Hoofdstuk 2: Servercontrols

Server controls inleiding

Server controls zijn speciale tags die op de web server worden verwerkt. Dit gebeurt ongeveer op dezelfde manier als HTML tags door uw browser worden geïnterpreteerd. Server control tags kunt u herkennen doordat zij het *runat="server"* attribuut bevatten. Op die manier kan de server een onderscheid maken tussen een server control en standaard HTML-tags of andere tekst in de pagina.

De code runat="server" zorgt ervoor dat we de eigenschappen van deze controls (bijvoorbeeld een textbox) kunnen gaan benaderen en wijzigen in onze code.

Op het moment dat de server een request voor een ASP.NET pagina binnenkrijgt, gaat deze op zoek naar server controls, maakt voor elke server control de nodige objecten aan met hun eigen attributen en methoden, triggert bepaalde server-events tijdens de uitvoer van de pagina, om uiteindelijk de gegenereerde HTML-code terug te sturen naar de browser.

Een voorbeeld van een server control is de Calendar control. Wanneer we deze uit onze Toolbox op een aspx-pagina slepen, wordt de control als volgt in onze broncode geplaatst:

```
<%@ Page Language="VB" AutoEventWireup="false"
CodeFile="voorbeeld_006.aspx.vb" Inherits="voorbeeld_006" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <asp:Calendar ID="Calendar1" runat="server"></asp:Calendar>
    </div>
    </form>
</body>
</html>
```

Merk op dat deze control binnen speciale <asp>-tags staat en dat het attribuut runat="server" aanwezig is. Dankzij dit attribuut is de control aanspreekbaar vanuit onze code door gebruik te maken van de toegekende ID "Calendar1". Een ID moet steeds uniek zijn binnen eenzelfde pagina.

```
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load

        Calendar1.SelectedDate = System.DateTime.Today

    End Sub
```

Wanneer we het resultaat van deze applicatie in onze browser bekijken, zullen we merken dat het ASP.NET-framework de calendar control heeft omgezet naar standaard html-code en javascript. Met andere woorden: elke servercontrol zal uiteindelijk omgezet worden naar een standaard HTML-pagina, al dan niet met javascript erin verwerkt.

Er bestaan vier verschillende soorten Server Controls:

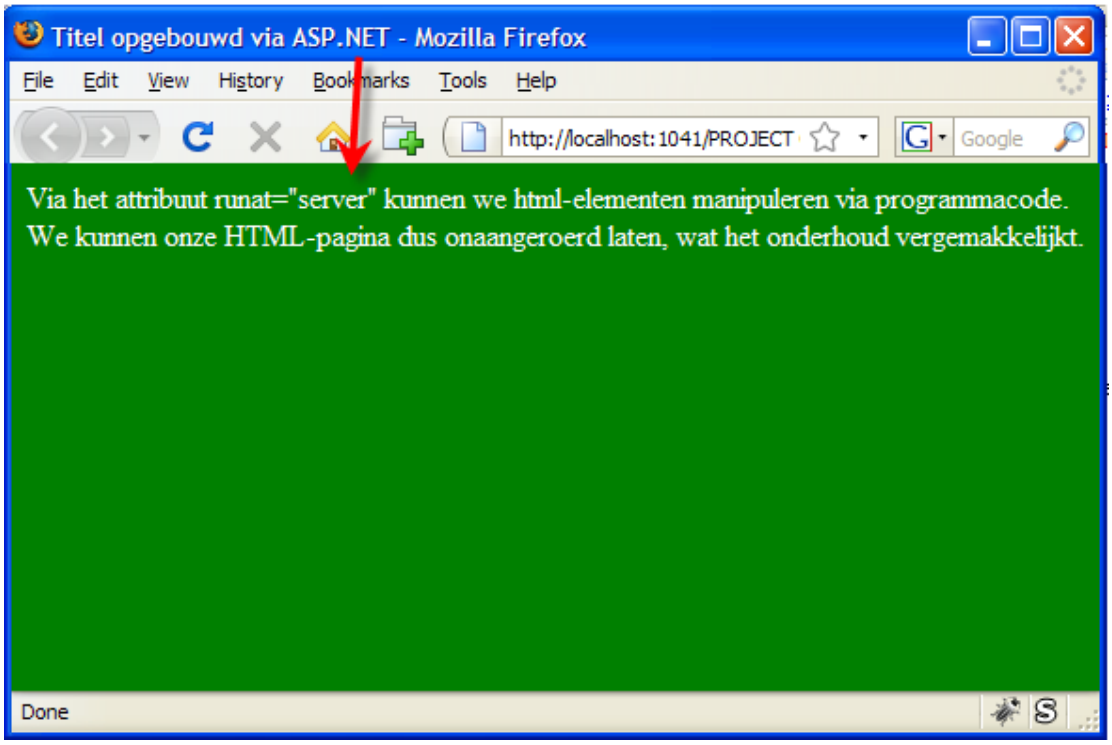
1. - *HTML-controls*: dit zijn HTML tags die een overeenkomstige klasse hebben in de System.Web.UI.HtmlControls namespace. Zo bestaat er een HtmlTable voor de <table> tag en een HtmlImage voor de <img> tag. De HTML-controls werden ontwikkeld om binnen ASP.NET eenvoudig met standaard HTML-elementen te kunnen werken.

Het grote voordeel van HTML-controls in vergelijking met de overeenkomstige standaard HTML-elementen is dat we in onze achterliggende programmacode de elementen kunnen manipuleren zonder in de HTML-code te moeten duiken.

Om in HTML een element aan te duiden als HTML-control, dient u het attribuut **runat="server"** toe te voegen aan het element. Bijvoorbeeld, om het HTML-element <title> en onze <body> binnen ASP.NET te kunnen aanspreken en manipuleren doen we het volgende (let op de ID en runat="server" attributen bij title en body):

```
<%@ Page Language="VB" AutoEventWireup="false"
CodeFile="voorbeeld_006.aspx.vb" Inherits="voorbeeld_006" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title id="paginaTitel" runat="server">Untitled Page</title>
</head>
<body id="paginaBody" runat="server">
    <form id="form1" runat="server">
    <div>

    </div>
    </form>
</body>
</html>
```

```
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load

        paginaTitel.Text = "Titel opgebouwd via ASP.NET"
        paginaBody.Attributes("bgcolor") = "green" 'layout via attributen
        paginaBody.Style.Add("color", "white") 'layout via CSS properties

    End Sub
```



REFERENTIE: VOORBEELD_006.ASPX

Op deze manier kunnen we onze HTML manipuleren en het uitzicht bepalen zonder daarbij de HTML-code te moeten aanpassen. Onze programmacode blijft zo netjes gescheiden van onze HTML-code, waardoor de applicatie makkelijker te onderhouden zal zijn en van meer dynamiek voorzien kan worden.

HTML server controls lijken enorm hard op overeenkomstige HTML tags. Het enige verschil met de HTML tags is het 'runat="server"' attribuut dat werd toegevoegd. Deze controls hebben minder mogelijkheden dan web controls, maar u dient te beseffen dat u uw oude HTML-code eenvoudig kan aanroepen binnen ASP.NET door simpelweg runat="server" toe te voegen aan uw traditionele HTML-elementen.

Een lijst van enkele basis HTML server controls:

HtmlAnchor	<a>
HtmlButton	<button>
HtmlForm	<form>
HtmlGenericControl	Voor alle HTML tags die geen control object hebben
HtmlImage	<img>

HtmlInputButton	<input type= button>, <input type= submit>, <input type= reset>
HtmlInputCheckBox	<input type= checkbox>
HtmlInputFile	<input type= file>
HtmlInputHidden	<input type= hidden>
HtmlInputImage	<input type= image>
HtmlInputRadioButton	<input type= radio>
HtmlInputText	<input type= text>, <input type= password>
HtmlSelect	<input type= select>
HtmlTable	<table>
HtmlTableCell	<td>, <th>
HtmlTableRow	<tr>
HtmlTextArea	<textarea>

Elke control heeft dus zijn overeenkomstig object dat heel dicht bij de HTML ligt die wordt gegenereerd.

Een voorbeeld ter illustratie van gebruik van oude HTML-code in combinatie met het **runat="server"** attribuut:

Oefening

1. Maak een nieuwe pagina **"voorbeeld_006b.aspx"** (webform) aan en voeg een klassieke select box toe aan uw pagina via gewone html. Noem deze "select1" en zorg ervoor dat we ze kunnen gebruiken in onze broncode (runat=???). Voeg drie options toe waardoor een gebruiker de keuze krijgt tussen een drietal kleuren.
2. Voeg een textbox toe genaamd "txtTekst" (<input type=…)
3. Voeg een paragraaf <p> toe met ID "deParagraaf"
4. zorg ervoor dat we al deze elementen in onze code kunnen benaderen en geef de body een unieke ID "deBody" zodat we deze op die manier kunnen aanspreken in de code
5. voeg een submit knop toe
6. schrijf code in uw externe code-file die ervoor zorgt dat de pagina de geselecteerde achtergrond kleur en text (in de paragraaf) bevat die we hebben ingevoerd in ons nieuwe formulier, nadat een user op submit heeft gedrukt.

LET OP: gebruik hier enkel klassieke HTML in combinatie met het runat="server" attribuut waar nodig. Sleep nog geen web controls uit uw toolbox naar uw applicatie.

REFERENTIE: VOORBEELD_006B.ASPX

Laten we nu het voorbeeld nemen waar er met geneste FOR lussen een tabel werd opgebouwd, met in elke cel de som der indexen. We gaan hier gebruik maken van de HTML-server control 'HtmlTable'. Tevens voegen we nog 2 HtmlSelect controls toe om het aantal rijen en kolommen te bepalen. We bouwen alle elementen op via de code-behind van onze pagina en niet in de aspx-pagina zelf.

Tracht volgend code-fragment uit de code-behind te begrijpen, u zal dit nadien zelf moeten nabouwen in een werkend voorbeeld:

```
        Dim tabel As New HtmlTable
        tabel.Attributes("border") = 1

        PlaceHolder1.Controls.Add(tabel)

        For i = 1 To intRows ' loop over rijen
            Dim rij As New HtmlTableRow

            For j = 1 To intCols ' loop over columns
                Dim col As New HtmlTableCell
                col.InnerText = "rij" & i & " cel" & j
                rij.Controls.Add(col)
            Next

            tabel.Rows.Add(rij)
        Next
```

Let op, deze oefening moet u maken aan het einde van dit hoofdstuk.

REFERENTIE: VOORBEELD_007.ASPX

2. - *Web-controls*: dit zijn nieuwe ASP.NET tags die na uitvoer één of meerdere HTML tags zal genereren met eventueel ook bijhorende clientside scripting en dergelijke. In de praktijk worden deze ook het meest gebruikt, in tegenstelling tot de klassiekere HTML-controls. Een voorbeeld is de Calendar-control en de label-control.

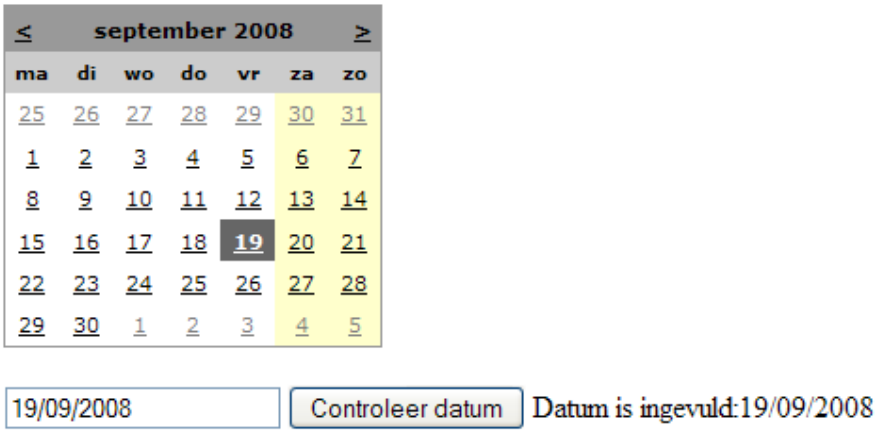
Oefening:

Maak een nieuwe pagina aan en voeg een Calendar control toe. Voeg ook een textbox, button en label toe.

U zou ongeveer het volgende resultaat moeten zien in uw "design view":

Zorg ervoor dat de geselecteerde datum ingevuld wordt in de textbox. Wanneer je op "Controleer datum" klikt dient er gecontroleerd te worden of de datum al dan niet is ingevuld in de textbox. Het is voldoende te controleren of de textbox leeg is of niet.

Indien de datum is ingevuld geeft u volgende boodschap weer: “Datum is ingevuld: 19/09/2008”. Als de datum nog niet is ingevuld (textbox leeg) dan geeft u boodschap “Datum niet ingevuld” weer in het label.



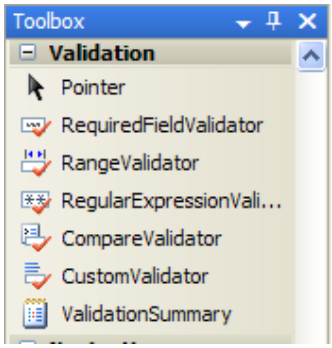
REFERENTIE: VOORBEELD_008.ASPX

3. - *Validatie-controls*: speciale ASP.NET tags voor de validatie van input in formulieren. Via ASP.NET kan u op heel eenvoudige wijze input in formulieren valideren. Hier komen we later uitgebreid op terug.

Een kort voorbeeld maakt duidelijk hoe deze controls eruit zien en werken:

Stel dat we volgend (wel heel erg) klein formulier hebben waar we een gebruiker willen verplichten iets in te typen vooraleer het formulier verzonden kan worden.

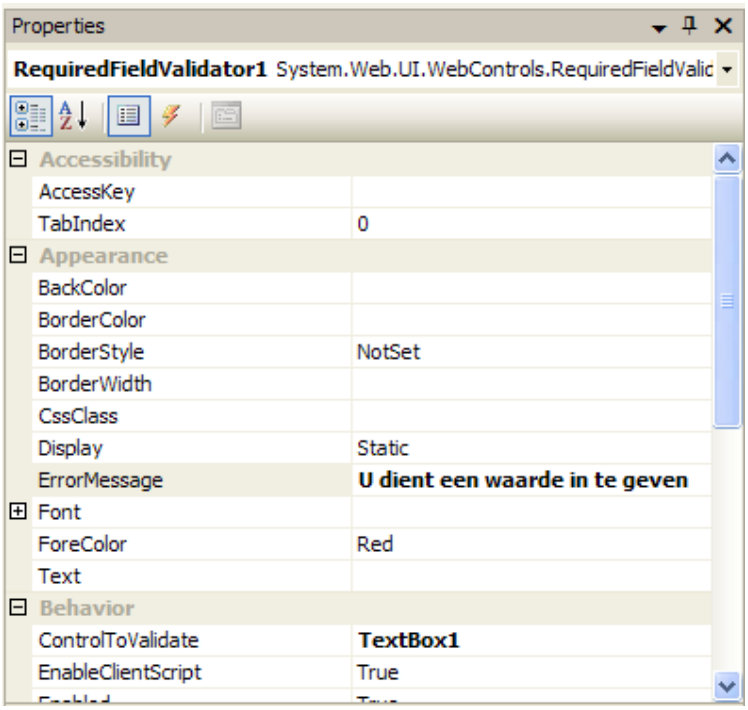
We kunnen via de RequiredFieldValidator de gebruiker verplichten input te tikken in onze textbox. De RequiredFieldValidator en andere validation controls kan u in de Toolbox van Visual Studio terugvinden onder rubriek “Validation”.



We slepen de RequiredFieldValidator naast onze textbox en kunnen via de Properties van deze control allerlei instellingen gaan doen, enkele hiervan zijn:

ErrorMessage: de boodschap die u wenst weer te geven wanneer een gebruiker niets heeft ingegeven in een “required field”.

ControlToValidate: hier zal u de control moeten selecteren waarvoor de RequiredFieldValidator moet werken. We koppelen in dit voorbeeld de control aan onze textbox.



Resultaat: zonder ook maar een regel code te schrijven hebben we al een eerste validatie aangemaakt op ons klein formulier. Voor grotere formulieren kunnen we uiteraard op exact dezelfde manier gaan werken.

In sommige gevallen zijn de standaard validatie-controls te beperkt. Stelt dat een klant een formulier nodig heeft, waarop gebruikers enkel even getallen mogen ingeven. Wanneer een gebruiker een oneven getal ingeeft moet er een foutboodschap getoond worden.

Gelukkig hebben we in dat geval Custom Validators ter beschikking waarin we eender wat voor input kunnen valideren volgens onze eigen regels.

Voorbeeld:

Stel een formulier dat er als volgt uit ziet :

Wanneer een gebruiker op “valideer” klikt, dient onze custom validator te gaan controleren of het getal werkelijk even is of niet. Dit kan door een eigen routine te gaan programmeren die controleert of een getal even is of niet.

Wat moet ons formulier kunnen:

1. valideren of een ingegeven getal tussen 10 en 99 ligt
2. valideren of het getal even is of niet

   Indien niet aan deze voorwaarden wordt voldaan dient de gebruiker een duidelijk foutboodschap te zien op het scherm en mag bijvoorbeeld de achtergrondkleur van onze pagina wijzigen.

Tip: zorg ervoor dat u in uw code “onServerValidate=”true” hebt gezet.

```
    Sub check_even(ByVal source As Object, ByVal args As
ServerValidateEventArgs)
        If (CInt(args.Value) Mod 2) = 0 Then
            args.IsValid = True
        Else
            args.IsValid = False
        End If
    End Sub
```

```
<%@ Page Language="VB" AutoEventWireup="false"
CodeFile="voorbeeld_011.aspx.vb" Inherits="voorbeeld_011" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        <asp:Label ID="Label1" runat="server" Text="Even
getal:"></asp:Label>
        <asp:TextBox ID="txtGetal" runat="server"></asp:TextBox>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator1"
runat="server"
            ControlToValidate="txtGetal" ErrorMessage="Getal moet tussen 10
en 99 liggen"
            ValidationExpression="\d{2}"></asp:RegularExpressionValidator>
```

```
        <br />
        <asp:Button ID="btnValideer" runat="server" Text="Valideer" />
        <asp:CustomValidator ID="CustomValidator1" runat="server"
OnServerValidate="check_even"
            ControlToValidate="txtGetal" EnableClientScript="False"
            ErrorMessage="Getal moet even zijn"
ValidateEmptyText="True"></asp:CustomValidator>

    </div>
    </form>
</body></html>
```

REFERENTIE: VOORBEELD_011.ASPX

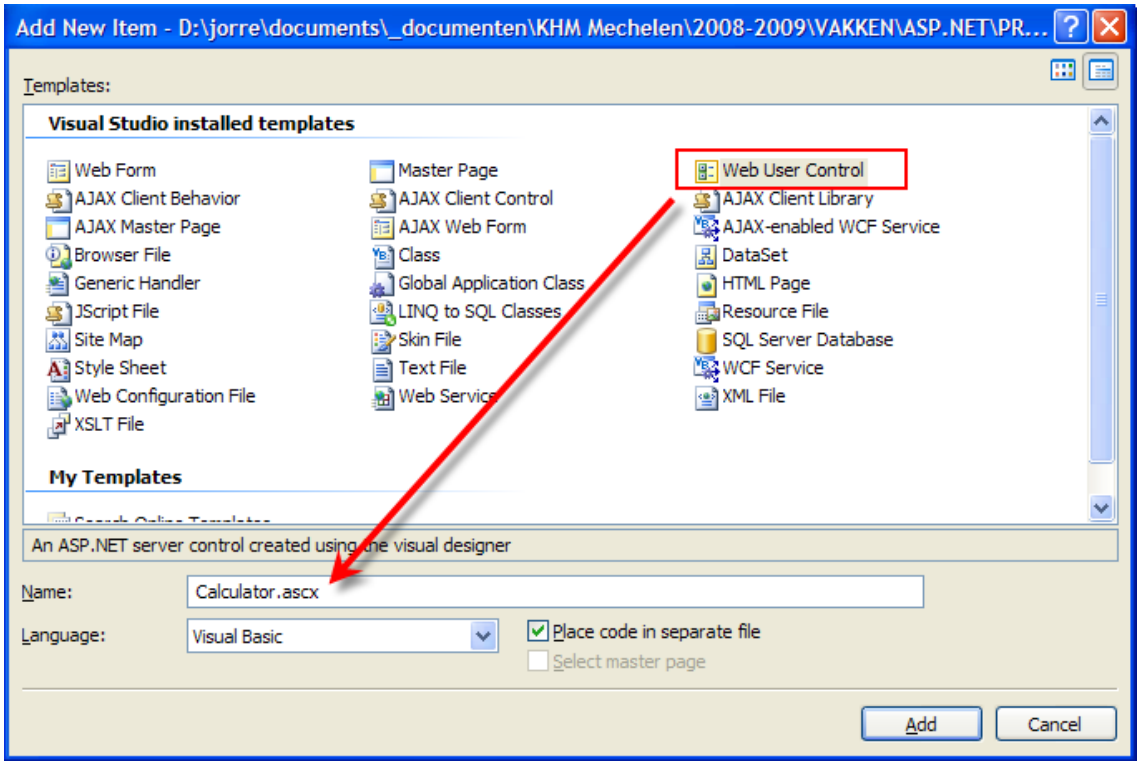
4. - *Gebruiker-controls*: ASP.NET tags waarvan de attributen en het gedrag door de gebruiker zelf werd geprogrammeerd. Zo kunt u bijvoorbeeld zelf een control gaan schrijven om functionaliteit die u vaak opnieuw nodig heeft herbruikbaar te gaan maken. Als u bijvoorbeeld vaak eenzelfde menustructuur gebruikt, kan u hiervoor een user control gaan schrijven.

Microsoft heeft verschillende HTML controls, Web controls en Validatie controls gemaakt die door attributen bepaalde waarden te geven specifieke HTML genereren. Gebruiker-controls zijn zelfgemaakte ASP tags die net zoals bovenstaande controls HTML genereren. Dergelijke zelfgemaakte controls beginnen natuurlijk pas te renderen indien u op meerdere plaatsen of in verschillende pagina's een gelijkaardige combinatie van HTML nodig hebt.

Dergelijke controls worden in een apart bestand met extensie .ascx opgeslagen. Binnen de ASP.NET pagina moet er wel gespecificeerd worden uit welk bestand de gebruikers-control geladen moet worden en welke tag men met deze control wil verbinden.

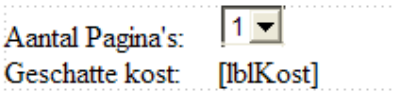
Wanneer we werken met gebruikers-controls (of User Controls) wordt het mogelijk eigen properties, methoden en events hieraan toe te voegen.

Binnen Visual Studio kan je een nieuwe User Control aanmaken via "add new item -> Web User Control"

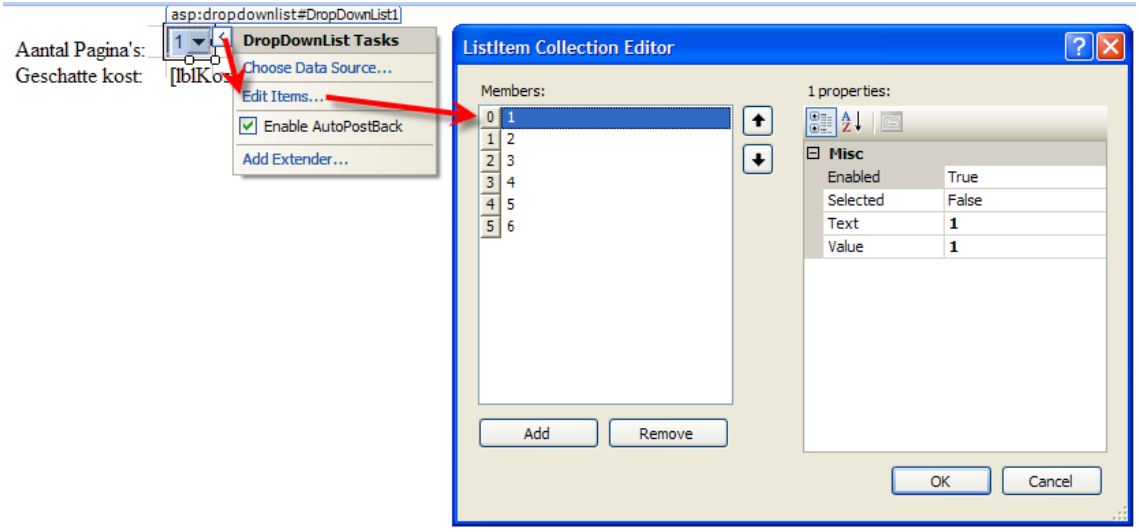


In dit korte voorbeeld zullen we een control toevoegen die automatisch een simpele berekening maakt voor een webdesign offerte, gebaseerd op het aantal pagina's dat een klant zou willen laten ontwerpen.

Stel dat we in verschillende pagina's van onze website/applicatie deze calculator willen plaatsen, dan wordt het nuttig deze functionaliteit te programmeren in een user control. We voegen om te beginnen enkele labels toe en een DropDownList met als resultaat het volgende:



De waardes in de DropDownList kunnen we eenvoudig aanpassen in Visual Studio als volgt:



Door te dubbelklikken op de DropDownList komen we automatisch in de code-behind terecht in het event “DropDownList1_SelectedIndexChanged”. Hier kunnen we functionaliteit gaan schrijven waarmee we kunnen berekenen hoeveel een gebruiker ongeveer zal moeten betalen voor het aantal geselecteerde pagina’s in de calculator.

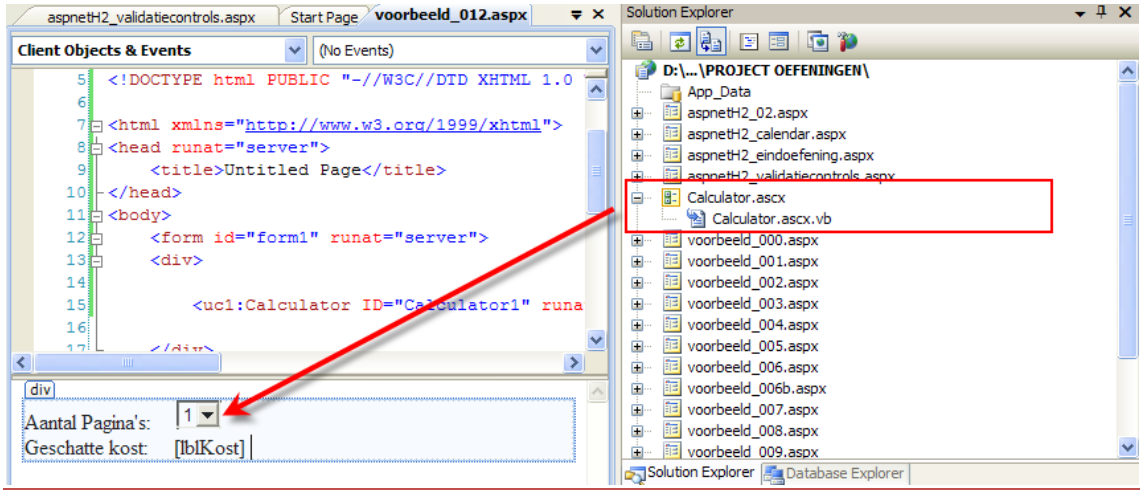
Een eenvoudige berekening kan er als volgt uitzien:

```
Protected Sub DropDownList1_SelectedIndexChanged(ByVal sender As Object,
ByVal e As System.EventArgs) Handles DropDownList1.SelectedIndexChanged

Dim hourlyrate As Double = 55.5
lblKost.Text = CInt(DropDownList1.SelectedValue) * hourlyrate & " Euro"

End Sub
```

REFERENTIE: CALCULATOR.ASCX.VB

Om deze nieuw aangemaakte user control te gaan gebruiken in onze pagina’s, voegen we eerst een nieuw web form toe aan ons project (add new item -> web form).

U kunt gebruiker-controls op twee manieren aan uw pagina toevoegen: declaratief of programmatorisch. Declaratief toevoegen is het eenvoudigste omdat we eenvoudig de control vanuit onze solution explorer kunnen slepen naar ons web form.



In onze broncode kunnen we nu zien dat volgende code automatisch werd toegevoegd:

```
<%@ Register src="Calculator.ascx" tagname="Calculator" tagprefix="uc1" %>
```

Onze control zelf wordt als volgt op onze pagina geplaatst:

```
<uc1:Calculator ID="Calculator1" runat="server" />
```

REFERENTIE: VOORBEELD_012.ASPX

Om via programmacode een user control toe te voegen dient u volgende stappen uit te voeren:

1. voeg een referentie toe naar uw control in uw aspx-page onder de @ Page directive (meestal staat dit op de bovenste lijn van uw aspx-pagina):
   `<%@ Reference Control="~/Calculator.ascx" %>`
2. Nu is de control gekend in uw pagina en kan je hier een nieuwe instantie van aanmaken als volgt:

```
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As
System.EventArgs) Handles Me.Load

        Dim deCalculator As Calculator = LoadControl("Calculator.ascx")
        PlaceHolder1.Controls.Add(deCalculator)

    End Sub
```

3. PlaceHolder1 moet natuurlijk reeds in uw aspx-pagina aanwezig zijn. Hierdoor kan de control via programmacode op de plek van uw PlaceHolder geplaats worden.

REFERENTIE: VOORBEELD_012.ASPX

De mogelijkheden met gebruikers controls zijn oneindig. Bij grote projecten zal men ernaar streven om standaard controls te gaan die de herbruikbaarheid en aanpasbaarheid van de site doorzichtiger maakt.

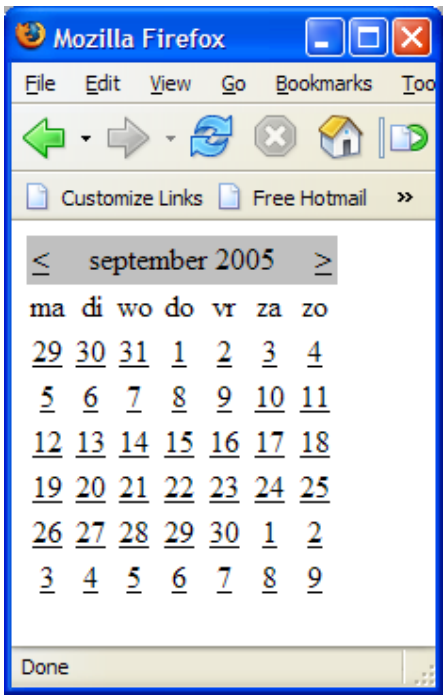
Web-controls in meer detail

Web server controls lijken erg hard op HTML controls. Web controls zijn net zoals HTML controls te herkennen aan het 'runat="server"' attribuut. Daarenboven beginnen de web control tags met 'ASP:'. Het grote verschil tussen HTML en Web controls is dat Web controls **geen** overeenkomstige HTML tag hebben die wordt gegenereerd bij uitvoer. Web controls zullen meerdere HTML tags genereren in allerlei combinaties om zo hun taak te vervullen. De gegenereerde HTML zal een hogere functie hebben dan enkelvoudige HTML tags. De objecten die gerelateerd zijn met een Web control zijn dan ook complexer.

Stel dat u op uw webpagina een tabel wil aanmaken die dienst doet als dynamische kalender: 7 kolommen van maandag tot zondag, met daarin de dagen van de huidige maand. Om dit te realiseren kan men perfect een dynamische tabel maken met een geneste for-lus. Men moet dan de huidige datum ophalen, het begin van deze maand zoeken, de dag van de begindag bepalen om zo te weten in welke kolom men moet starten, enz.

Eenvoudiger wordt het wanneer u een web control gebruikt die voorgeprogrammeerd is en die al deze taken voor u uitvoert. Bovendien kunt u web controls eenvoudigweg slepen naar de webpagina in kwestie.

```
<html>
<head>
</head>
<body>
  <form runat="server">
    <asp:Calendar id="calendar1" runat="server"/>
  </form>
</body>
</html>
```



Deze Web control genereert naast de tabel, ook nog javascript die het mogelijk maakt om naar de vorige of de volgende maand te gaan. Ook kunnen de dagen die zijn aangeklikt worden opgevangen en verwerkt worden. Volgende code is de HTML code die werd gegenereerd door één simpele Web control.

```
<html>
<head>
</head>
<body>
  <form name="_ctl0" method="post" action="aspnetH2_03.aspx" id="_ctl0">
<input type="hidden" name="__EVENTTARGET" value="" />
<input type="hidden" name="__EVENTARGUMENT" value="" />
<input type="hidden" name="__VIEWSTATE"
value="dDwtMTg1NDkwMjc0Nzs7Pogh8WRSmNwxZhX1Jg7Z2lqaQYxy" />

<script language="javascript" type="text/javascript">
<!--
        function __doPostBack(eventTarget, eventArgument) {
                var theform;
                if (window.navigator.appName.toLowerCase().indexOf("microsoft") > -1) {
                        theform = document._ctl0;
                }
                else {
                        theform = document.forms["_ctl0"];
                }
                theform.__EVENTTARGET.value = eventTarget.split("$").join(":");
                theform.__EVENTARGUMENT.value = eventArgument;
                theform.submit();
        }
// -->
</script>

    <table id="calendar1" cellspacing="0" cellpadding="2" border="0">
        <tr><td colspan="7" bgcolor="Silver">
<table cellspacing="0" border="0" width="100%">
<tr><td width="15%"><a href="javascript:__doPostBack('calendar1','V2039')"
style="color:Black">&lt;</a></td><td align="Center" width="70%">september 2005</td><td align="Right"
width="15%"><a href="javascript:__doPostBack('calendar1','V2100')" style="color:Black">&gt;</a>
</td></tr>
</table>
        </td></tr>
<tr><td align="Center">ma</td><td align="Center">di</td><td align="Center">wo</td><td
align="Center">do</td><td align="Center">vr</td><td align="Center">za</td><td align="Center">zo</td></tr>
<tr><td align="Center" width="14%"><a href="javascript:__doPostBack('calendar1','2067')"
style="color:Black">29</a></td><td align="Center" width="14%"><a
href="javascript:__doPostBack('calendar1','2068')" style="color:Black">30</a></td><td align="Center"
width="14%"><a href="javascript:__doPostBack('calendar1','2069')" style="color:Black">31</a></td><td
align="Center" width="14%"><a href="javascript:__doPostBack('calendar1','2070')"
style="color:Black">1</a></td><td align="Center" width="14%"><a
href="javascript:__doPostBack('calendar1','2071')" style="color:Black">2</a></td><td align="Center"
width="14%"><a href="javascript:__doPostBack('calendar1','2072')" style="color:Black">3</a></td><td
align="Center" width="14%"><a href="javascript:__doPostBack('calendar1','2073')"
style="color:Black">4</a></td></tr>
…….

</table>
  </form>
</body>
</html>
```

In dit voorbeeld hebben we enkel maar gebruik gemaakt van de kalender die meegeleverd wordt met de .NET-omgeving in de System.Web.UI.WebControls namespace, zonder dat we de attributen van deze kalender hebben gemanipuleerd.

Validatie-controls in meer detail

Alle input die een gebruiker in een applicatie ingeeft moet worden gevalideerd. Of met andere woorden, er moet worden nagekeken of deze voldoet aan bepaalde business rules. Deze validatie gebeurt meestal op verschillende niveaus, in de verschillende lagen.

In de databank worden er tabel- en kolomconstraints gelegd en referentiële integriteit wordt afgedwongen. In de business klassen zorgen constructors en methoden ervoor dat de attributen van een object worden ingekapseld. Als alle methoden correct geschreven zijn zullen de attributen steeds aan de bedrijfsregels voldoen. Ook in de userinterface gaat men de input controleren voordat deze wordt doorgegeven aan de business klasse. Om de ingegeven data reeds vanaf de bron te valideren (nog voor deze verwerkt wordt) zijn er speciale validatie-controls gecreëerd. Deze zijn eveneens mee opgenomen in de System.Web.UI.WebControls namespace.

Er zijn een vijftal types van validatie-controls en één rapporterings-control:

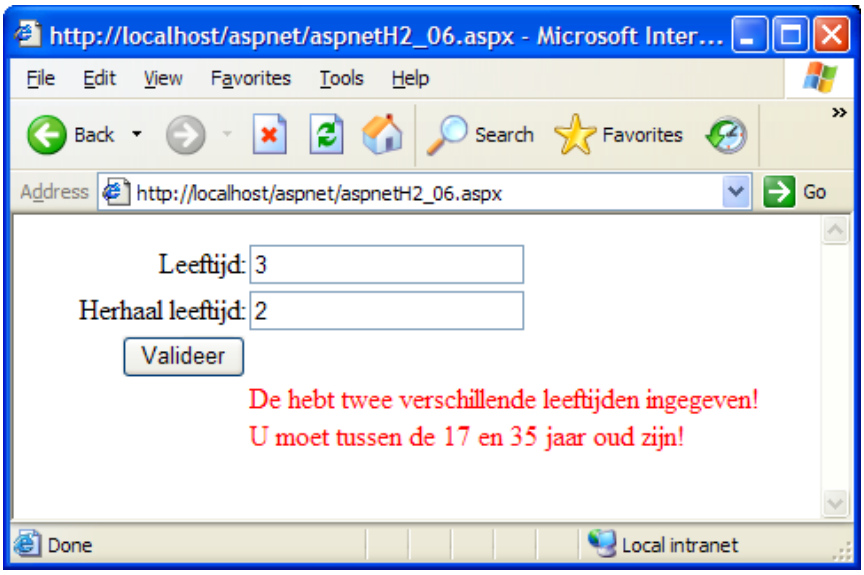
validatie control	**Omschrijving**
CompareValidator	Vergelijkt de ingegeven waarde met de waarde in een ander input veld of met een vaste waarde
CustomValidator	Hiermee kunt u een eigen validatie methode schrijven
RangeValidator	Controleert of de ingegeven waarde tussen twee waarden ligt
RegularExpressionValidator	Zorgt ervoor dat de ingegeven waarde aan een bepaald patroon voldoet (is numerisch, # karakters)
RequiredFieldValidator	Deze waarde MOET worden ingevuld
ValidationSummary	Met deze control kunt u alle validatiefouten weergeven naar de gebruiker

Omdat validatie-controls gegevens uit andere controls gaan valideren, worden deze steeds aan elkaar gekoppeld. Elke validatie-control kunt u terugkoppelen aan een ValidationSummary control.

De volgende voorbeelden geven u een idee van wat er mogelijk is met validatie-controls.

Oefening

1. **maak een nieuwe pagina “aspnetH2_validatiecontrols.aspx” waarin u een formulier aanmaakt dat er als volgt uitziet:**



opmerking: gebruik webcontrols!

2. **Voeg validatiecontrols toe zodanig dat uw formulier volgende mogelijkheden heeft:**

   ”Leeftijd dient tussen 17 en 35 jaar te zijn”
   ”U dient twee keer dezelfde leeftijd in te geven”
   ”Leeftijd mag niet leeg zijn”

3. **Indien aan deze voorwaarden niet voldaan is moet via een validation control een overeenkomstige fout weergegeven worden aan de gebruiker**

Resultaat: zonder 1 regel code te schrijven heb je een goede validatie op je formulier

REFERENTIE: VOORBEELD_010.ASPX

Oefeningen bij dit hoofdstuk

Oefening 1

Zorg voor volgende elementen in uw aspx-pagina:

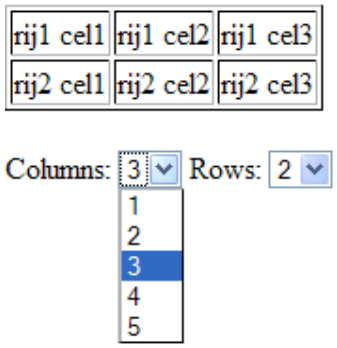
Placeholder “Placeholder1”
DropDownList “selectCols”
DropDownList “selectRows”

Via uw code-behind file dient u te zorgen voor volgende functionaliteit:

- Wanneer de pagina gestart wordt (page_load) moeten er vijf rijen en vijf colommen in een tabel getoond worden. Elke cel bevat op zijn beurt de beschrijving “rij 1 cel 1”, “rij 1 cel 2”, enzovoort.

rij1 cel1	rij1 cel2	rij1 cel3	rij1 cel4	rij1 cel5
rij2 cel1	rij2 cel2	rij2 cel3	rij2 cel4	rij2 cel5
rij3 cel1	rij3 cel2	rij3 cel3	rij3 cel4	rij3 cel5
rij4 cel1	rij4 cel2	rij4 cel3	rij4 cel4	rij4 cel5
rij5 cel1	rij5 cel2	rij5 cel3	rij5 cel4	rij5 cel5

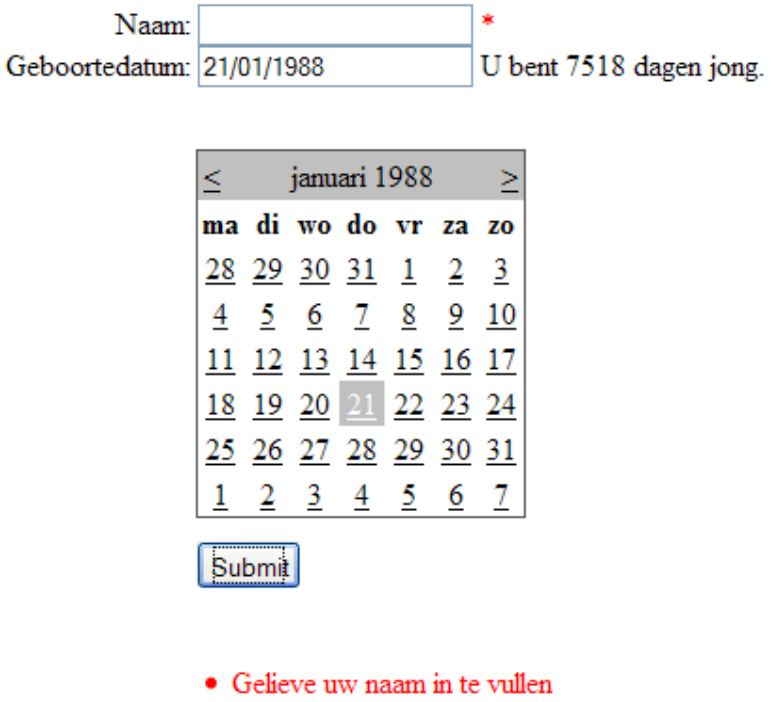
- Dit alles bouwt u op via HTML server controls. **Tip**: HtmlTable, HtmlTableCell, HtmlTableRow
- Wanneer dit werkt zorgt u ervoor dat via twee DropDownLists het aantal rijen en kolommen geselecteerd kan worden (max 5 rijen en max 5 kolommen). Deze dropdownlijsten vult u ook dynamisch op in uw code-behind
- Uiteindelijk dient u in uw ASPX-bestand slechts drie elementen toe te voegen: 1 placeholder en 2 dropdownlists. De rest van de opbouw gebeurt in uw code-file.



REFERENTIE: VOORBEELD_007.ASPX

Oefening 2

Maak volgende ASP.NET pagina “aspnetH2_eindoefening.aspx” met externe codefile.



De pagina bevat 2 input controls waarvan de eerste 'Naam' verplicht is in te vullen. De rode karakters geven aan wat er moet gebeuren op het moment dat het formulier wordt gevalideerd. Op de pagina staat ook een kalender. Als men daarop klikt dan wordt automatisch het veld 'geboortedatum' ingevuld en het aantal dagen sinds die dag wordt uitgerekend. Als u voor het eerst op deze pagina komt dan start de kalender op 1 januari van 20 jaar geleden.

Tips: Date.Today, DateTime.Now.Substract

REFERENTIE: ASPNETH2_EINDOEFENING.ASPX