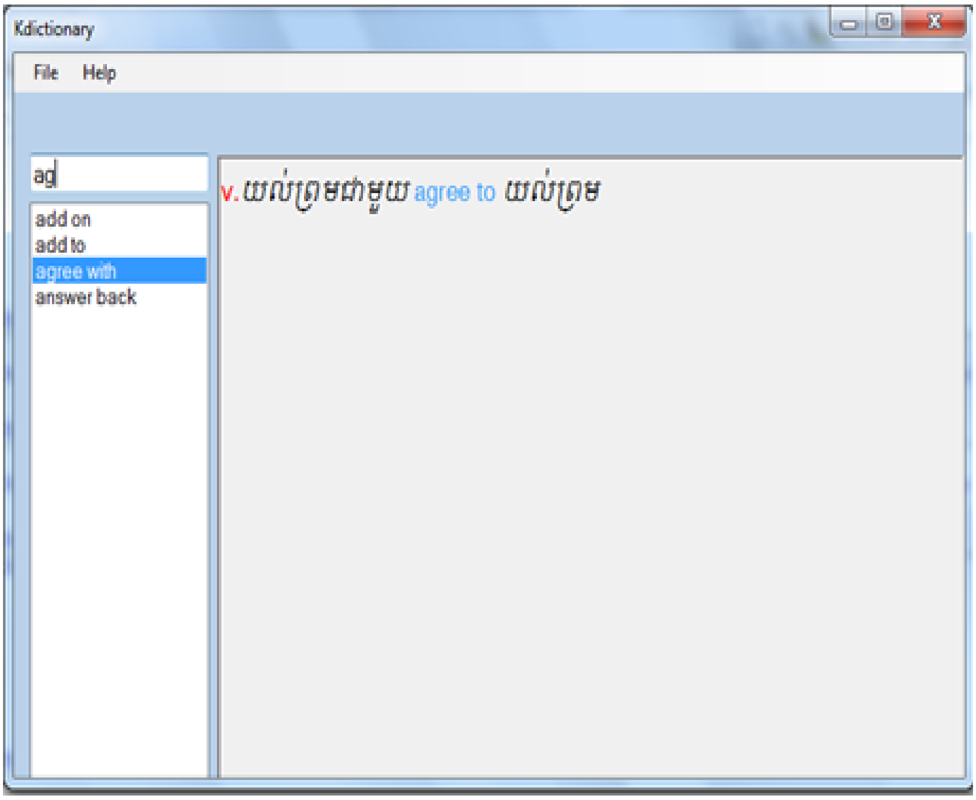
C# tutorial-A simple dictionary

In this C# sample program, you will learn to create a simple dictionary. The key terms are English words and each of English word is translated in to Khmer language when the word is selected from the list or when it is typed in to the search box. The word that is typed in the search textbox is searched through the list. If it is found, the word will be selected in the list and its translation will display in a right-hand rich text box.

We use Microsoft Access 2007 database to store the English words and their translations. When the dictionary form first loads, the English words will be retrieved from the database file and populated in to a list box.



ag

add on
add to
agree with
answer back

v.យល់ព្រមជាមួយ agree to យល់ព្រម

In the rich textbox, the English words will be colored in different colors rather than black. The translation worlds (Khmer) will be colored in black.

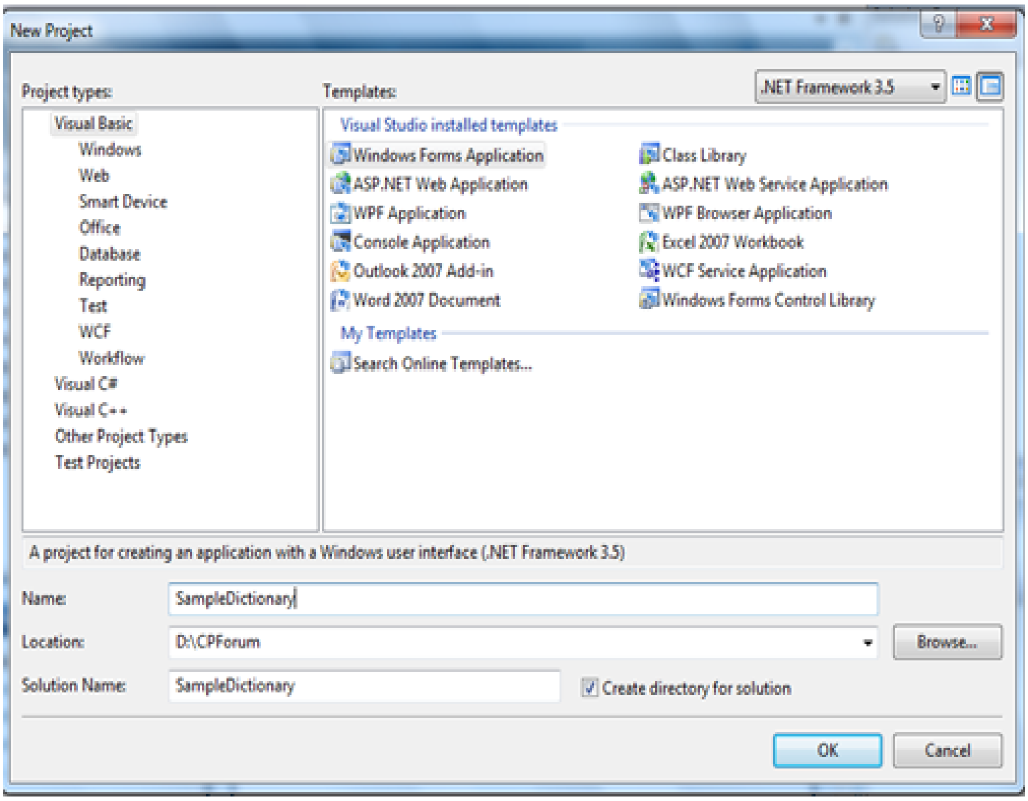
Interface Design

Now, you understand overall idea of project. Let’s move to the interface design phase. To complete this phase you need to follow the steps below:

-Open Visual Studio 2008 and create a new project by going to File menu-->select New and click on Project...

-The New Project dialog opens.

-In the New Project dialog, under **Project Types** select Visual C#, and select Windows Form Application under **Tamplates.** In the Name box type SampleDictionary.



A project for creating an application with a Windows user interface (.NET Framework 3.5)

Name: SampleDictionary

Location: D:\CPForum

Solution Name: SampleDictionary

☑ Create directory for solution

-You can click Browser you choose the location to store your project files.

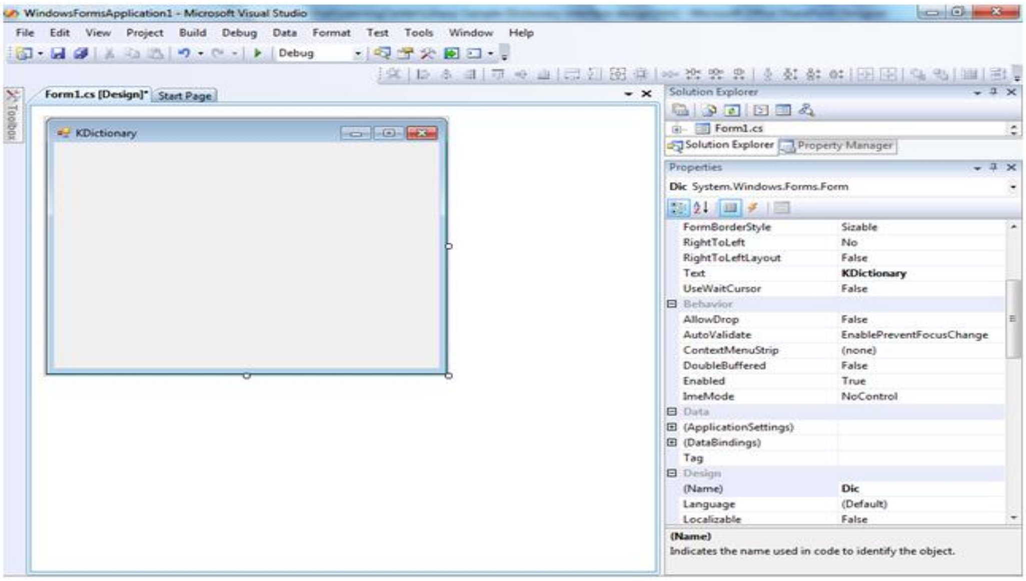
-Click Ok. Then you will see a default form named Form1.

-Right-Click on the form and select properties to display the properties window of the form.

-On the Property Manager tab. You will need to set the Text property to KDictionary and the Name property to Dic.

-Set the form Backcolor property to Inactivecaption.

-Set the form FormBorderStyle property to FixedDialog to make the form borders unresizable.



Dic System.Windows.Forms.Form

FormBorderStyle	Sizable
RightToLeft	No
RightToLeftLayout	False
Text	**KDictionary**
UseWaitCursor	False
⊟ Behavior	
AllowDrop	False
AutoValidate	EnablePreventFocusChange
ContextMenuStrip	(none)
DoubleBuffered	False
Enabled	True
ImeMode	NoControl
⊟ Data	
⊞ (ApplicationSettings)	
⊞ (DataBindings)	
Tag	
⊟ Design	
(Name)	**Dic**
Language	(Default)
Localizable	False

(Name)
Indicates the name used in code to identify the object.

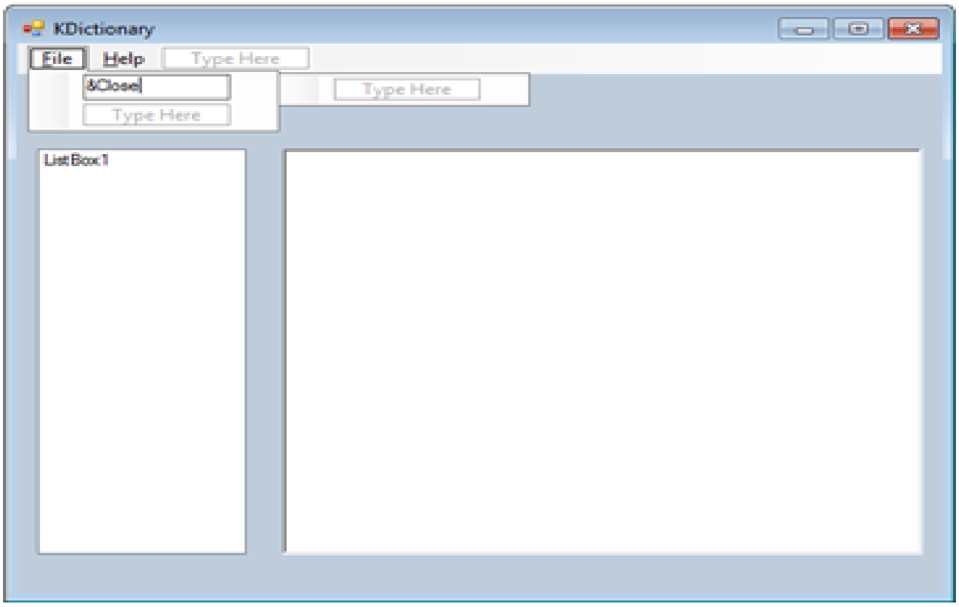
-Expand the Toolbox on the left side to display a list of Visual Basic controls and components.

-Drag and drop one ListBox on to the form.

-Rename the list box by setting its Name property to Lstterms.

-Drag and drop one Textbox on the form above the list box. Then drag and drop one RichTextbox to the right of the list box. You need to set the Name property of the textbox to Txtbox, and set the Name property of the rich text box to Txtresult. The fore color of the rich textbox is set to MenuHighlight by using the its Forecolor property. Its ReadOnly property is also set to True. This will not allow users to edit the text in the rich text box.

-To add a menu to the form, you need to drag and drop MenuScript from the Toolbox.



-Type &File in The Type Here box and in the next Type Here type &Help.

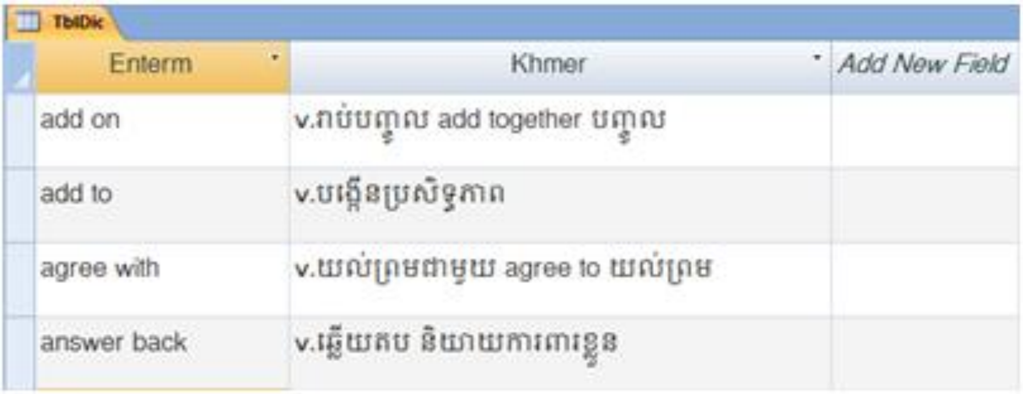
-Type &Close in to the Type Here box under the File menu and type &About... in to the Type Here box under the Help menu.

Now we finish the interface design of our simple dictionary. Then we will move to Create Database phase.

Create a database

As you already knew, to store the English key terms and their Khmer translations, we need a Microsoft Access 2007 database. Now lets create a database called **data.accdb** in Microsoft Access 2007 and save it in the **bin/debug** directory of your project. This database has one table called Tbldic that has only two fields--Enterm and Khmer.

The sample data is entered to each field as shown below:



TblDic

Enterm	Khmer	Add New Field
add on	v.ភប់បញ្ចូល add together បញ្ចូល	
add to	v.បង្កើនប្រសិទ្ធភាព	
agree with	v.យល់ព្រមជាមួយ agree to យល់ព្រម	
answer back	v.ឆ្លើយតប និយាយការពារខ្លួន	

After you create a database and enter some sample data, you can move to Create connection and populate data in the list phase.

Create connection to the database file

The code below will help you to connect to the database file and retrieve data to put in the Lstterm listbox:

-Double-click on the form to open the code window

-You will see the code as shown below:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;


namespace KDictionary
{
public partial class Dic : Form


{
public Dic()
{
InitializeComponent();
}

private void Dic_Load(object sender, EventArgs e)
{

}
}
}
```

You need to add the statement Using System.Data.OleDb; to the Using group statements in the code above. So it will be come:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.OleDb ;

namespace KDictionary
{
public partial class Dic : Form


{
public Dic()
{
InitializeComponent();
}

private void Dic_Load(object sender, EventArgs e)
{

}
}
}
```

-Under the "public partial class Dic: Form" line type the following code:

```
private OleDbConnection cn;
private OleDbDataReader reader;
private OleDbCommand com;
private DataSet rs;
private OleDbDataAdapter ad;
```

This code will declare cn variable as the OleDbConnection class, reader variable as OleDbReader class, com variable as OleDbReader Class, rs variable as DatSet class, and ad as OleDbDataAdapter classs. These variables will be used to create their objects of their types.

-Create a method called myconnect by typing the code as shown below:

```
private void myconnect(){
try
{
```

```
//connect to database
rs = new DataSet();
cn = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" +
Application.StartupPath + "\\data.accdb;");
cn.Open();
ad = new OleDbDataAdapter("SELECT * FROM Tblterms ORDER BY Enterm", cn);
ad.Fill(rs, "Tblterms");
com = new OleDbCommand("SELECT Enterm FROM Tblterms ORDER BY Enterm", cn);
reader = com.ExecuteReader();

//clear list
Lstterms.Items.Clear();
//clear txtresult
Txtresult.Text = "";
while (reader.Read())
{
Lstterms.Items.Add(reader[0].ToString());

}
}
catch (Exception ex) { }
Txtbox.Focus();



}
```

The code starts by create a DataSet object to store the table got from the database and a connection object to connect to the database file:

```
//connect to database
rs = new DataSet();
cn = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" +
Application.StartupPath + "\\data.accdb;");
cn.Open();
```

The Adapter object is created to get data from the Tblterms table and

the data is filled to the DataSet object rs by using the Fill() method of the adapter:

```
ad = new OleDbDataAdapter("SELECT * FROM Tblterms ORDER BY Enterm", cn);
ad.Fill(rs, "Tblterms");
```

The Command object com also is also created to get data from the table and populate the list box:

```
com = new OleDbCommand("SELECT Enterm FROM Tblterms ORDER BY Enterm", cn);
reader = com.ExecuteReader();
//clear list
Lstterms.Items.Clear();
//clear txtresult
Txtresult.Text = "";
```

```
while (reader.Read())
{
Lstterms.Items.Add(reader[0].ToString());

}
}
```

The search text box is set the focus to make sure that the cursor is active in this box for text input.

```
Txtbox.Focus();
```

By putting the method myconnect() in the form load event, the list box Lstterms will be populated by the English words:

```
private void Dic_Load(object sender, EventArgs e)
{
myconnect();

}
```

Note: we use try...catch block to catch any errors that may occur.

Now you have data filled in the list box Lstterms when the form loads. The next phase is to write some C# code to create a workable search text box.

Create a workable search text box

To make the search box workable you need the following code that is applied to the TextChanged event of the text box:

```
private void Txtbox_TextChanged(object sender, EventArgs e)
{
int i;
DataRow dr;
try
{
for (i = 0; i <= Lstterms.Items.Count - 1; i++)
{

if (Txtbox.Text.Trim().ToUpper() == Lstterms.Items[i].ToString().ToUpper().Substring(0,
Txtbox.TextLength))
{
//Select matched term
Lstterms.SelectedIndex = i;
//Display translation
dr = rs.Tables[0].Rows[i];
Txtresult.Clear();
Txtresult.Text = dr[1].ToString().Trim();
highlight();
break;
}
}
}
catch (Exception ex) { }
}
```

When a user types a word in the search text box TxtBox, This word is used to compare with a word in the list box. If the words are matched the word in the list will be selected and its translation will display in the rich text box Txtresult. The highlight() procedure is called to change colors of the English key terms in the rich text box Txtresult. This procedure will be discussed in the next phase.

If the use chooses to select from the list rather than typing in to the search box, you need another piece of code applied to the SelectedIndexChanged event of the list box:

```
private void Lstterms_SelectedIndexChanged(object sender, EventArgs e)
{

DataRow dr;
dr = rs.Tables[0].Rows[Lstterms.SelectedIndex];
Txtresult.Clear(); //clear result
Txtresult.Text =dr[1].ToString().Trim ();
highlight();

}
```

Note: The DataRow dr is used to get data of a selected row of the table.

Change colors of the English characters and Khmer translation characters

To change colors of the English characters and Khmer characters in the rich textbox, you need a highlight() method as shown below:

```
private void highlight()
{

try
{
char[] c = Txtresult.Text.ToString().ToCharArray();
//coloring v.,a., and n. in red
int i;
for (i = 1; i <= Txtresult.Text.ToString().Length - 2; i++)
{
if ((c[i] == '.' && c[i - 1] == 'v') || (c[i] == '.' && c[1 - 1] == 'a') || (c[i] == '.' && c[i - 1] == 'n'))
//If (((c[i] == ".") && (c[i - 1] == "v")) || ((c[i] == ".") && (c[i - 1] == "a")) || ((c[i] == ".") && (c[i
- 1] =="n")))
{
Txtresult.SelectionStart = i - 1;
Txtresult.SelectionLength = 2;
Txtresult.SelectionColor = Color.Red;

}
}
//coloring adj., adv. in red
for (i = 3; i <= Txtresult.Text.ToString().Length - 1; i++)
{


if ((c[i] == '.' && c[i - 1] == 'j' && c[i - 2] == 'd') || (c[i] == '.' && c[i - 1] == 'g' && c[i - 2] == 'i') ||
(c[i] == '.' && c[i - 1] == 'v' && c[i - 2] == 'd'))
{
Txtresult.SelectionStart = i - 3;
Txtresult.SelectionLength = 4;
Txtresult.SelectionColor = Color.Red;
}

}
//coloring Khmer characters in black color

for (i = 0; i <= Txtresult.Text.ToString().Length - 1; i++)
{

if ((int)c[i] > 127)
{

Txtresult.SelectionStart = i;
Txtresult.SelectionLength = 1;
Txtresult.SelectionColor = Color.Black;

}

}


}
catch (Exception ex) { }

}
```

By using the above code, the English text such as v., a., n., adj., and adv. will be in red color and the Khmer characters will be in black color. The rest of the text in the rich textbox is MenuHighlight color style.

Make a smarter form

Now if you maximize the form, the items are not smart enough to fit the form. What we want to talk in this page is to help you create a form that its items still stay fit with the form when it is resized for maximum state. To accomplish this task you need the following code that is applied to the Resize event of the form:

```
private void Dic_Resize(object sender, EventArgs e)
{
//Adjust listbox and textbox heights when the form resizes
//Dic dictionary = new Dic();


if (this.WindowState == FormWindowState.Maximized)
{
Lstterms.Height = this.Height;
Txtresult.Height = this.Height;
Txtresult.Width = this.Width ;
}



}
```

Note: If you try to resize the form and the code doesn't do its job you need to add another line of code(as shown below) to the form load event:

```
 private void Dic_Load(object sender, EventArgs e)
{
myconnect();
this.Resize+= new EventHandler(Dic_Resize); //add this code

}
```

In the next phase, we will talk about cleaning up the objects from the memory when the dictionary is closed.

Clean objects up

When the form loads many objects such as cn, rs, ad, com,and reader, are also created in the program for uses and these objects occupy spaces in computer memory. When the form is closed these objects should be released from the memory.

```
private void Dic_FormClosed(object sender,EventArgs e)
{
cn.Close();
cn = null;
com = null;
reader = null;
ad = null;
rs = null;



}
```

You also need to add code: this.FormClosed+=new FormClosedEventHandler(Dic_FormClosed); to the form load event to make Dic_FormClosed() method callable.

```
private void Dic_Load(object sender, EventArgs e)
{
myconnect();
this.Resize+= new EventHandler(Dic_Resize);
this.FormClosed += new FormClosedEventHandler(Dic_FormClosed);

}
```

Another thing we want to talk here is to let the user close the program when his/her clicks on the Close sub menu item form the File menu. This task can be done simply by attaching the following code to the click event of the Close sub menu item:

```
private void closeToolStripMenuItem_Click(object sender, EventArgs e)
{

Close();

}
```

Now, let test your program and enjoy with it. If you have any questions regard this sample project, we recommend you to post them in our computer forum found in top menu of every web page.

Putting the C# code together

This is the complete C# code of the simple dictionary:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.OleDb;


namespace KDictionary
{
public partial class Dic : Form{
private OleDbConnection cn;
private OleDbDataReader reader;
private OleDbCommand com;
private DataSet rs;
private OleDbDataAdapter ad;


public Dic()
{
InitializeComponent();
}


private void Dic_Resize(object sender, EventArgs e)
{
```

```
//Adjust listbox and textbox heights when the form resizes
//Dic dictionary = new Dic();


if (this.WindowState == FormWindowState.Maximized)
{
Lstterms.Height = this.Height;
Txtresult.Height = this.Height;
Txtresult.Width = this.Width ;
}



}


private void Dic_Load(object sender, EventArgs e)
{
myconnect();
this.Resize+= new EventHandler(Dic_Resize);
this.FormClosed += new FormClosedEventHandler(Dic_FormClosed);

}

private void myconnect(){
try
{

//connect to database
rs = new DataSet();
cn = new OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=" +
Application.StartupPath + "\\data.mdb;");
cn.Open();
ad = new OleDbDataAdapter("SELECT * FROM Tblterms ORDER BY Enterms", cn);

ad.Fill(rs, "Tblterms");
com = new OleDbCommand("SELECT Enterms FROM Tblterms ORDER BY Enterms", cn);
reader = com.ExecuteReader();

//clear list
Lstterms.Items.Clear();
//clear txtresult
Txtresult.Text = "";
while (reader.Read())
{
Lstterms.Items.Add(reader[0].ToString());

}
}
catch (Exception ex) { }
Txtbox.Focus();
```

```
}

private void Txtbox_TextChanged(object sender, EventArgs e)
{
int i;
DataRow dr;
try
{
for (i = 0; i <= Lstterms.Items.Count - 1; i++)
{

if (Txtbox.Text.Trim().ToUpper() == Lstterms.Items[i].ToString().ToUpper().Substring(0,
Txtbox.TextLength))
{
//Select matched term
Lstterms.SelectedIndex = i;
//Display translation
dr = rs.Tables[0].Rows[i];
Txtresult.Clear();
Txtresult.Text = dr[1].ToString().Trim();
highlight();
break;
}
}
}
catch (Exception ex) { }
}

private void Lstterms_SelectedIndexChanged(object sender, EventArgs e)
{

DataRow dr;
dr = rs.Tables[0].Rows[Lstterms.SelectedIndex];
Txtresult.Clear(); //clear result
Txtresult.Text =dr[1].ToString().Trim ();
highlight();

}
private void highlight()
{

try
{
char[] c = Txtresult.Text.ToString().ToCharArray();
//coloring v.,a., and n.
int i;
for (i = 1; i <= Txtresult.Text.ToString().Length - 2; i++)
{
if ((c[i] == '.' && c[i - 1] == 'v') || (c[i] == '.' && c[1 - 1] == 'a') || (c[i] == '.' && c[i - 1] == 'n'))
//If (((c[i] == ".") && (c[i - 1] == "v")) || ((c[i] == ".") && (c[i - 1] == "a")) || ((c[i] == ".") && (c[i
- 1] =="n")))
{
Txtresult.SelectionStart = i - 1;
Txtresult.SelectionLength = 2;
Txtresult.SelectionColor = Color.Red;

}

}
//coloring adj., adv.
for (i = 3; i <= Txtresult.Text.ToString().Length - 1; i++)
{


if ((c[i] == '.' && c[i - 1] == 'j' && c[i - 2] == 'd') || (c[i] == '.' && c[i - 1] == 'g' && c[i - 2] == 'i') ||
(c[i] == '.' && c[i - 1] == 'v' && c[i - 2] == 'd'))
{
Txtresult.SelectionStart = i - 3;
Txtresult.SelectionLength = 4;
Txtresult.SelectionColor = Color.Red;
}

}
//coloring Khmer character

for (i = 0; i <= Txtresult.Text.ToString().Length - 1; i++)
{

if ((int)c[i] > 127)
{
//int ix = (int)c[i];
//MessageBox.Show("Hello"+ix);
Txtresult.SelectionStart = i;
Txtresult.SelectionLength = 1;
Txtresult.SelectionColor = Color.Black;

}

}



}
catch (Exception ex) { }

}

private void closeToolStripMenuItem_Click(object sender, EventArgs e)
{

Close();
```

```
}



private void menuStrip1_ItemClicked(object sender, ToolStripItemClickedEventArgs e)
{

}
private void Dic_FormClosed(object sender,EventArgs e)
{
cn.Close();
cn = null;
com = null;
reader = null;
ad = null;
rs = null;


}


}
}
```

Now, let test your program and enjoy with it. If you have any questions regard this sample project, we recommend you to post them in our computer forum found in top menu of every web page.