

Re: Passing parameters between form classes – C#

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vc/2005-02/0190.html>

From: Antti Keskinen (*antti.keskinen_at_REMOVEME.ee.tpu.fi*)

Date: 02/07/05

Date: Mon, 7 Feb 2005 11:59:50 +0200

Hi !

I've solved the case, but this approach has it's limits. First of, it does NOT use references. Second, it requires that while the lookup form is visible, the user cannot do anything else with the first form. Thirdly, it's not very object-oriented.

In the fclsLookup class, make sure you have the following variables:

```
private System.String m_strSource = String.Empty; // Zip Code passed by the
first form
public System.String m_strDestination = String.Empty; // Returns the
results of the lookup
```

Change fclsLookup's constructor to this:

```
public fclsLookup(string SourceZipCode)
{
    InitializeComponent();

    m_strSource = SourceZipCode; // Get the source zip code

    /* Rest of this function's code goes here as normal */
}
```

In the click handler, do this:

```
protected void dataGrid1_MouseDown(object sender,
    System.Windows.Forms.MouseEventArgs e)
{
    // Hit testing
    int x = dataGrid1.HitTest(e.X, e.Y).Row;

    // This is the value passed back to the first form
    m_strDestination = dataGrid1[x,0].ToString();
}
```

}

In the first form, where you create and show the lookup form, make sure it looks something like this:

Also, note the order in which the methods are called. This is very important.

```
private void ShowLookup_Click( object sender, System.EventArgs e )
{
    // Launch the second form (sourceZip is a control which takes the zip
code)
    fclsLookup Form2 = new fclsLookup( sourceZip.Text );

    // Display a modal version of the second form. Execution of this
function stops here until the lookup form
// is closed.
    Form2.ShowDialog();

    // After the form has been closed, execution continues.
    // When we are here, the click handler in the second form has updated
the
    // internal value m_strDestination, so we can copy it out
    System.String strReturnValue = objForm2.m_strDestination;

    // Use strReturnValue to display the lookup result information.
}
```

This is the only approach I could fathom. It seems that the idea of 'reference variables' that exists in C++ is a completely unknown feature in C#. In C#, you can only have reference variables that live inside a method. Reference variables inside a class are an unknown concept altogether. This is clearly a limitation of the language, and a one more reason to stick to C++ or C++/CLI, when it's finished.

–Antti Keskinen

"Johnny" <Johnny@discussions.microsoft.com> wrote in message
news:EE7EEB70–8EA2–47EF–80A3–C8D714D60C1E@microsoft.com...

> Antti,

>

> *I really appreciate the feedback. I should have stated the fact that I am*

> *passing two variables by reference but I don't know how to set the*

> *variable*

> *that I pass to the second form to the value selected in the second form.*

> *When I try to set the variable. I get a message from the compiler that*

> *states the variable does not exist in the namespace. My code is below.*

>

> *using System;*

> *using System.Drawing;*

> *using System.Collections;*

> *using System.ComponentModel;*

```

> using System.Windows.Forms;
> using System.Data;
> using System.Data.SqlClient;
>
> namespace EnspireUtilities
> {
> /// <summary>
> /// Summary description for Lookup.
> /// </summary>
> public class fclsLookup : System.Windows.Forms.Form
> {
> private System.Data.SqlClient.SqlDataAdapter sqlDataAdapter1;
> private System.Data.SqlClient.SqlCommand sqlSelectCommand1;
> private System.Data.SqlClient.SqlConnection sqlConnection1;
> private EnspireUtilities.dsLookup dsLookup1;
> private System.Windows.Forms.DataGrid dataGrid1;
> private System.Windows.Forms.Button btnLoad;
> private System.Windows.Forms.TextBox tboxLookupZipCode;
> private System.Windows.Forms.Label lblLookupParameter;
> private System.Windows.Forms.Button btnOK;
> private System.Windows.Forms.Button btnCancel;
> //private string strZipCode = string.Empty;
> private System.Windows.Forms.DataGridTableStyle dataGridTableStyle1;
> private System.Windows.Forms.DataGridTextBoxColumn dataGridTextBoxColumn1;
> private System.Windows.Forms.DataGridTextBoxColumn dataGridTextBoxColumn2;
> private System.Windows.Forms.DataGridTextBoxColumn dataGridTextBoxColumn3;
> private System.Windows.Forms.DataGridTextBoxColumn dataGridTextBoxColumn4;
> public string strZipCode = String.Empty;
> public string strZipCodeDtId = String.Empty;
>
>
> /// <summary>
> /// Required designer variable.
> /// </summary>
> private System.ComponentModel.Container components = null;
>
> public fclsLookup(ref string ZipCode, ref string ZipCodeDtId)
> {
> //
> // Required for Windows Form Designer support
> //
> InitializeComponent();
>
> //
> // TODO: Add any constructor code after InitializeComponent call
> //
> //tboxLookupZipCode.Text = ZipCode;
> sqlSelectCommand1.Parameters["@zip"].Value = ZipCode;
> dsLookup1.Clear();
> sqlDataAdapter1.Fill(dsLookup1);
> strZipCode = ZipCode;

```

```
> strZipCodeDtlId = ZipCodeDtlId;
> tboxLookupZipCode.Text = strZipCode;
> this.dataGrid1.MouseDown += new
> System.Windows.Forms.MouseEventHandler(this.dataGrid1_MouseDown);
>
> }
>
> /// <summary>
> /// Clean up any resources being used.
> /// </summary>
> protected override void Dispose( bool disposing )
> {
> if( disposing )
> {
> if(components != null)
> {
> components.Dispose();
> }
> }
> base.Dispose( disposing );
> }
>
> #region Windows Form Designer generated code
> /// <summary>
> /// Required method for Designer support – do not modify
> /// the contents of this method with the code editor.
> /// </summary>
> private void InitializeComponent()
> {
> this.sqlDataAdapter1 = new System.Data.SqlClient.SqlDataAdapter();
> this.sqlSelectCommand1 = new System.Data.SqlClient.SqlCommand();
> this.sqlConnection1 = new System.Data.SqlClient.SqlConnection();
> this.dsLookup1 = new EnspireUtilities.dsLookup();
> this.dataGrid1 = new System.Windows.Forms.DataGrid();
> this.dataGridTableStyle1 = new System.Windows.Forms.DataGridTableStyle();
> this.dataGridTextBoxColumn1 = new
> System.Windows.Forms.DataGridTextBoxColumn();
> this.dataGridTextBoxColumn2 = new
> System.Windows.Forms.DataGridTextBoxColumn();
> this.dataGridTextBoxColumn3 = new
> System.Windows.Forms.DataGridTextBoxColumn();
> this.dataGridTextBoxColumn4 = new
> System.Windows.Forms.DataGridTextBoxColumn();
> this.btnLoad = new System.Windows.Forms.Button();
> this.tboxLookupZipCode = new System.Windows.Forms.TextBox();
> this.lblLookupParameter = new System.Windows.Forms.Label();
> this.btnOK = new System.Windows.Forms.Button();
> this.btnCancel = new System.Windows.Forms.Button();
> ((System.ComponentModel.ISupportInitialize)(this.dsLookup1)).BeginInit();
> ((System.ComponentModel.ISupportInitialize)(this.dataGrid1)).BeginInit();
> this.SuspendLayout();
```

```

> //
> // sqlDataAdapter1
> //
> this.sqlDataAdapter1.SelectCommand = this.sqlSelectCommand1;
> this.sqlDataAdapter1.TableMappings.AddRange(new
> System.Data.Common.DataTableMapping[] {
> new System.Data.Common.DataTableMapping("Table",
> "p_getZipCodeData", new System.Data.Common.DataColumnMapping[] {
> new
> System.Data.Common.DataColumnMapping("zip", "zip"),
> new
> System.Data.Common.DataColumnMapping("city", "city"),
> new
> System.Data.Common.DataColumnMapping("county", "county"),
> new
> System.Data.Common.DataColumnMapping("state", "state"),
> new
> System.Data.Common.DataColumnMapping("zip_code_dtl_id",
> "zip_code_dtl_id")});
> //
> // sqlSelectCommand1
> //
> this.sqlSelectCommand1.CommandText = "[p_getZipCodeData]";
> this.sqlSelectCommand1.CommandType =
> System.Data.CommandType.StoredProcedure;
> this.sqlSelectCommand1.Connection = this.sqlConnection1;
> this.sqlSelectCommand1.Parameters.Add(new
> System.Data.SqlClient.SqlParameter("@RETURN_VALUE",
> System.Data.SqlDbType.Int, 4, System.Data.ParameterDirection.ReturnValue,
> false, ((System.Byte)(0)), ((System.Byte)(0)), "",
> System.Data.DataRowVersion.Current, null));
> this.sqlSelectCommand1.Parameters.Add(new
> System.Data.SqlClient.SqlParameter("@zip", System.Data.SqlDbType.VarChar,
> 10));
> //
> // sqlConnection1
> //
> this.sqlConnection1.ConnectionString = "workstation id=CENGENXP2;packet
> size=4096;user id=sa;data source=cengenxp2;persis" +
> "t security info=True;initial catalog=src;password=cocacola";
> //
> // dsLookup1
> //
> this.dsLookup1.DataSetName = "dsLookup";
> this.dsLookup1.Locale = new System.Globalization.CultureInfo("en-US");
> //
> // dataGrid1
> //
> this.dataGrid1.DataMember = "p_getZipCodeData";
> this.dataGrid1.DataSource = this.dsLookup1;
> this.dataGrid1.HeaderForeColor = System.Drawing.SystemColors.ControlText;

```

```
> this.dataGrid1.Location = new System.Drawing.Point(0, 40);
> this.dataGrid1.Name = "dataGrid1";
> this.dataGrid1.Size = new System.Drawing.Size(568, 288);
> this.dataGrid1.TabIndex = 0;
> this.dataGrid1.TableStyles.AddRange(new
> System.Windows.Forms.DataGridTableStyle[] {
> this.dataGridTableStyle1});
> this.dataGrid1.Navigate += new
> System.Windows.Forms.NavigateEventHandler(this.dataGrid1_Navigate);
> //
> // dataGridTableStyle1
> //
> this.dataGridTableStyle1.DataGrid = this.dataGrid1;
> this.dataGridTableStyle1.GridColumnStyles.AddRange(new
> System.Windows.Forms.DataGridColumnStyle[] {
> this.dataGridTextBoxColumn1,
> this.dataGridTextBoxColumn2,
> this.dataGridTextBoxColumn3,
> this.dataGridTextBoxColumn4});
> this.dataGridTableStyle1.HeaderForeColor =
> System.Drawing.SystemColors.ControlText;
> this.dataGridTableStyle1.MappingName = "p_getZipCodeData";
> //
> // dataGridTextBoxColumn1
> //
> this.dataGridTextBoxColumn1.Format = "";
> this.dataGridTextBoxColumn1.FormatInfo = null;
> this.dataGridTextBoxColumn1.HeaderText = "Zip Code";
> this.dataGridTextBoxColumn1.MappingName = "zip";
> this.dataGridTextBoxColumn1.Width = 75;
> //
> // dataGridTextBoxColumn2
> //
> this.dataGridTextBoxColumn2.Format = "";
> this.dataGridTextBoxColumn2.FormatInfo = null;
> this.dataGridTextBoxColumn2.HeaderText = "City";
> this.dataGridTextBoxColumn2.MappingName = "city";
> this.dataGridTextBoxColumn2.Width = 75;
> //
> // dataGridTextBoxColumn3
> //
> this.dataGridTextBoxColumn3.Format = "";
> this.dataGridTextBoxColumn3.FormatInfo = null;
> this.dataGridTextBoxColumn3.HeaderText = "County";
> this.dataGridTextBoxColumn3.MappingName = "county";
> this.dataGridTextBoxColumn3.Width = 75;
> //
> // dataGridTextBoxColumn4
> //
> this.dataGridTextBoxColumn4.Format = "";
> this.dataGridTextBoxColumn4.FormatInfo = null;
```

```
> this.dataGridTextBoxColumn4.HeaderText = "State";
> this.dataGridTextBoxColumn4.MappingName = "state";
> this.dataGridTextBoxColumn4.Width = 75;
> //
> // btnLoad
> //
> this.btnLoad.Location = new System.Drawing.Point(472, 8);
> this.btnLoad.Name = "btnLoad";
> this.btnLoad.TabIndex = 2;
> this.btnLoad.Text = "Load";
> this.btnLoad.Click += new System.EventHandler(this.btnLoad_Click);
> //
> // tboxLookupZipCode
> //
> this.tboxLookupZipCode.Location = new System.Drawing.Point(136, 8);
> this.tboxLookupZipCode.Name = "tboxLookupZipCode";
> this.tboxLookupZipCode.Size = new System.Drawing.Size(200, 20);
> this.tboxLookupZipCode.TabIndex = 1;
> this.tboxLookupZipCode.Text = "";
> //
> // lblLookupParameter
> //
> this.lblLookupParameter.Location = new System.Drawing.Point(24, 8);
> this.lblLookupParameter.Name = "lblLookupParameter";
> this.lblLookupParameter.Size = new System.Drawing.Size(100, 20);
> this.lblLookupParameter.TabIndex = 3;
> this.lblLookupParameter.Text = "Zip Code";
> this.lblLookupParameter.TextAlign =
> System.Drawing.ContentAlignment.MiddleRight;
> //
> // btnOK
> //
> this.btnOK.DialogResult = System.Windows.Forms.DialogResult.OK;
> this.btnOK.Location = new System.Drawing.Point(378, 352);
> this.btnOK.Name = "btnOK";
> this.btnOK.TabIndex = 3;
> this.btnOK.Text = "OK";
> this.btnOK.Click += new System.EventHandler(this.btnOK_Click);
> //
> // btnCancel
> //
> this.btnCancel.DialogResult = System.Windows.Forms.DialogResult.Cancel;
> this.btnCancel.Location = new System.Drawing.Point(472, 352);
> this.btnCancel.Name = "btnCancel";
> this.btnCancel.TabIndex = 4;
> this.btnCancel.Text = "Cancel";
> this.btnCancel.Click += new System.EventHandler(this.btnCancel_Click);
> //
> // fclsLookup
> //
> this.AutoScaleBaseSize = new System.Drawing.Size(5, 13);
```

```
> this.ClientSize = new System.Drawing.Size(568, 390);
> this.Controls.Add(this.btnCancel);
> this.Controls.Add(this.btnOK);
> this.Controls.Add(this.lblLookupParameter);
> this.Controls.Add(this.tboxLookupZipCode);
> this.Controls.Add(this.btnLoad);
> this.Controls.Add(this.dataGrid1);
> this.Name = "fclsLookup";
> this.Text = "Lookup";
> ((System.ComponentModel.ISupportInitialize)(this.dsLookup1)).EndInit();
> ((System.ComponentModel.ISupportInitialize)(this.dataGrid1)).EndInit();
> this.ResumeLayout(false);
>
> }
> #endregion
>
>
> private void btnLoad_Click(object sender, System.EventArgs e)
> {
> sqlSelectCommand1.Parameters["@zip"].Value = tboxLookupZipCode.Text;
> dsLookup1.Clear();
> sqlDataAdapter1.Fill(dsLookup1);
> }
>
> private void btnOK_Click(object sender, System.EventArgs e)
> {
> Close();
> }
>
> private void btnCancel_Click(object sender, System.EventArgs e)
> {
> Close();
> }
>
> private void dataGrid1_Navigate(object sender,
> System.Windows.Forms.NavigateEventArgs ne)
> {
> Close();
> }
>
> protected void dataGrid1_MouseDown(object sender,
> System.Windows.Forms.MouseEventHandler e)
> {
> string strMessage = "TEST";
>
> //strMessage = "Row: " + dataGrid1.HitTest(e.X,e.Y).Row.ToString();
> int x = dataGrid1.HitTest(e.X, e.Y).Row;
>
> strZipCode = dataGrid1[x, 0].ToString();
```

```
> strZipCodeDtId = dataGrid1[x, 3].ToString();
> MessageBox.Show(strZipCode);
> MessageBox.Show("I still don't know what I'm doing, but I'm at " +
> strMessage + ". I need to send back " + strZipCode);
> }
>
> }
> }
>
> "Antti Keskinen" wrote:
>
>> Hello !
>>
>> There are no stupid questions.
>>
>> The forms you define in C# always consist of a class. A single form is a
>> single class. Thus, your problem comes from the interaction between
>> classes.
>> You already knew this, however.
>>
>> So, in addition to passing a reference variable to the zip code the user
>> has
>> entered (the one which you wish to check), pass in another reference to
>> the
>> ACTUAL zip code returned by the lookup form. Now, when the lookup form is
>> launched, you use the first reference to pass in the zipcode and
>> initialize
>> the datagrid to show possible values, and the second reference to pass
>> back
>> the value which the user clicked on in the datagrid.
>>
>> Here's the most simplest example I can think of. It's written in C++, but
>> the language syntax is pretty much similar in C#.
>>
>> class MyForm1
>> {
>> public:
>> int m_nUsersZipCode;
>> int m_nDatagridResult;
>> MyForm2* m_pLookupForm;
>> };
>>
>> class MyForm2
>> {
>> public:
>> DWORD ValidateZipCode(int& nRefSource, int& nRefSelected);
>> };
>>
>> int main(void)
>> {
>> MyForm1 form1;
```

```
>> MyForm2 form2;
>> form1.Show();
>> }
>>
>> // In the ShowLookupForm handler of MyForm1
>> ..
>> m_pLookupForm->ValidateZipCode( m_nUsersZipCode,
>> m_nDatagridResult );
>> ..
>>
>> This would be the most simplistic way to accomplish it. The problem
>> itself
>> is not C# related by nature, thus it's quite easy to answer it.
>>
>> -Antti Keskinen
>>
>> "Johnny" <Johnny@discussions.microsoft.com> wrote in message
>> news:8E2E1AD0-B607-4671-B2BE-1B8BCB10AF3D@microsoft.com...
>> > I'm a rookie at C# and OO so please don't laugh! I have a form
>> > (fclsTaxCalculator) that contains a text box (tboxZipCode) containing a
>> > zip
>> > code. The user can enter a zip code in the text box and click a button
>> > to
>> > determine whether the zip code is unique. If the zip code is not
>> > unique,
>> > another form/dialog is displayed (fclsLookup) – lookup form/dialog.
>> > The
>> > zip
>> > code is passed to the lookup form/dialog by reference. I then load a
>> > datagrid with the possible matches (i.e. zip, city, county, state,
>> > zipid).
>> > When the user clicks the row in the datagrid, I want to pass the zipid
>> > back
>> > to the original form/dialog and run a stored procedure. My problem is
>> > I
>> > do
>> > not know how to get the ZipId back to the fclsTaxCalculator
>> > form/dialog.
>> > I've search for examples of this and I've not been able to track one
>> > down –
>> > please help!!!
>>
>>
>>
```