

Using .NET controls with umbraco

This tutorial is a teaser of the coming Umbraco Developer Lounge product. If you like what you read, you'll love "the lounge". It's a site containing tutorials like this, best-practices, sample-code and much more. Umbraco Developer Lounge will be available in the second quarter of 2005, and will be priced at a yearly fee as low as €300 pr. User (or €30 a month).

You can use .NET controls together with umbraco whether it be UserControl (.ascx files) or CustomControls (.dll) by using macros. You're even able to communicate with your control by using Public Properties and macro elements.

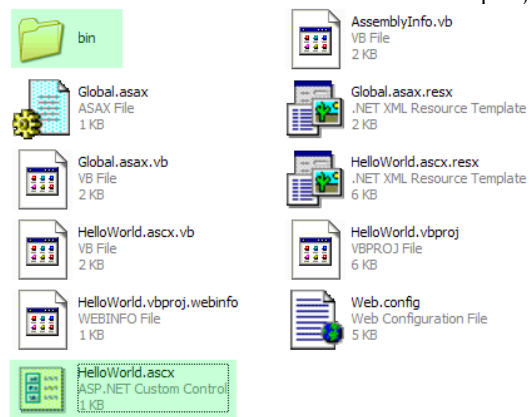
This document will tell you how to implement both types of controls and how to communicate.

Copying files

First you need to copy your control. Whether it's a usercontrol or a customcontrol you'll need to copy the assembly (dll) into the /bin folder of your umbraco installation. If it's a UserControl you'll need to copy the .ascx file as well. Your install already contains a /usercontrols folder which we'll recommend that you use, but you're free to place them anywhere in your application.

Example

We have a usercontrol called HelloWorld written in Visual Basic.NET containing a Textbox and a button. Let's have a look at the files in the project folder:

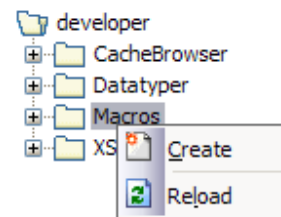


What we need here is the `HelloWorld.ascx` file at the project root and the `HelloWorld.dll` stored in the bin-folder.

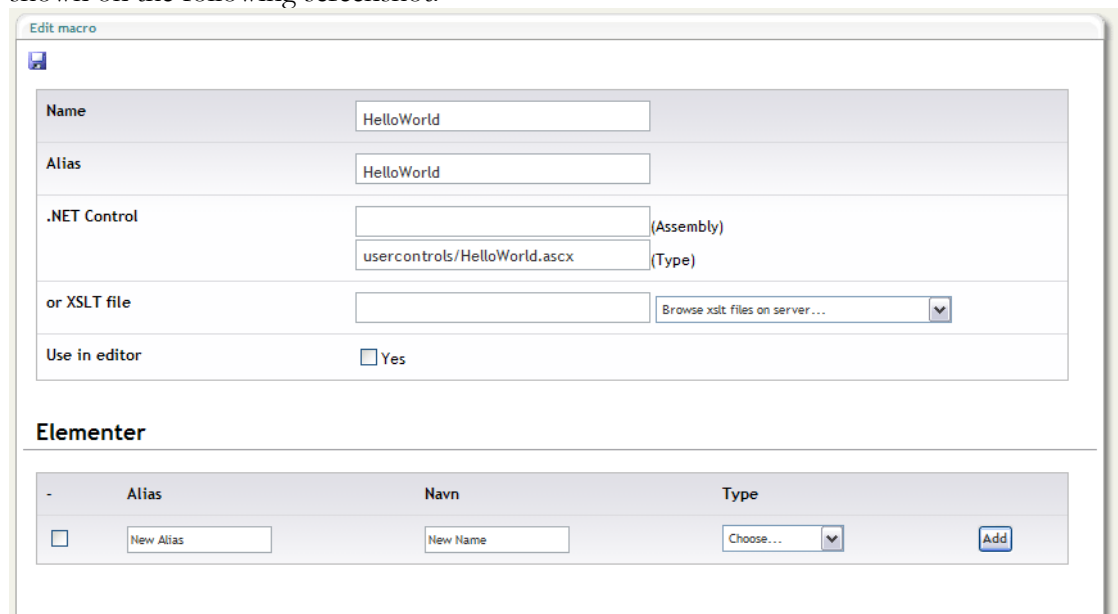
We'll copy `HelloWorld.ascx` into `/usercontrols/HelloWorld.ascx` and `HelloWorld.dll` will go into `/bin/HelloWorld.dll`. Now we're ready to register the usercontrol into umbraco using a macro!

Register .NET controls using macros

To be able to use the .NET controls directly from umbraco, it's necessary to register them into a macro. So open up umbraco, go to the "Developer" section and right click the macro-folder. Select Create and call it "HelloWorld".



Now select the HelloWorld macro in the opened macro folder and type "usercontrols/HelloWorld.ascx" into the field called "Type" as shown on the following screenshot:



Edit macro

Name: HelloWorld

Alias: HelloWorld

.NET Control: (Assembly)
usercontrols/HelloWorld.ascx (Type)

or XSLT file: Browse xslt files on server...

Use in editor: Yes

Elementer

-	Alias	Navn	Type	
<input type="checkbox"/>	New Alias	New Name	Choose...	Add

Hit the save button and you're ready to use your HelloWorld.ascx usercontrol just like any other macro. That's it! If you're using assemblies the assembly should be the name of your dll (without the dll-extension) and the type should be the full name including namespace!

Communicating through Elements and Public Properties

By creating Public Properties in your control and match them with macro elements, you're able to communicate with your control directly from within umbraco. umbraco will even create the UI when inserting the macro. It's important that the Alias of your macro element matches the name of your Public Property and be aware of case-sensitivity!

Example

First of – let's create a Public Property in our HelloWorld usercontrol and call it "HelloWorld" too. If not posted back – we'll set the text inside a TextBox to the contents of the Public Property – here's a simple way of doing it in Visual Basic.NET – you can use any .NET supported language you want (like c# og Delphi.NET):

```

Private _helloWorld As String = ""

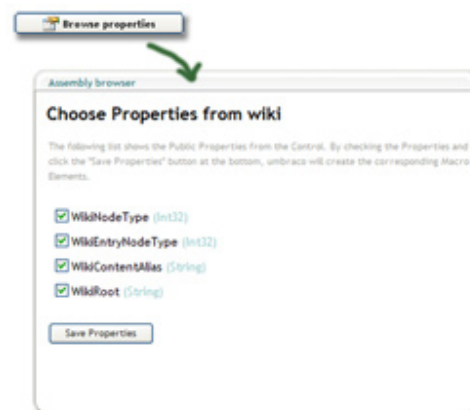
' Public variable which should be used to communicate with the control
' from umbraco
Public Property HelloWorld()
    Get
        Return _helloWorld
    End Get
    Set(ByVal Value)
        _helloWorld = Value
    End Set
End Property

Private Sub Page_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles
    If Not IsPostBack Then
        ' On postback - type the contents of the public property
        TextBox1.Text = HelloWorld
    End If
End Sub

```

Compile and copy the files again as described above.

Next we'll need to create a macro element, so go back to umbraco and edit the "HelloWorld" macro. The checkbox (with missing label) is whether the element should be showed to the user when the macro is inserted, so check it. The Alias of the element should be **exactly** the same as the name of the Public Property – and watch out, it's case sensitive. If you click the "browse properties" button, you can choose which public properties you'd like and umbraco will create them. The name of the element is what the user will see inside umbraco when the macro is inserted, so type something meaningful like "Message on postback". Next decide what type of element you're creating, when used together with .NET controls its only about the UI that the user will see (different from using XSLT macros where the type can influence the XML that the macro will use). Choose "Text" as we'll let the user type a message. Click add.




Elementer

-	Alias	Navn	Type	
<input checked="" type="checkbox"/>	HelloWorld	Message on postback	Choose... Choose... bool contentCurrent contentRandom contentSubs contentTree number text	Add

And that's all – now umbraco can communicate with your control! Let's try it!

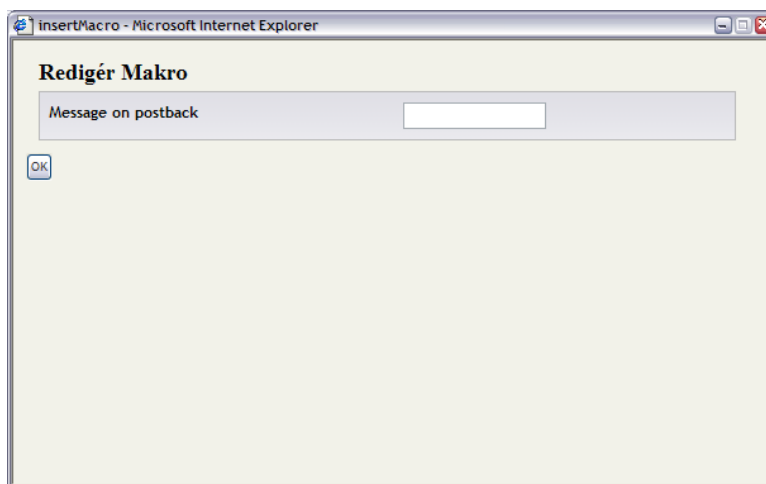
Inserting a macro

To insert our macro with the .NET control, you'll need to go to the Template section inside "Settings" or insert it directly inside the editor. In this sample will insert it into the template.

Open up a template – like "Textpage" if you're using the sample site. Make a new line before the "<?UMBRACO_GETITEM/>" tag and hit the insert macro button () at the toolbar. Choose the "HelloWorld" macro and you should see this:

.NET form elements and umbraco templates

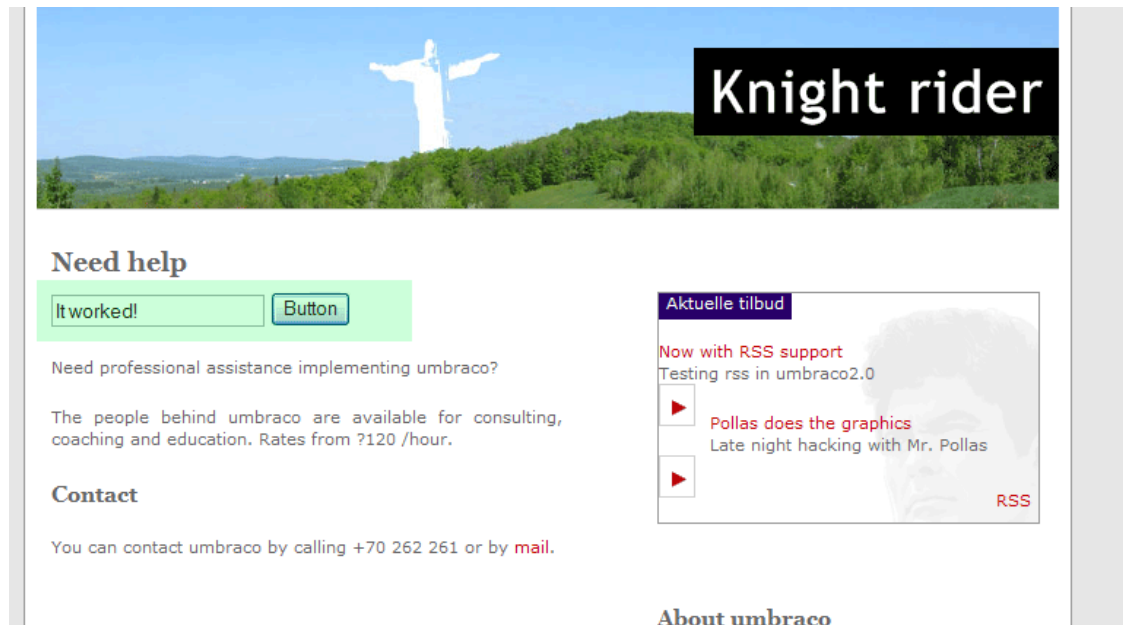
Umbraco does not by default insert a .NET serverside form into your content. If you're using server-side elements in your controls (with attribute runat="server"), then it's important that you insert the <?ASPNET_FORM> umbraco tag into the top and bottom of your template.



Type whatever you like and click ok. Your template should now look something like this:

```
<?UMBRACO_MACRO macroID="7" HelloWorld="It worked!"></?UMBRACO_MACRO>
<?UMBRACO_GETITEM field="bodyText"/>
```

Hit the save button and go see a page using your macro. It should look something like this:



Try changing the text in the textbox and hit the button. Umbraco will even take care of state...

Multiple .NET control macros on same page

If you're using more than one instance of the same .NET control macro on a page or if you just want to be sure of the Id given to the Control, you can add an extra attribute to the <?UMBRACO_MACRO> tag, called "controlID". To do so, the tag above will look as follows:

```
<?UMBRACO_MACRO macroID="7" HelloWorld="It worked!"
controlID="HelloWorld1"></?UMBRACO_MACRO>
```

As you can see it's easy to use .NET controls with umbraco – and you can use existing controls that aren't made for umbraco. Enjoy...