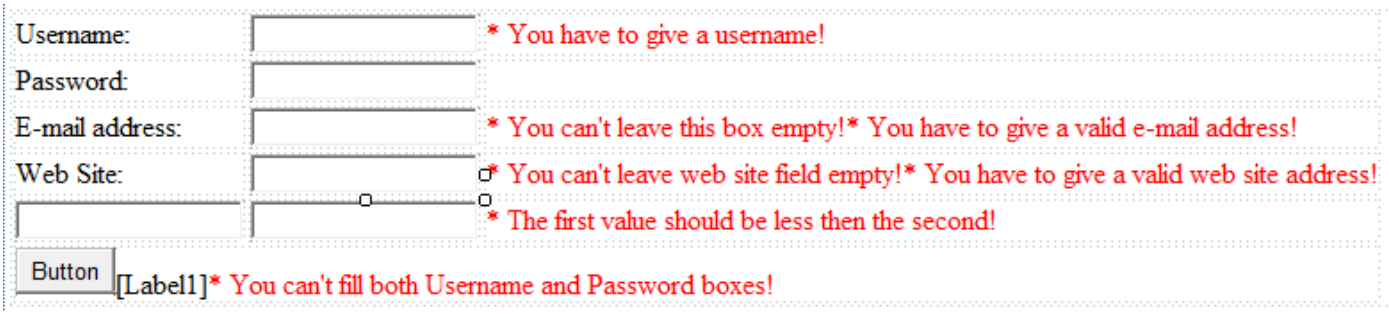
RangeValidator : Input control'ün özel nümerik, alfabetik veya özel bir aralık değerinde olup olmadığını kontrol etmeyi sağlar.

CompareValidator : Input control'ün diğer bir input control'le aynı olup olmadığını veya bizim belirlediğimiz değerin girilip girilmediğini kontrol etmeyi sağlar. Mesela email adresinin yeniden girilmesi istendiğinde bu control görev yapar.

RegularExpressionValidator : Input control'ün özel bir ifadeye sahip olup olmadığını kontrol etmemizi sağlar. Mesela email adresinin girildiği **TextBox'a stringIfadesi@stringIfadesi.com** şeklinde girilip girilmediğini kontrol edebiliriz.

CustomValidator : **CustomValidator'u** sayfamıza eklediğimizde üzerine çift tıklayarak bu control'ün bize sağladığı **event** handler içerisinde kendi özel kontrol'lerimizi yapabiliriz.

Kullanımı:




```
<asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
ControlToValidate="TextBox1" Display="Dynamic" ErrorMessage="* You have to give a username!">
</asp:RequiredFieldValidator>

<asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server"
ControlToValidate="TextBox3" Display="Dynamic" ErrorMessage="* You can't leave this box empty!">
</asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server"
ControlToValidate="TextBox3" Display="Dynamic" ErrorMessage="* You have to give a valid e-mail address!"
ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*">
</asp:RegularExpressionValidator>

<asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server"
ControlToValidate="TextBox4" Display="Dynamic" ErrorMessage="* You can't leave web site field empty!">
</asp:RequiredFieldValidator>

<asp:RegularExpressionValidator ID="RegularExpressionValidator2" runat="server"
ControlToValidate="TextBox4" Display="Dynamic" ErrorMessage="* You have to give a valid web site address!"
ValidationExpression="http(s)?://([\w-]+\.)+[\w-]+(/[\w- ./?%&=]*)?">
</asp:RegularExpressionValidator>

<asp:CompareValidator ID="CompareValidator1" runat="server" ControlToCompare="TextBox5"
ControlToValidate="TextBox6" ErrorMessage="* The first value should be less then the second!"
Operator="GreaterThan" Type="Date">
</asp:CompareValidator>
```

`Display="Dynamic"` ifadesiyle bir control'ü birden fazla validation control'e tabi tuttuğumuzda Validation control'leri yan yana koyarız. Sağ tarafa konulan validation control'ün hata mesajı gösterileceği zaman boşlukların oluşmaması için kullanılır.

`ControlToValidate="TextBox1"` ifadesinden de anlaşıldığı gibi bu attribute bir control'ü değer olarak alır ve onu kontrol eder. Eğer **TextBox1** boş bırakılırsa **ErrorMessage**

ile belirtilen mesaj gösterilecektir.

CompareValidator control'ünün **ControlToCompare'e** attribute'sine **TextBox5** control'ü atanmış, **ControlToValidate** attribute'sine ise **TextBox6** control'ü atanmış. Bundan dolayı **TextBox5** ile **TextBox6,** Operator ve Type attribute'lerindeki değere göre compare edilecek ve Operator attribute'sindeki değer'e uygun bir sonuç çıkmazsa hata mesajı gösterilecektir.

Operator attribute'sinin aldığı değerler	Type attribute'sinin aldığı değerler
DataTypeCheck Equal GreaterThan GreaterThanEqual LessThan LessThanEqual NotEqual	Currency Date Double Integer String

Not: RegularExpressionValidator control'ündeki `ValidationExpression="http(s)?://([\w-]+\.)+[\w-]+(/[\w- ./?%&=]*)?"` ifadesinin anlamak için **Regular Expressions(Düzenli İfadeler)** (http://www.softwarevol.com/tutorial/Regular-Expressions-Duzenli-Ifadeler) isimli makaleye bakabilirsiniz.

```
<asp:CustomValidator ID="CustomValidator1" runat="server"
ErrorMessage="* You can't fill both Username and Password boxes!"
onservervalidate="CustomValidator1_ServerValidate">
</asp:CustomValidator>
```

```
protected void CustomValidator1_ServerValidate(object source, ServerValidateEventArgs args)
{
    if (TextBox1.Text != string.Empty && TextBox2.Text != string.Empty)
    {
        args.IsValid = false;
    }
}
```

CustomValidator control'ünün üzerine çift tıkladığımız zaman yukarıdaki **CustomValidator1_ServerValidate** isimli **event** handler metodu görülür. Bu metod'ta dilediğimiz gibi validation işlemi yapabiliriz. **CustomValidator** control'ünün **ClientValidationFunction** attribute'sine script'teki bir fonksiyonu atayabiliriz.

```
protected void Button1_Click(object sender, EventArgs e)
{
    if (Page.IsValid)
    {
        Label1.Text = "You have succesfully filled the form!";
    }
}
```

Button1_Click metodu ile de tüm validation control'lerin doğru olup olmadığını `if cümleciği` içerisinde kullanılan **Page.IsValid** property ile test ederiz. Mesela bu if cümleciği içerisinde input'lara girilen değerleri database'e kaydedebiliriz, çünkü yaptığımız kontrollerden başarılı olan bilgiler doğru bilgilerdir ve database'e bir sorun olmadan aktarılabilir ?

Bence hemen karar vermemeliyiz çünkü belki kullanıcı tüm validation control'lerimizi başarıyla geçmiş olabilir; fakat bir hacker sayfamızı crash ettirmek isterse ne yapacağız? Örneğin kullanıcı ismi kısmına kullanıcı ismini girdikten sonra ' işareti koyarsa ne olacak ? Hatırlarsak ' işareti database'e bilgi kaydederken `insert into('...')` ifadesinde kullanılıyordu.

Bir örnek vermek gerekirse kullanıcı ismini **olyanren'** olarak girdi diyelim. `insert into('olyanren'')` şeklinde database kayıt işlemi yapılmaya çalışılır. Böyle olduğunda ise büyük bir hata meydana gelebilir. Gelen bu hata hacker'a bazı ipuçları sağlar.

Sonuç olarak web form doldurma işlemi sonunda girilen değerlerin kontrol edilmesi güvenlik açısından çok önemlidir, bundan dolayı formu mutlaka bir kontrol aşamasından geçirmek gerekmektedir