The 50 Things You Need to Know about VB6 to VB.NET (Beginner's Guide)

Assumption: Using Visual Studio 2008 SP1

This documentation documented the first few things that you should worry about when converting your application from VB6 to VB.NET

This is will give you an idea of how much work you have to do in terms of the conversion.

One of the hardest part is GUI, such as usage of ListView.

This eBook is written by Brandon Teoh based on actual experience of performing VB6 to VB.NET conversion.

Contact me for discrepancies. (like if you can't download the sample codes or the codes doesn't work as it is supposed to).

Contacts:

Email: brandonteohno1@yahoo.com
Blogs: www.it-sideways.com

Table of contents

40. Exit Application	
41. Combo Box	
42. ByVal & ByRef	
43. Create Leading Zeros	
44. Generate Random Numbers	
45. StdPicture	
46. DCOM	
47. ‘Set’ Keyword	
48. Array	
49. Printers (Getting A Listing of Installed Printers)	
50. Crystal Report	

1.0 Assembly

The term assembly is quite confusing.

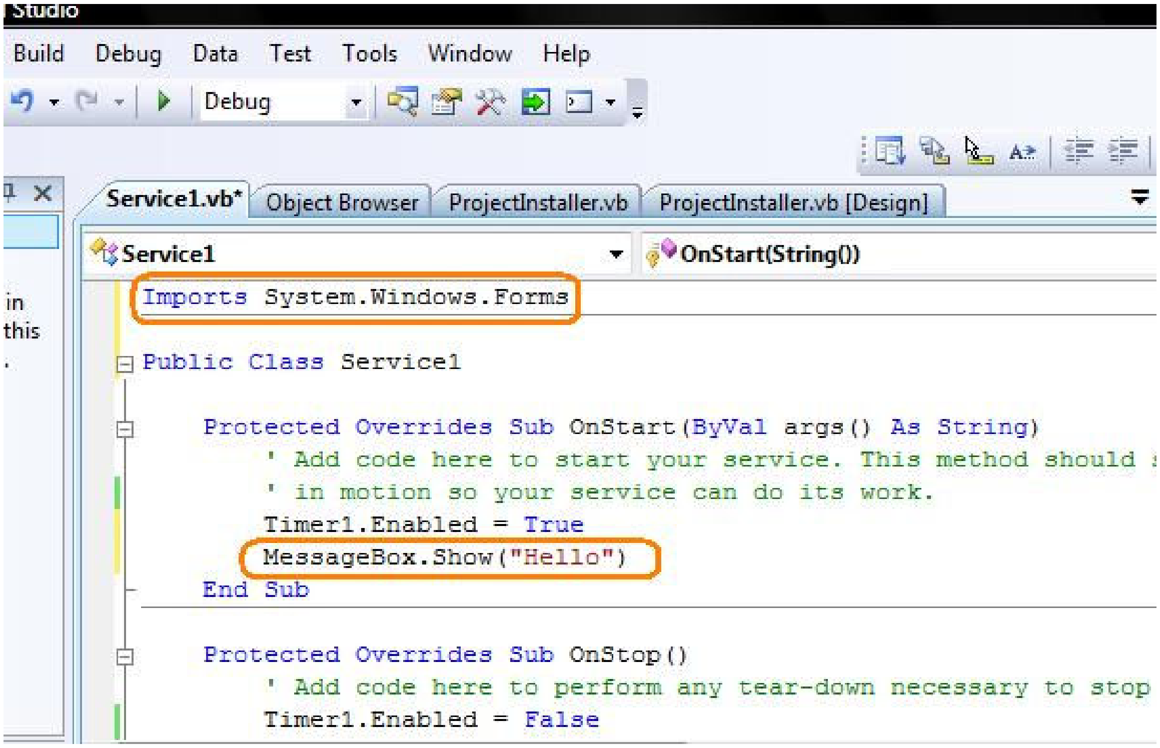
http://en.wikipedia.org/wiki/.NET_assembly

A partially compiled codes used for deployment.

Is either an exe or a DLL (as we know it in VB6).

2.0 Messagebox

If the project is not ‘Windows Forms Application’, you need to ‘Imports System.Windows.Forms’



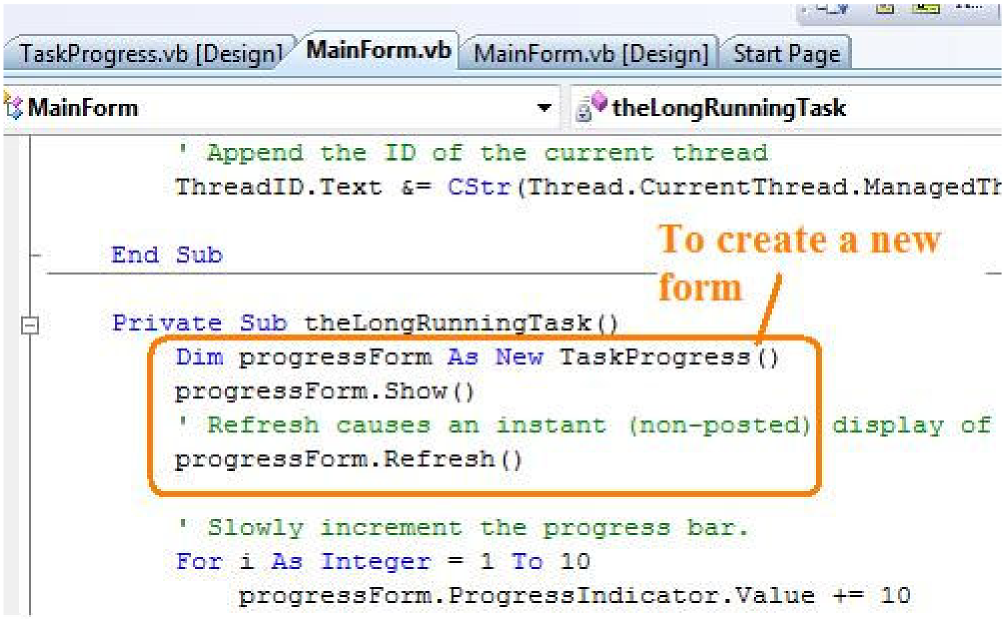
```
Imports System.Windows.Forms

Public Class Service1

    Protected Overrides Sub OnStart(ByVal args() As String)
        ' Add code here to start your service. This method should
        ' in motion so your service can do its work.
        Timer1.Enabled = True
        MessageBox.Show("Hello")
    End Sub

    Protected Overrides Sub OnStop()
        ' Add code here to perform any tear-down necessary to stop
        Timer1.Enabled = False
```

3.0 Windows Form

Windows form is not loaded by default. You need to create a new instance to use it and also to reference it. In other words, in VB.NET you need to make use of an object reference (to a windows form) to use it such as loading it, updating the text box and etc.

3.1 Loading Form:

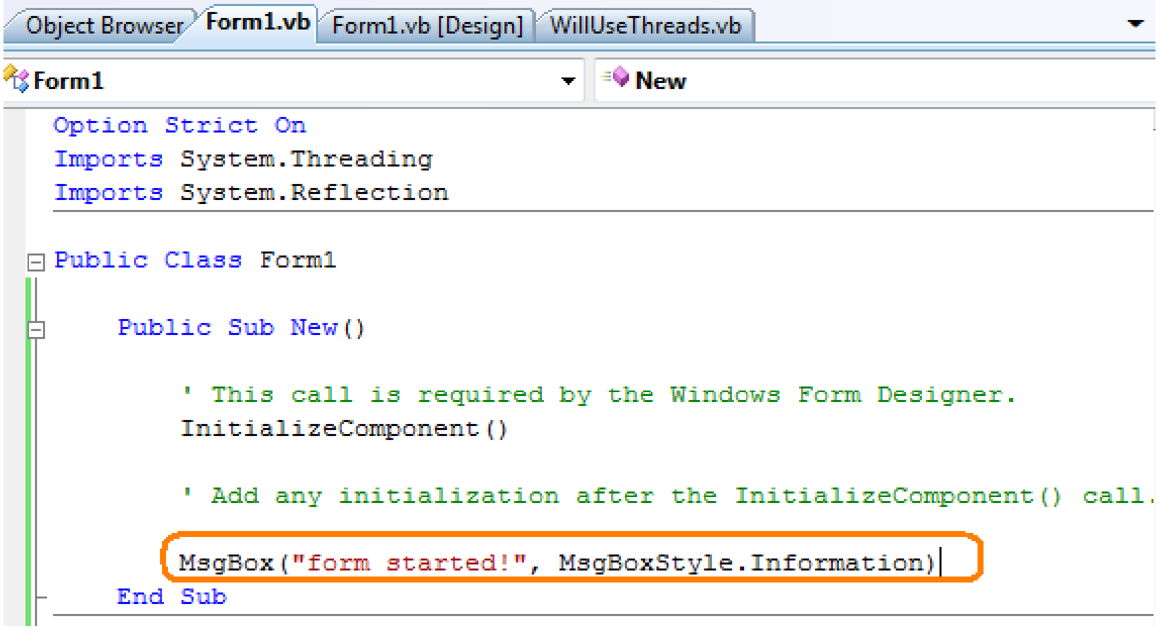
Create a new form instance and only call the ‘show’ method.



To create a new form

```
        ' Append the ID of the current thread
        ThreadID.Text &= CStr(Thread.CurrentThread.ManagedTh

    End Sub

    Private Sub theLongRunningTask()
        Dim progressForm As New TaskProgress()
        progressForm.Show()
        ' Refresh causes an instant (non-posted) display of
        progressForm.Refresh()

        ' Slowly increment the progress bar.
        For i As Integer = 1 To 10
            progressForm.ProgressIndicator.Value += 10
```

There is no form_load procedure. You have to use the New(), unless you go and make a custom form_load procedure.



```
Option Strict On
Imports System.Threading
Imports System.Reflection

Public Class Form1

    Public Sub New()

        ' This call is required by the Windows Form Designer.
        InitializeComponent()

        ' Add any initialization after the InitializeComponent() call.

        MsgBox("form started!", MsgBoxStyle.Information)
    End Sub
```

3.2 Multi-thread:

If you do not use multi-thread, the form may not have enough time to update the GUI in keeping up with the process. The process will finish long before the GUI update can finish.

http://rapidshare.com/files/304856823/PorgressBar-Sample-1.zip

3.3 VBModal

To show form modally ?

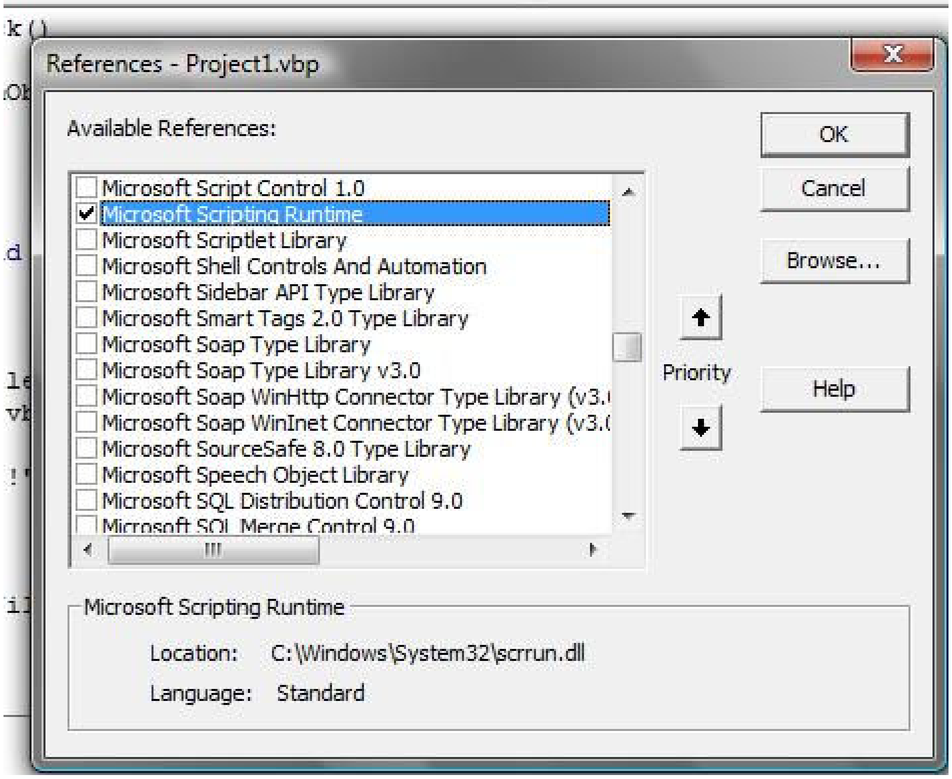
http://www.experts-exchange.com/Programming/Languages/.NET/Visual_Basic.NET/Q_22666209.html

Use ‘showDialog’ instead of ‘show’ method.

4.0 FileSystemObject & File

VB6:

Using ‘Microsoft Scripting Runtime’ component to get reference to ‘FileSystemObject’.



- □ Microsoft Script Control 1.0
- ☑ Microsoft Scripting Runtime
- □ Microsoft Scriptlet Library
- □ Microsoft Shell Controls And Automation
- □ Microsoft Sidebar API Type Library
- □ Microsoft Smart Tags 2.0 Type Library
- □ Microsoft Soap Type Library
- □ Microsoft Soap Type Library v3.0
- □ Microsoft Soap WinHttp Connector Type Library (v3.(
- □ Microsoft Soap WinInet Connector Type Library (v3.(
- □ Microsoft SourceSafe 8.0 Type Library
- □ Microsoft Speech Object Library
- □ Microsoft SQL Distribution Control 9.0
- □ Microsoft SQL Merge Control 9.0

Microsoft Scripting Runtime

Location: C:\Windows\System32\scrrun.dll

Language: Standard

Using ‘FileSystemObject’ for file operation.

Write Text To File:

```
'On Error GoTo m_err2:
Dim strm As TextStream
Dim fso As New FileSystemObject

Public Sub writeHisTransData(MyData As Variant)
Dim strm As TextStream
Dim fso As New FileSystemObject

With fso
'check if file exist
If .FileExists(CStr(App.Path & "\HistoryData\DataFile.txt")) Then
   'update log file
   Set strm = .OpenTextFile(CStr(App.Path & "\HistoryData\DataFile.txt"),
ForAppending, False)
   strm.Write (MyData)
Else
  'create new log
  Set strm = .CreateTextFile(CStr(App.Path & "\HistoryData\DataFile.txt"), True)
  strm.Write (MyData)
End If

End With

End Sub
```

Write Binary To File

```
Private Sub writeBinFile(MyData As Variant, Optional tFileHandle As String)

Dim nFileNum As Integer

'delete any existing file
'Kill App.Path & "\FingData.bin"
If FileExists(tFileHandle) Then Kill tFileHandle

nFileNum = FreeFile
' Open App.Path & "\FingData.bin" For Binary Access Write Lock Read Write As
#nFileNum
Open tFileHandle For Binary Access Write Lock Read Write As #nFileNum

Put #nFileNum, , MyData
Close #nFileNum

End Sub
```

VB.NET:

Need to ‘Imports System.IO'

Write Text To File:

```
Imports System.IO

Public Sub writeFileStrData(ByVal MyData As Object, ByVal filePath As
String, Optional ByVal transType As String = "", Optional ByVal
dataEncoding As String = "")

        Dim Str As String
        Dim fs As FileStream
        Dim tempBytes() As Byte

        tempBytes = Nothing

        If transType = "" Then
            transType = "Append" 'Set default
        End If

        If dataEncoding = "" Then
            dataEncoding = "ANSI"
        End If

        Try

            Str = CType(MyData, String)

            If dataEncoding = "ANSI" Then
                tempBytes = System.Text.Encoding.Default.GetBytes(Str)
            ElseIf dataEncoding = "Unicode" Then
                tempBytes = System.Text.Encoding.Unicode.GetBytes(Str)
            End If

            fs = New FileStream(filePath, FileMode.OpenOrCreate,
FileAccess.Write)
            If transType = "Append" Then
                fs.Seek(0, SeekOrigin.End)
            ElseIf transType = "Overwrite" Then
                fs.Seek(0, SeekOrigin.Begin)
            End If

            fs.Write(tempBytes, 0, tempBytes.Length)
            fs.Close()

        Catch ex As Exception

            MsgBox(ex.Message & vbCrLf & ex.StackTrace)
        End Try

    End Sub
```

For descriptions and sample code.

Refer to http://www.it-sideways.com/2010/02/write-text-to-file-write-binary-to-file.html

5.0 What is Protected Overrides ?

Protected:

This method is only visible to this class and derived classes!

Overrides:

Indicates that this Sub procedure overrides an identically named procedure in a base class

6.0 CallBack Function (Creating Custom Class and Custom Event Handler)

Download sample code.

http://rapidshare.com/files/216015564/MyService.rar

7.0 F8 – Step by Step Debugging

http://www.vbforums.com/showthread.php?referrerid=61394&t=506169

There is another option called 'step over' where you do not go nested into sub procedural calls, just focus on existing procedure.

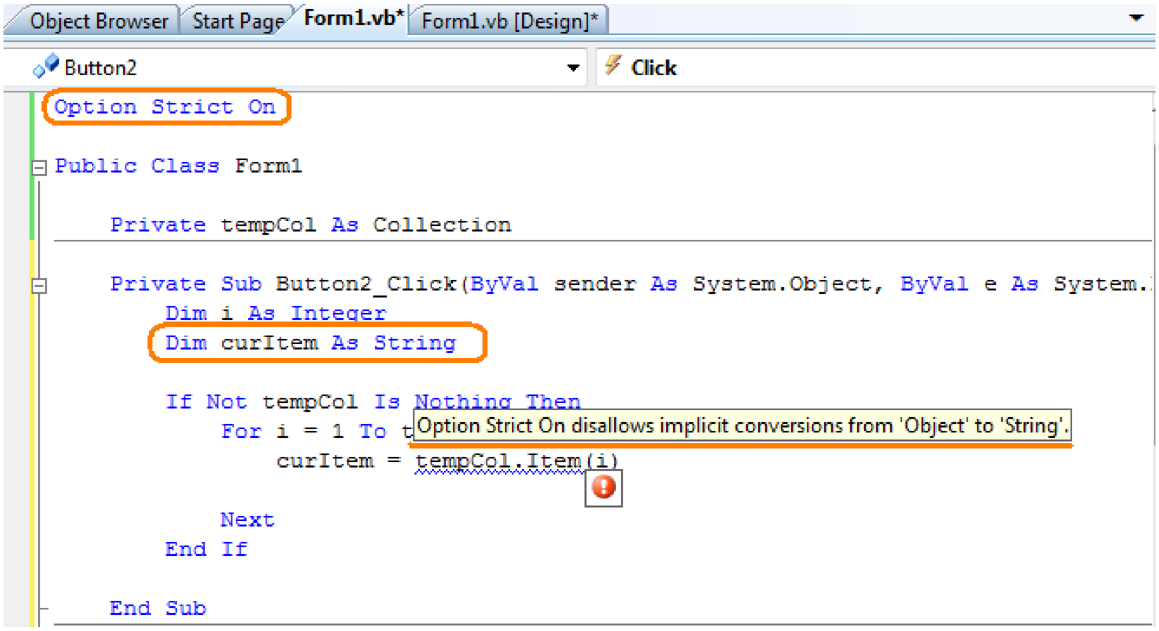
8.0 Collection

Not much big difference from VB6, but a few things to take note.

First of all, in VB6 and VB.NET, both will accept object as the item.

8.1 Option Strict On

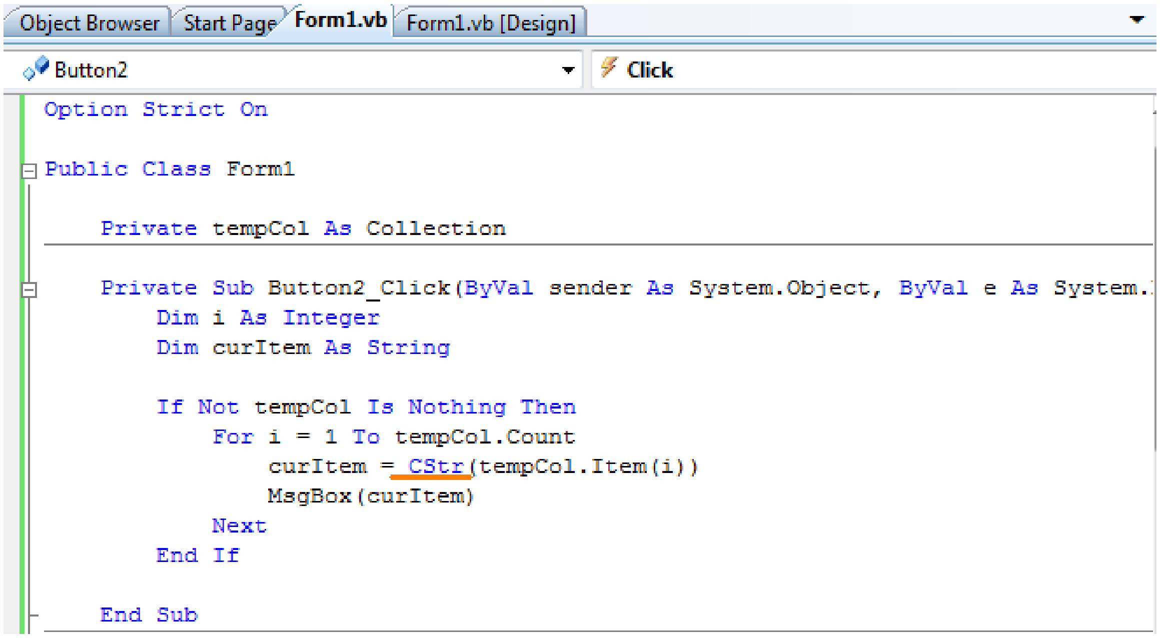
If you have declared 'Option Strict On', you have to use explicit conversion for any types of variable including native types such as string, integer and etc.



```
Option Strict On

Public Class Form1

    Private tempCol As Collection

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.
        Dim i As Integer
        Dim curItem As String

        If Not tempCol Is Nothing Then
            For i = 1 To t
                curItem = tempCol.Item(i)

            Next
        End If

    End Sub
```

Option Strict On disallows implicit conversions from 'Object' to 'String'.

→ Solution:

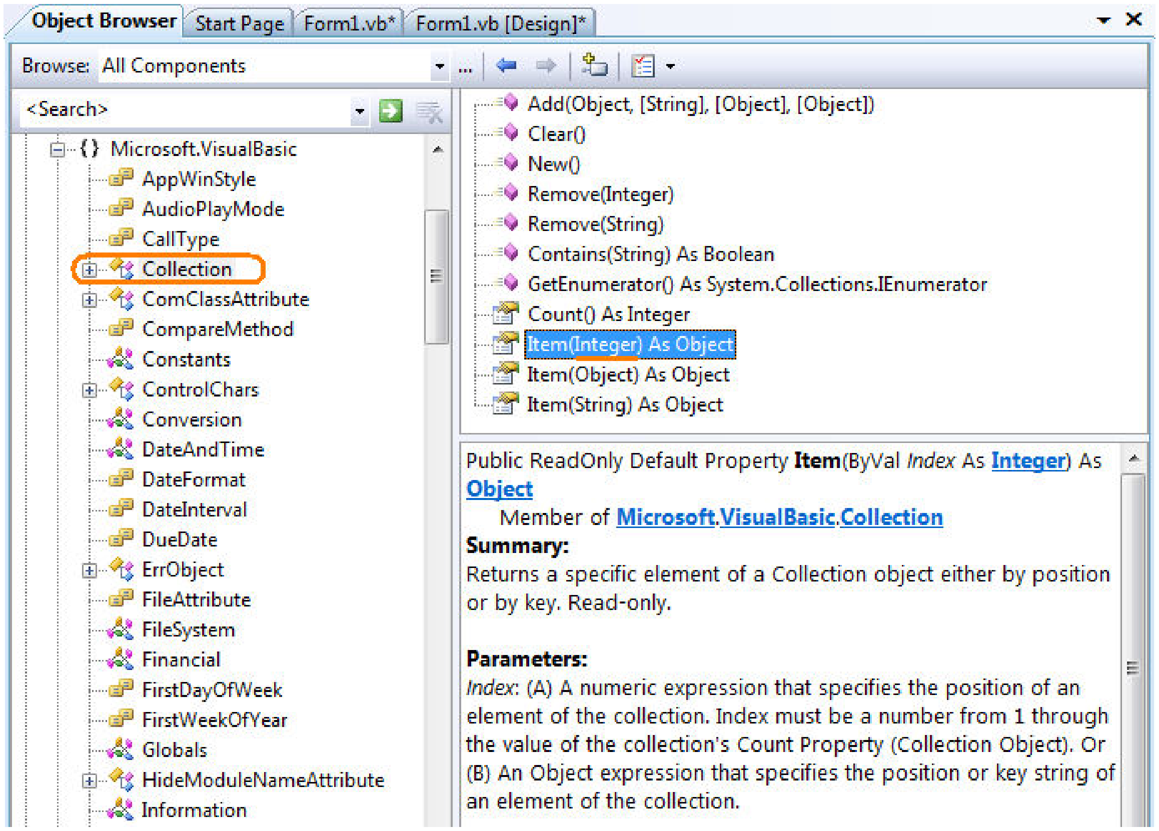


```
Option Strict On

Public Class Form1

    Private tempCol As Collection

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.
        Dim i As Integer
        Dim curItem As String

        If Not tempCol Is Nothing Then
            For i = 1 To tempCol.Count
                curItem = CStr(tempCol.Item(i))
                MsgBox(curItem)
            Next
        End If

    End Sub
```

8.2 1-Based Index

1-based Index (unlike ListView)

The index starts from 1. (This is actually the same as in VB6)



Browse: All Components

Public ReadOnly Default Property **Item**(ByVal *Index* As Integer) As Object
Member of Microsoft.VisualBasic.Collection

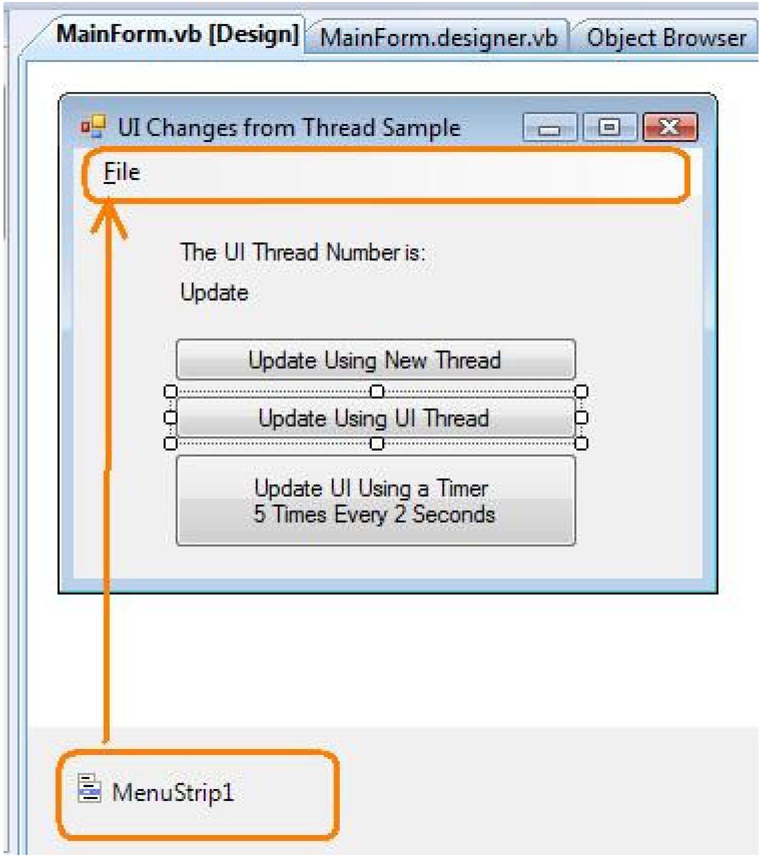
Summary:
Returns a specific element of a Collection object either by position or by key. Read-only.

Parameters:
Index: (A) A numeric expression that specifies the position of an element of the collection. Index must be a number from 1 through the value of the collection's Count Property (Collection Object). Or (B) An Object expression that specifies the position or key string of an element of the collection.

8.3 Index Type is Integer

Index type should be Integer, but long can work as well. However, the limit is still integer which is maximum = 2^31 = 2147483648

9.0 Menu Editor is called MenuStrip



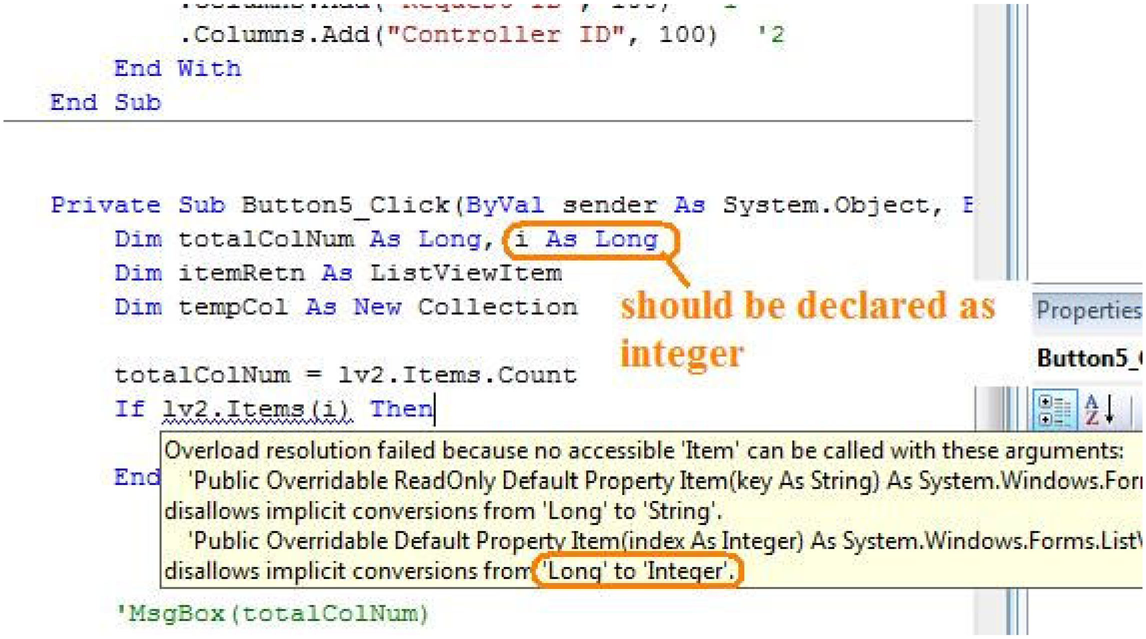
10.0 DoEvents

Use ‘Application.DoEvents()’

11.0 ListView

11.1 Integer Type:

First of all, the index must be of integer type.

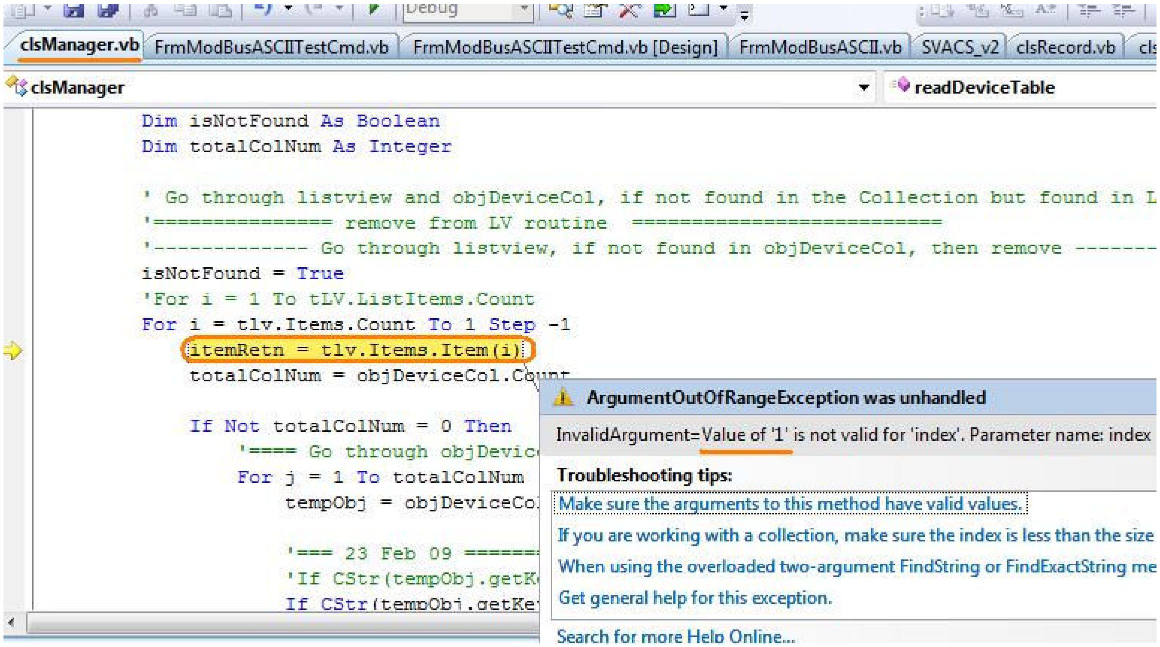


```
            .Columns.Add("Controller ID", 100)   '2
        End With
    End Sub


    Private Sub Button5_Click(ByVal sender As System.Object, E
        Dim totalColNum As Long, i As Long
        Dim itemRetn As ListViewItem
        Dim tempCol As New Collection

        totalColNum = lv2.Items.Count
        If lv2.Items(i) Then
        Overload resolution failed because no accessible 'Item' can be called with these arguments:
        End   'Public Overridable ReadOnly Default Property Item(key As String) As System.Windows.Forı
        disallows implicit conversions from 'Long' to 'String'.
          'Public Overridable Default Property Item(index As Integer) As System.Windows.Forms.List\
        disallows implicit conversions from 'Long' to 'Integer'.

        'MsgBox(totalColNum)
```

should be declared as integer

11.2 Zero-Based Index:

Secondly, index starts with 0 and not 1.



```
        Dim isNotFound As Boolean
        Dim totalColNum As Integer

        ' Go through listview and objDeviceCol, if not found in the Collection but found in L
        '=============== remove from LV routine  ==========================
        '------------- Go through listview, if not found in objDeviceCol, then remove -------
        isNotFound = True
        'For i = 1 To tLV.ListItems.Count
        For i = tlv.Items.Count To 1 Step -1
            itemRetn = tlv.Items.Item(i)
            totalColNum = objDeviceCol.Count

            If Not totalColNum = 0 Then
                '==== Go through objDevic
                For j = 1 To totalColNum
                    tempObj = objDeviceCo

                    '=== 23 Feb 09 ======
                    'If CStr(tempObj.getK
                    If CStr(tempObj.getKe
```

ArgumentOutOfRangeException was unhandled

InvalidArgument=Value of '1' is not valid for 'index'. Parameter name: index

Troubleshooting tips:

Make sure the arguments to this method have valid values.

If you are working with a collection, make sure the index is less than the size

When using the overloaded two-argument FindString or FindExactString me

Get general help for this exception.

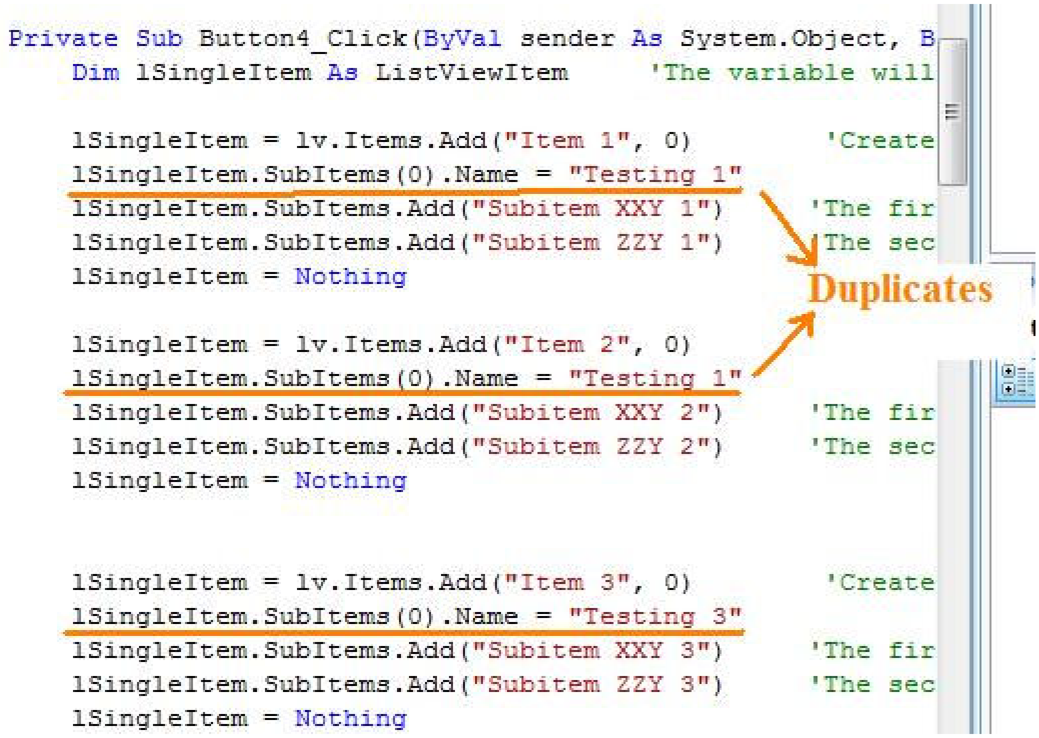
Search for more Help Online...

11.3 Item Key:

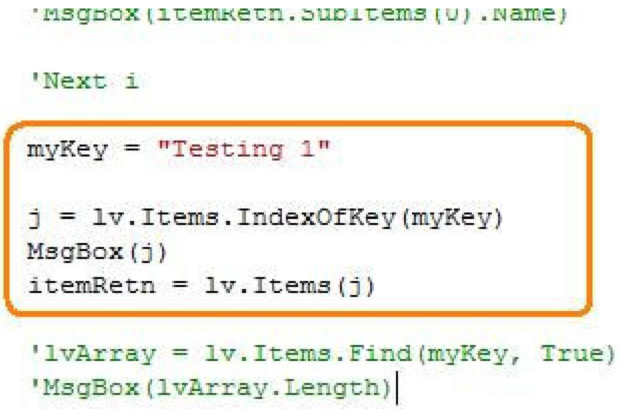
What about key ?

The key is actually the name property of each listviewitem's subitem. And each subitem can have a name.

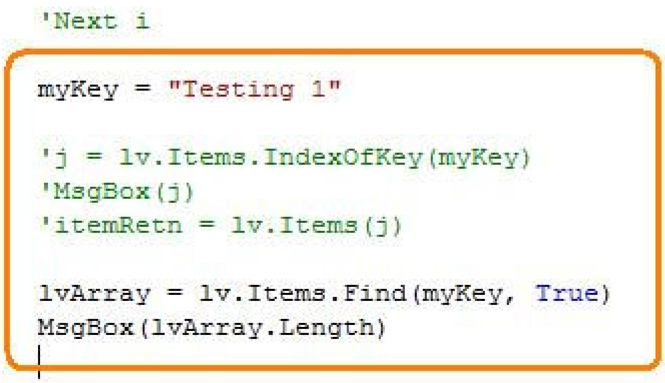
And you are able to find listviewitem by specifying either one of the name of the subitems by using listviewitem.indexofkey method. This is powerful. and if there are duplicate name, this method will just return with the first listviewitem. Otherwise, just use the the listviewitem.find method.



```
Private Sub Button4_Click(ByVal sender As System.Object, B
    Dim lSingleItem As ListViewItem     'The variable will

    lSingleItem = lv.Items.Add("Item 1", 0)          'Create
    lSingleItem.SubItems(0).Name = "Testing 1"
    lSingleItem.SubItems.Add("Subitem XXY 1")       'The fir
    lSingleItem.SubItems.Add("Subitem ZZY 1")       'The sec
    lSingleItem = Nothing

    lSingleItem = lv.Items.Add("Item 2", 0)
    lSingleItem.SubItems(0).Name = "Testing 1"
    lSingleItem.SubItems.Add("Subitem XXY 2")       'The fir
    lSingleItem.SubItems.Add("Subitem ZZY 2")       'The sec
    lSingleItem = Nothing


    lSingleItem = lv.Items.Add("Item 3", 0)          'Create
    lSingleItem.SubItems(0).Name = "Testing 3"
    lSingleItem.SubItems.Add("Subitem XXY 3")       'The fir
    lSingleItem.SubItems.Add("Subitem ZZY 3")       'The sec
    lSingleItem = Nothing
```



```
'MsgBox(itemRetn.SubItems(0).Name)

'Next i

myKey = "Testing 1"

j = lv.Items.IndexOfKey(myKey)
MsgBox(j)
itemRetn = lv.Items(j)

'lvArray = lv.Items.Find(myKey, True)
'MsgBox(lvArray.Length)
```

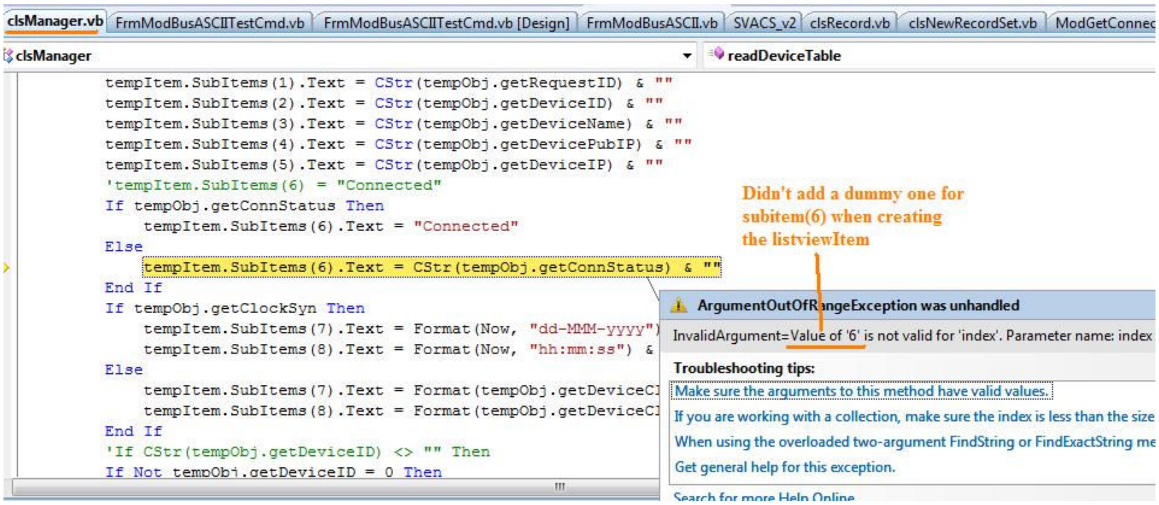


```
'Next i

myKey = "Testing 1"

'j = lv.Items.IndexOfKey(myKey)
'MsgBox(j)
'itemRetn = lv.Items(j)

lvArray = lv.Items.Find(myKey, True)
MsgBox(lvArray.Length)
```

11.4 ListItem

ListItem is now become ListViewItem

11.5 Add SubItem



```
        tempItem.SubItems(1).Text = CStr(tempObj.getRequestID) & ""
        tempItem.SubItems(2).Text = CStr(tempObj.getDeviceID) & ""
        tempItem.SubItems(3).Text = CStr(tempObj.getDeviceName) & ""
        tempItem.SubItems(4).Text = CStr(tempObj.getDevicePubIP) & ""
        tempItem.SubItems(5).Text = CStr(tempObj.getDeviceIP) & ""
        'tempItem.SubItems(6) = "Connected"
        If tempObj.getConnStatus Then
            tempItem.SubItems(6).Text = "Connected"
        Else
            tempItem.SubItems(6).Text = CStr(tempObj.getConnStatus) & ""
        End If
        If tempObj.getClockSyn Then
            tempItem.SubItems(7).Text = Format(Now, "dd-MMM-yyyy")
            tempItem.SubItems(8).Text = Format(Now, "hh:mm:ss") &
        Else
            tempItem.SubItems(7).Text = Format(tempObj.getDeviceCl
            tempItem.SubItems(8).Text = Format(tempObj.getDeviceCl
        End If
        'If CStr(tempObj.getDeviceID) <> "" Then
        If Not tempObj.getDeviceID = 0 Then
```

Didn't add a dummy one for subitem(6) when creating the listviewItem

ArgumentOutOfRangeException was unhandled

InvalidArgument=Value of '6' is not valid for 'index'. Parameter name: index

Troubleshooting tips:

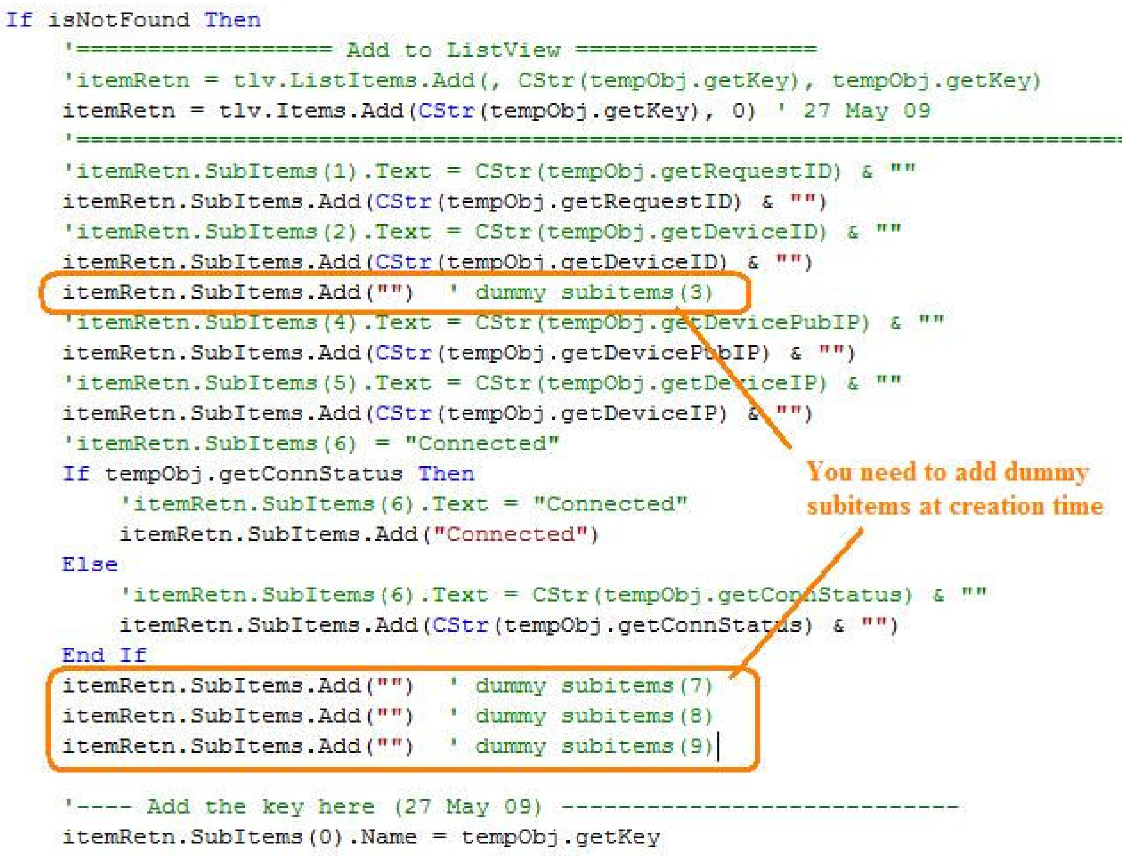
Make sure the arguments to this method have valid values.

If you are working with a collection, make sure the index is less than the size

When using the overloaded two-argument FindString or FindExactString me

Get general help for this exception.

Search for more Help Online



```
If isNotFound Then
    '=================== Add to ListView ==================
    'itemRetn = tlv.ListItems.Add(, CStr(tempObj.getKey), tempObj.getKey)
    itemRetn = tlv.Items.Add(CStr(tempObj.getKey), 0) ' 27 May 09
    '================================================================================
    'itemRetn.SubItems(1).Text = CStr(tempObj.getRequestID) & ""
    itemRetn.SubItems.Add(CStr(tempObj.getRequestID) & "")
    'itemRetn.SubItems(2).Text = CStr(tempObj.getDeviceID) & ""
    itemRetn.SubItems.Add(CStr(tempObj.getDeviceID) & "")
    itemRetn.SubItems.Add("")  ' dummy subitems(3)
    'itemRetn.SubItems(4).Text = CStr(tempObj.getDevicePubIP) & ""
    itemRetn.SubItems.Add(CStr(tempObj.getDevicePubIP) & "")
    'itemRetn.SubItems(5).Text = CStr(tempObj.getDeviceIP) & ""
    itemRetn.SubItems.Add(CStr(tempObj.getDeviceIP) & "")
    'itemRetn.SubItems(6) = "Connected"
    If tempObj.getConnStatus Then
        'itemRetn.SubItems(6).Text = "Connected"
        itemRetn.SubItems.Add("Connected")
    Else
        'itemRetn.SubItems(6).Text = CStr(tempObj.getConnStatus) & ""
        itemRetn.SubItems.Add(CStr(tempObj.getConnStatus) & "")
    End If
    itemRetn.SubItems.Add("")  ' dummy subitems(7)
    itemRetn.SubItems.Add("")  ' dummy subitems(8)
    itemRetn.SubItems.Add("")  ' dummy subitems(9)

    '---- Add the key here (27 May 09) ----------------------------
    itemRetn.SubItems(0).Name = tempObj.getKey
```

11.6 Column Sort

In VB6:

```
Private Sub lv_ColumnClick(ByVal ColumnHeader As MSComctlLib.ColumnHeader)
 '== 21 Feb 2009 ========================
 lv.Sorted = True 'enable
 lv.SortKey = ColumnHeader.Index - 1
 If lv.SortOrder = lvwAscending Then
   lv.SortOrder = lvwDescending
 Else
   lv.SortOrder = lvwAscending
 End If
 lv.Refresh
 '=====================================
End Sub
```

In VB.NET:

http://www.vb-helper.com/howto_net_listview_sort_clicked_column.html http://support.microsoft.com/default.aspx?scid=kb;EN-US;Q319401

http://www.fryan0911.com/2009/05/vbnet-how-to-sort-listview-by-clicked.html

http://www.devcity.net/Articles/20/1/20020304.aspx

```
Private Sub lv_ColumnClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.ColumnClickEventArgs) Handles lv.ColumnClick

        Dim newSortOrder As System.Windows.Forms.SortOrder

        If MyLVSortOrder = SortOrder.Ascending Then
            newSortOrder = SortOrder.Descending
            MyLVSortOrder = SortOrder.Descending
        Else
            newSortOrder = SortOrder.Ascending
            MyLVSortOrder = SortOrder.Ascending
        End If

        ' Create a comparer.
        lv.ListViewItemSorter = New clsListviewSorter(e.Column,
newSortOrder)
        ' Sort.
        lv.Sort()

    End Sub
```

Basically, the idea is to create an instance of ' System.Windows.Forms.SortOrder' and then assign it to the listview's property of 'listviewItemSorter'.

To do that listviewItemSorter must be referenced to an object which implements 'system.collections.iComparer' and with two paramters.

1. Clicked column
2. Desired sort order

Refer to the source code from…

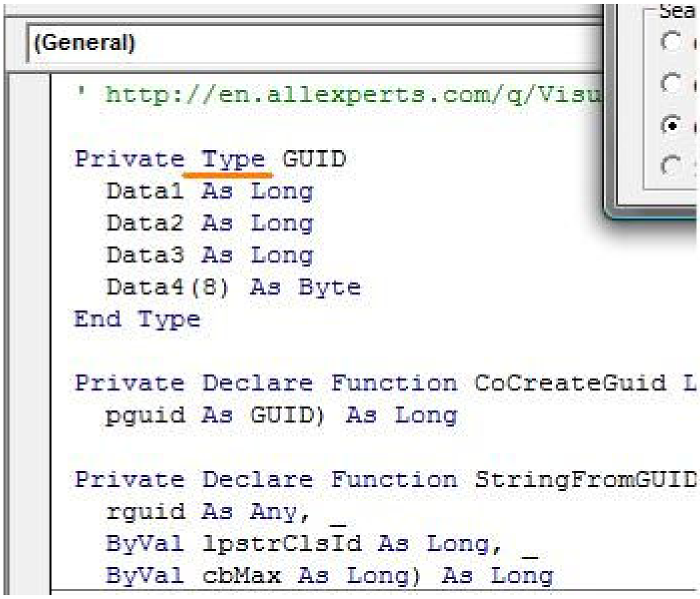
http://rapidshare.com/files/242215711/VB2008-Listview-Column-Sort.rar

11.7 ListViewItem Slow Loading:

http://www.it-sideways.com/2009/10/vbnet-listview-slow.html

12.0 Type:

'Type' is not longer supported in VB.NET



```
' http://en.allexperts.com/q/Visu

Private Type GUID
  Data1 As Long
  Data2 As Long
  Data3 As Long
  Data4(8) As Byte
End Type

Private Declare Function CoCreateGuid L
  pguid As GUID) As Long

Private Declare Function StringFromGUID
  rguid As Any, _
  ByVal lpstrClsId As Long, _
  ByVal cbMax As Long) As Long
```

Use ‘structure’

13.0 As Any

http://social.msdn.microsoft.com/Forums/en-US/vbinterop/thread/2c87df7c-48ad-4ab8-a6b8-e0e277eb74ec

Use ‘as object’

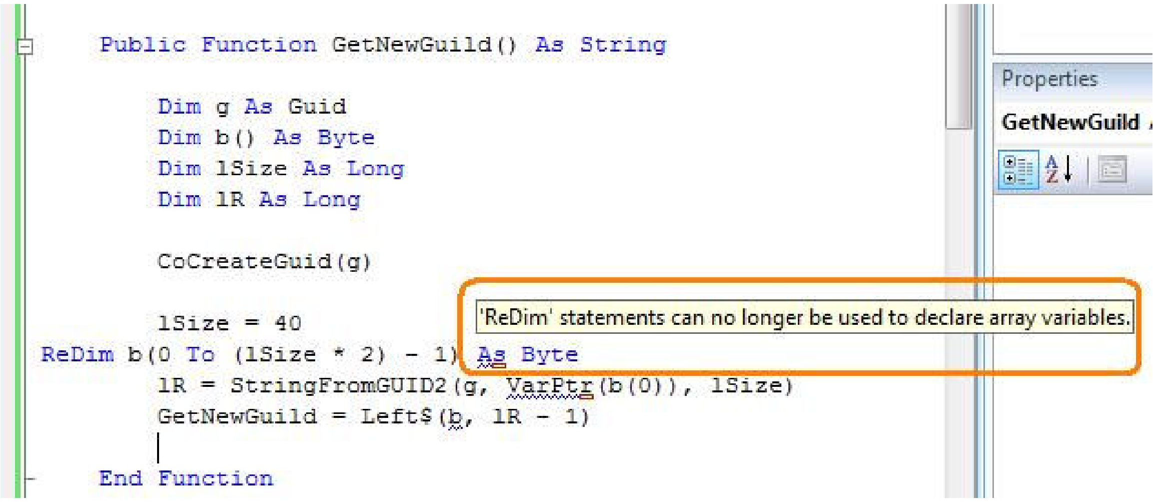
14.0 VarPtr

http://dotnet-snippets.com/dns/varptr-for-net-SID585.aspx
http://bytes.com/groups/net-vb/816523-coming-vb6-net-varptr-not-supported
http://forums.devx.com/archive/index.php/t-134177.html
http://social.msdn.microsoft.com/Forums/en-US/vbgeneral/thread/ab109cb7-4288-4e04-bc17-8e343d0db43b

But make sure you ‘Imports System.Runtime.InteropServices’

15.0 Redim

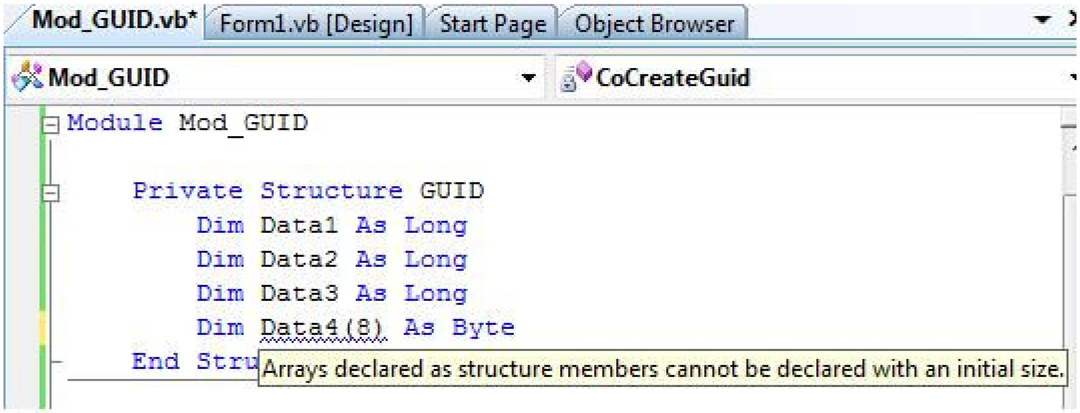
15.1 In VB.NET, you cannot use Redim statement to declare variable.



```
    Public Function GetNewGuild() As String

        Dim g As Guid
        Dim b() As Byte
        Dim lSize As Long
        Dim lR As Long

        CoCreateGuid(g)

        lSize = 40
ReDim b(0 To (lSize * 2) - 1) As Byte
        lR = StringFromGUID2(g, VarPtr(b(0)), lSize)
        GetNewGuild = Left$(b, lR - 1)

    End Function
```

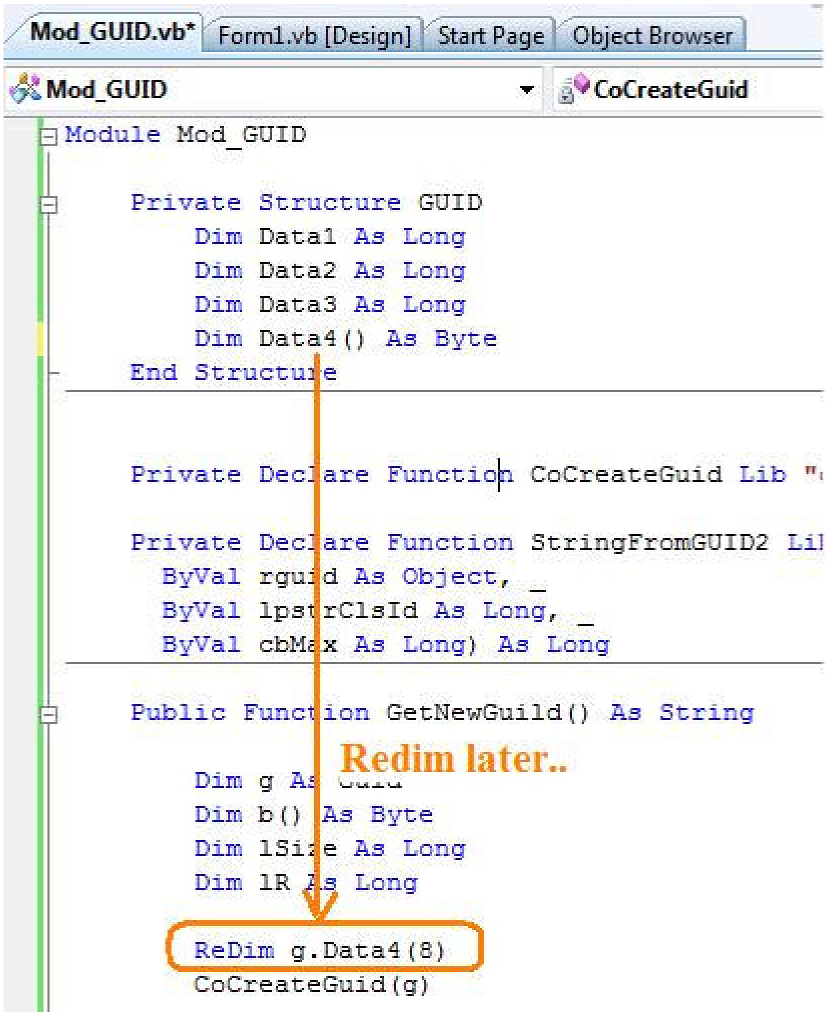
'ReDim' statements can no longer be used to declare array variables.

15.2 Use Redim to complement array declaration



```
Module Mod_GUID

    Private Structure GUID
        Dim Data1 As Long
        Dim Data2 As Long
        Dim Data3 As Long
        Dim Data4(8) As Byte
    End Stru
```

Arrays declared as structure members cannot be declared with an initial size.



```
Module Mod_GUID

    Private Structure GUID
        Dim Data1 As Long
        Dim Data2 As Long
        Dim Data3 As Long
        Dim Data4() As Byte
    End Structure


    Private Declare Function CoCreateGuid Lib "

    Private Declare Function StringFromGUID2 Li
      ByVal rguid As Object, _
      ByVal lpstrClsId As Long, _
      ByVal cbMax As Long) As Long

    Public Function GetNewGuild() As String

        Dim g As Guid
        Dim b() As Byte
        Dim lSize As Long
        Dim lR As Long

        ReDim g.Data4(8)
        CoCreateGuid(g)
```

16.0 Byte to String and vice versa

First of all, it is important to understand two things.

1. The difference between string and byte (binary)
2. The difference between ANSI and Unicode.

Refer to the following article on 'ANSI vs Unicode'.

http://www.it-sideways.com/2010/01/ansi-vs-unicode.html

Thus, when converting byte to string, you need to know the byte is in what encoding (ANSI, Unicode and etc) before using the following to convert.

```
Dim Str As String
Dim tempBytes() As Byte
```

```
Str  = System.Text.Encoding.Default.GetStringtempBytes
```

When converting string to bytes, you need to know what encoding you wanted for the bytes.

```
Dim Str As String
Dim tempBytes() As Byte

Str = "Hello"
tempBytes = System.Text.Encoding.Default.GetBytes(Str) ‘ Using default encoding
```

17.0 Generate GUID

VB6

```
' http://en.allexperts.com/q/Visual-Basic-1048/guid.htm

Private Type GUID
  Data1 As Long
  Data2 As Long
  Data3 As Long
  Data4(8) As Byte
End Type

Private Declare Function CoCreateGuid Lib "ole32.dll" ( _
  pguid As GUID) As Long

Private Declare Function StringFromGUID2 Lib "ole32.dll" ( _
  rguid As Any, _
  ByVal lpstrClsId As Long, _
  ByVal cbMax As Long) As Long

Public Function GetNewGuild() As String

Dim g As GUID
Dim b() As Byte
Dim lSize As Long
Dim lR As Long

CoCreateGuid g

lSize = 40
ReDim b(0 To (lSize * 2) - 1) As Byte
lR = StringFromGUID2(g, VarPtr(b(0)), lSize)
GetNewGuild = Left$(b, lR - 1)
```

```
End Function
```

VB.NET

http://www.a1vbcode.com/vbtip-145.asp

18.0 Variant

‘Variant’ is now ‘object’.

‘Variant’ is a reference-type variable.

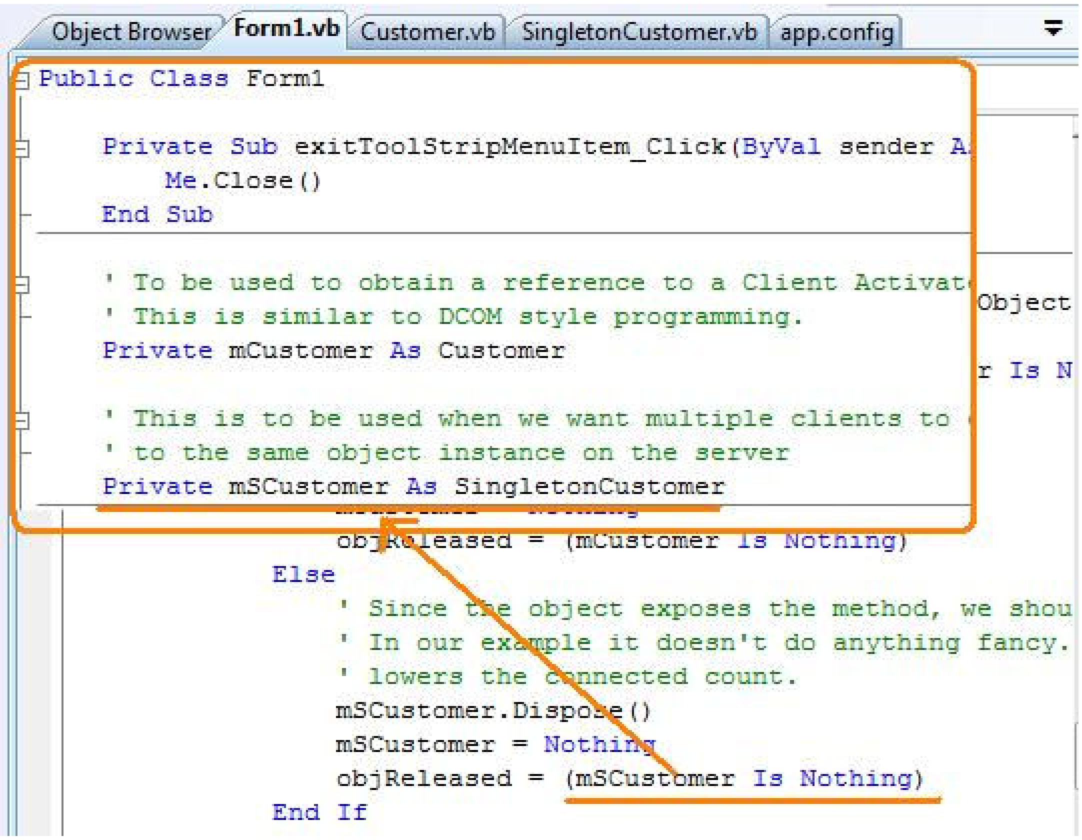
‘object’ is a value-type variable

And this can cause a big world of difference.

For instance, when accepting a date object as ‘variant’ and then reformatting it will be different from accepting a date object as ‘object’ and reformatting it.

19.0 IsNull()

‘Isnull’ now becomes ‘is nothing’



```
Public Class Form1

    Private Sub exitToolStripMenuItem_Click(ByVal sender A
        Me.Close()
    End Sub

    ' To be used to obtain a reference to a Client Activat
    ' This is similar to DCOM style programming.
    Private mCustomer As Customer

    ' This is to be used when we want multiple clients to
    ' to the same object instance on the server
    Private mSCustomer As SingletonCustomer
```

```
                                                          Object

                                                          r Is N

                objReleased = (mCustomer Is Nothing)
            Else
                ' Since the object exposes the method, we shou
                ' In our example it doesn't do anything fancy.
                ' lowers the connected count.
                mSCustomer.Dispose()
                mSCustomer = Nothing
                objReleased = (mSCustomer Is Nothing)
            End If
```

Or ‘isDBNull()’ (which is meant for database column)

http://msdn.microsoft.com/en-us/library/system.dbnull.aspx

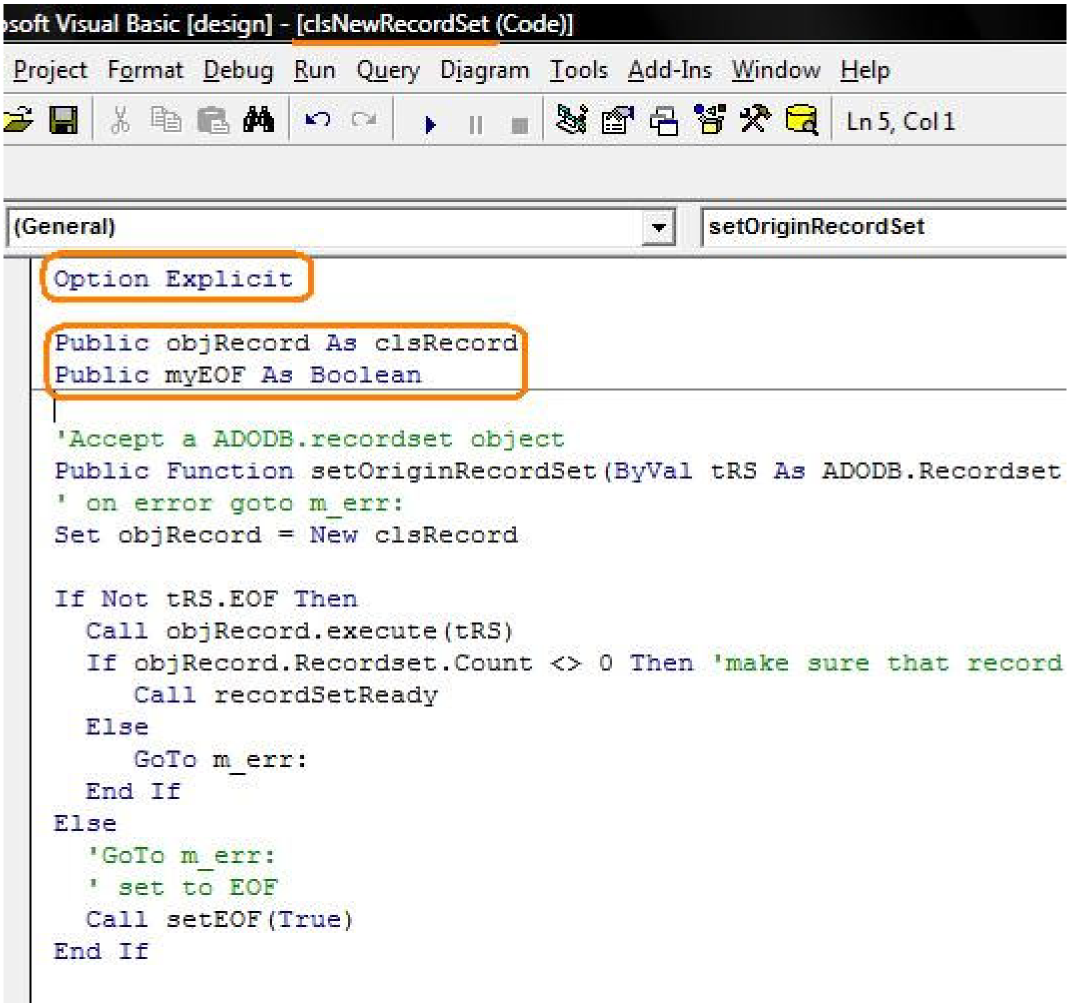
http://www.netcoole.com/asp2aspx/vbhtml/vbfuncs.htm

20.0 Ccur

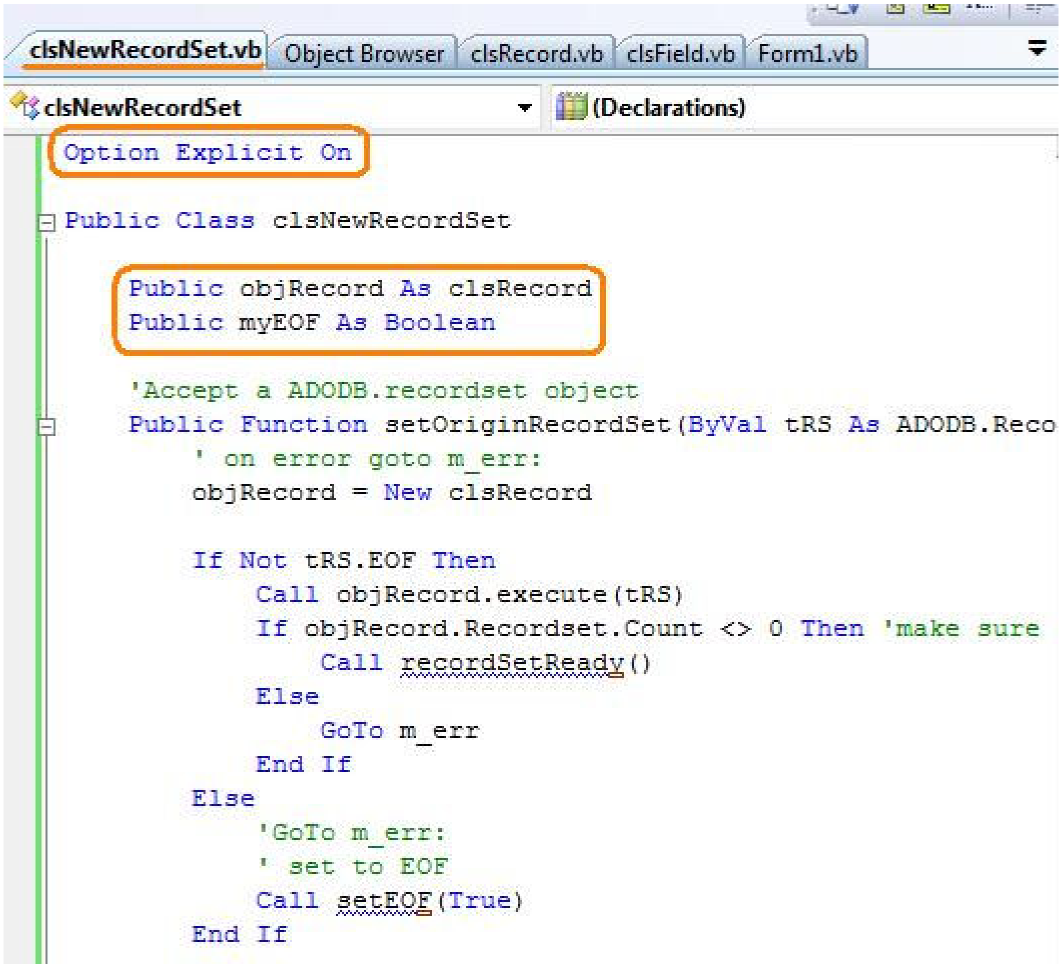
‘Ccur’ is not ‘cdec’

http://www.netcoole.com/asp2aspx/vbhtml/vbfuncs.htm

21.0 Basic Class Structure:


osoft Visual Basic [design] - [clsNewRecordSet (Code)]
Project Format Debug Run Query Diagram Tools Add-Ins Window Help
Ln 5, Col 1
(General)
setOriginRecordSet

```
Option Explicit

Public objRecord As clsRecord
Public myEOF As Boolean

'Accept a ADODB.recordset object
Public Function setOriginRecordSet(ByVal tRS As ADODB.Recordset
' on error goto m_err:
Set objRecord = New clsRecord

If Not tRS.EOF Then
  Call objRecord.execute(tRS)
  If objRecord.Recordset.Count <> 0 Then 'make sure that record
     Call recordSetReady
  Else
     GoTo m_err:
  End If
Else
  'GoTo m_err:
  ' set to EOF
  Call setEOF(True)
End If
```

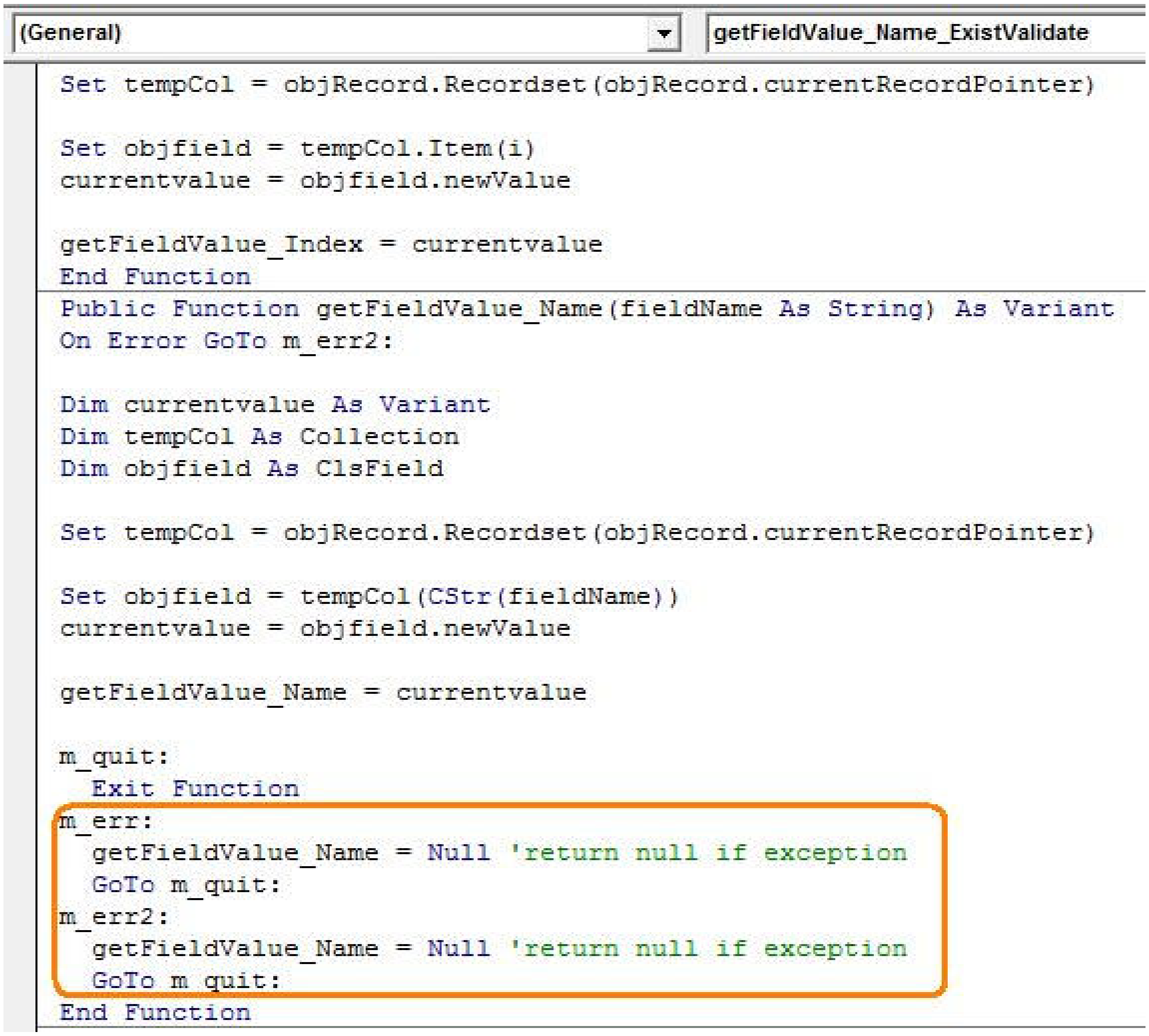


```
Option Explicit On

Public Class clsNewRecordSet

    Public objRecord As clsRecord
    Public myEOF As Boolean

    'Accept a ADODB.recordset object
    Public Function setOriginRecordSet(ByVal tRS As ADODB.Reco
        ' on error goto m_err:
        objRecord = New clsRecord

        If Not tRS.EOF Then
            Call objRecord.execute(tRS)
            If objRecord.Recordset.Count <> 0 Then 'make sure
                Call recordSetReady()
            Else
                GoTo m_err
            End If
        Else
            'GoTo m_err:
            ' set to EOF
            Call setEOF(True)
        End If
```

22.0 Null:

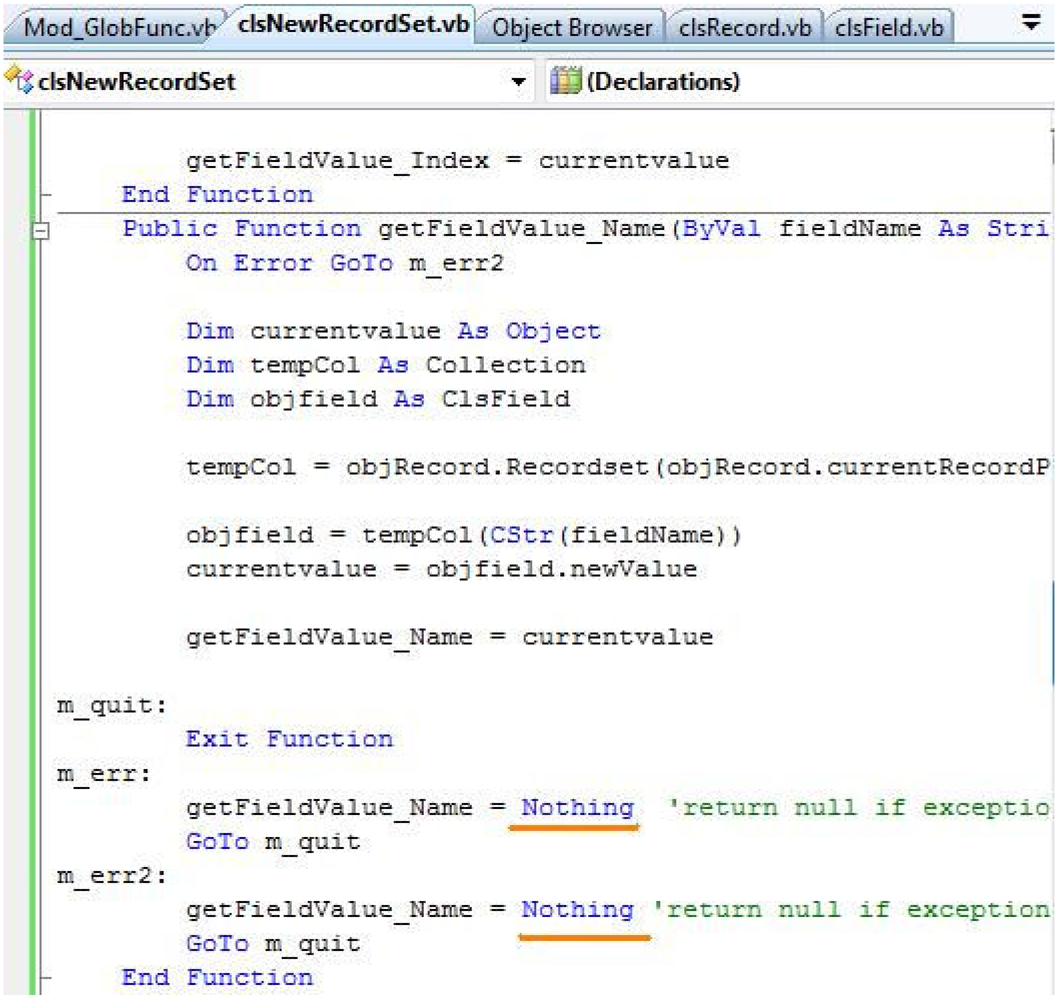
'Null' is now becomes 'Nothing'

VB6:



```
Set tempCol = objRecord.Recordset(objRecord.currentRecordPointer)

Set objfield = tempCol.Item(i)
currentvalue = objfield.newValue

getFieldValue_Index = currentvalue
End Function
Public Function getFieldValue_Name(fieldName As String) As Variant
On Error GoTo m_err2:

Dim currentvalue As Variant
Dim tempCol As Collection
Dim objfield As ClsField

Set tempCol = objRecord.Recordset(objRecord.currentRecordPointer)

Set objfield = tempCol(CStr(fieldName))
currentvalue = objfield.newValue

getFieldValue_Name = currentvalue

m_quit:
  Exit Function
m_err:
  getFieldValue_Name = Null 'return null if exception
  GoTo m_quit:
m_err2:
  getFieldValue_Name = Null 'return null if exception
  GoTo m_quit:
End Function
```

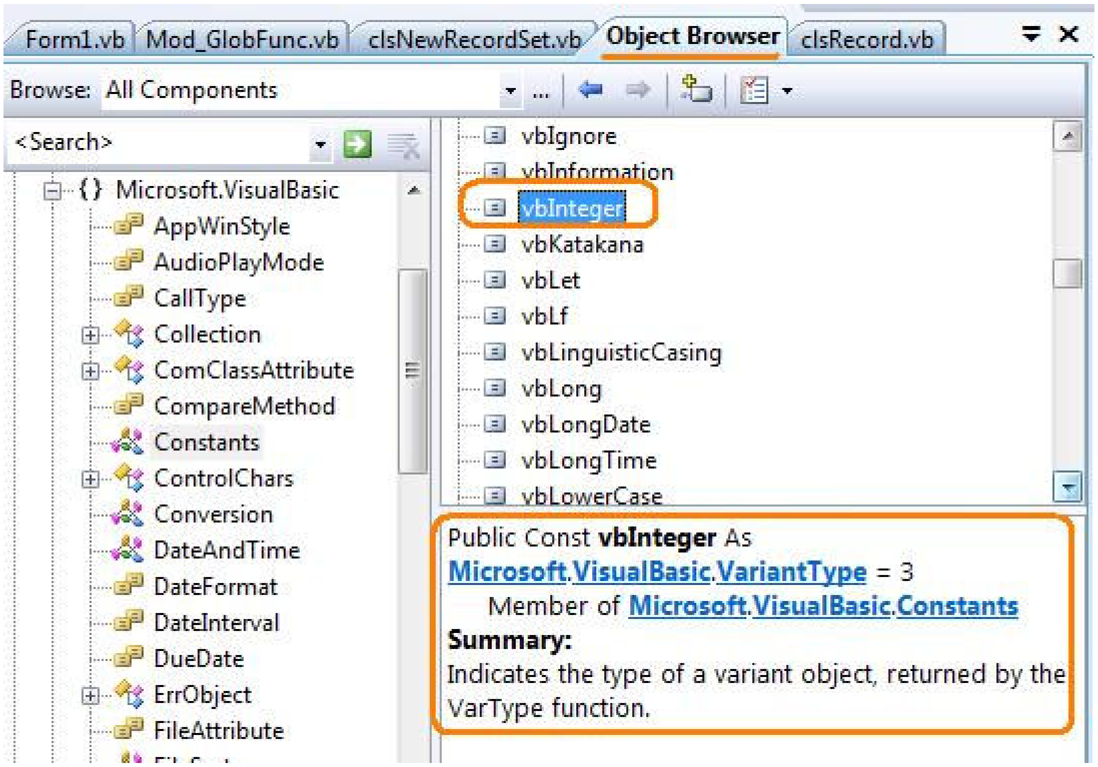
VB.NET



```
        getFieldValue_Index = currentvalue
    End Function
    Public Function getFieldValue_Name(ByVal fieldName As Stri
        On Error GoTo m_err2

        Dim currentvalue As Object
        Dim tempCol As Collection
        Dim objfield As ClsField

        tempCol = objRecord.Recordset(objRecord.currentRecordP

        objfield = tempCol(CStr(fieldName))
        currentvalue = objfield.newValue

        getFieldValue_Name = currentvalue

m_quit:
        Exit Function
m_err:
        getFieldValue_Name = Nothing  'return null if exceptio
        GoTo m_quit
m_err2:
        getFieldValue_Name = Nothing 'return null if exception
        GoTo m_quit
    End Function
```

23.0 VBVarType:

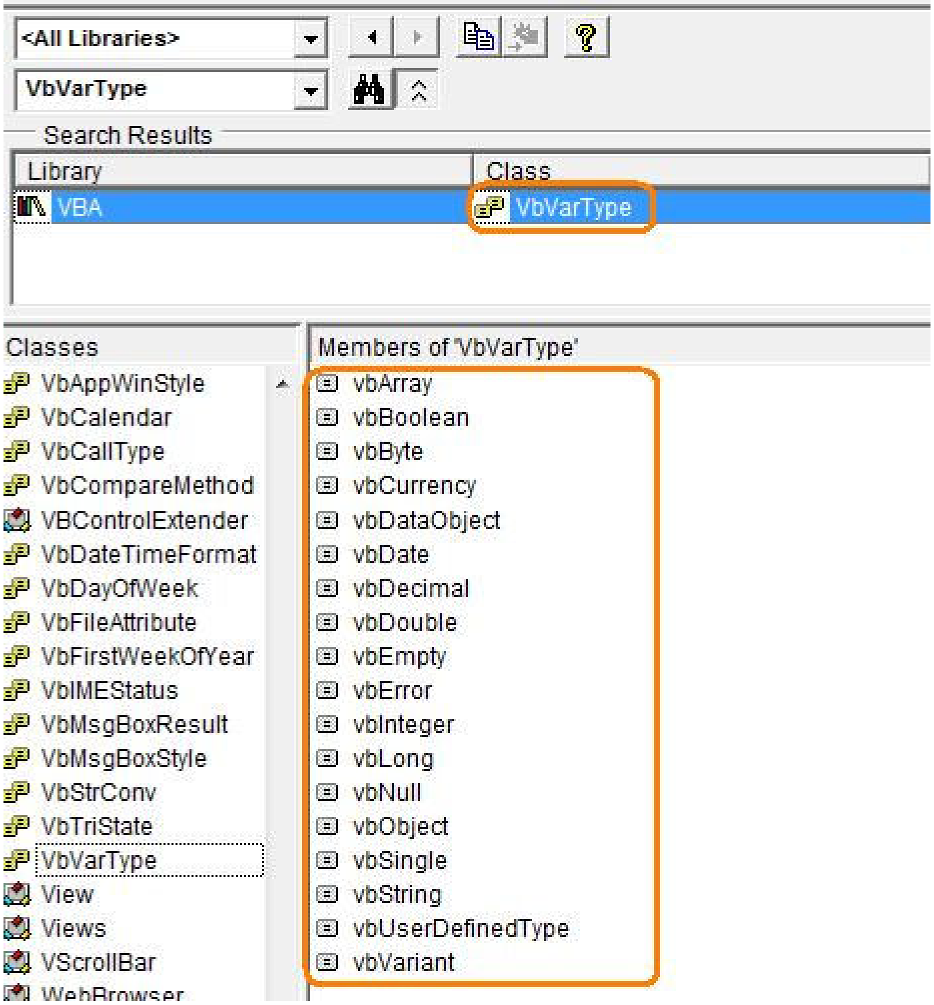
‘VBVarType’ now becomes 'VariantType'



Public Const **vbInteger** As
Microsoft.VisualBasic.VariantType = 3
Member of Microsoft.VisualBasic.Constants

Summary:

Indicates the type of a variant object, returned by the VarType function.



<All Libraries>

VbVarType

Search Results

Library	Class
VBA	VbVarType

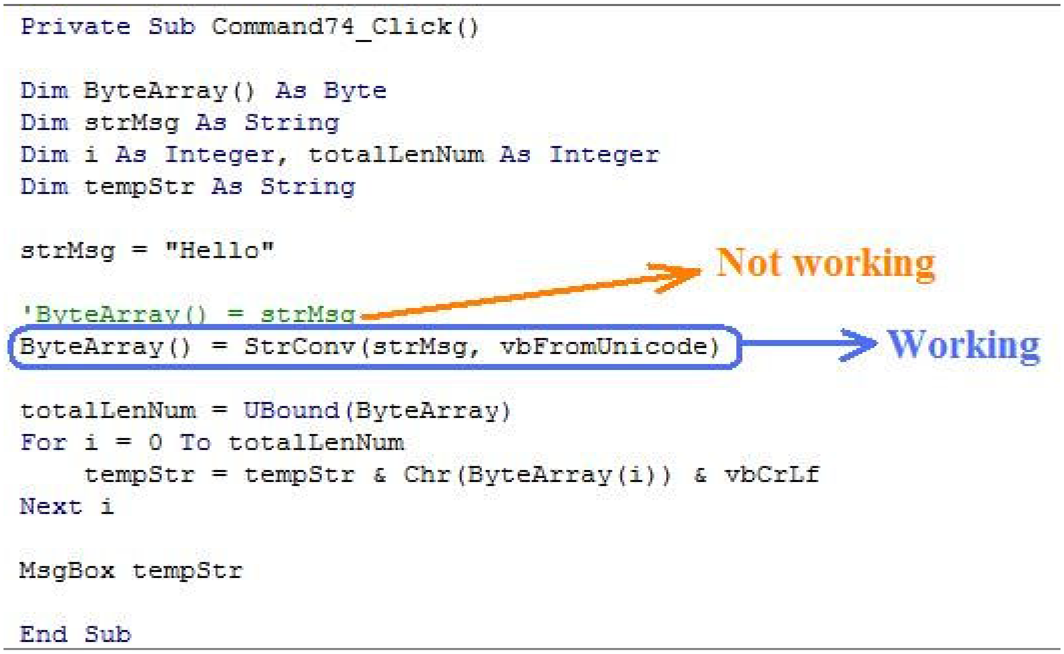
24.0 Control Array

There is no more control array in VB.NET

http://visualbasic.about.com/od/usingvbnet/l/bldykctrlarraya.htm

25.0 VBFromUnicode

In VB 6



```
Private Sub Command74_Click()

Dim ByteArray() As Byte
Dim strMsg As String
Dim i As Integer, totalLenNum As Integer
Dim tempStr As String

strMsg = "Hello"

'ByteArray() = strMsg
ByteArray() = StrConv(strMsg, vbFromUnicode)

totalLenNum = UBound(ByteArray)
For i = 0 To totalLenNum
    tempStr = tempStr & Chr(ByteArray(i)) & vbCrLf
Next i

MsgBox tempStr

End Sub
```

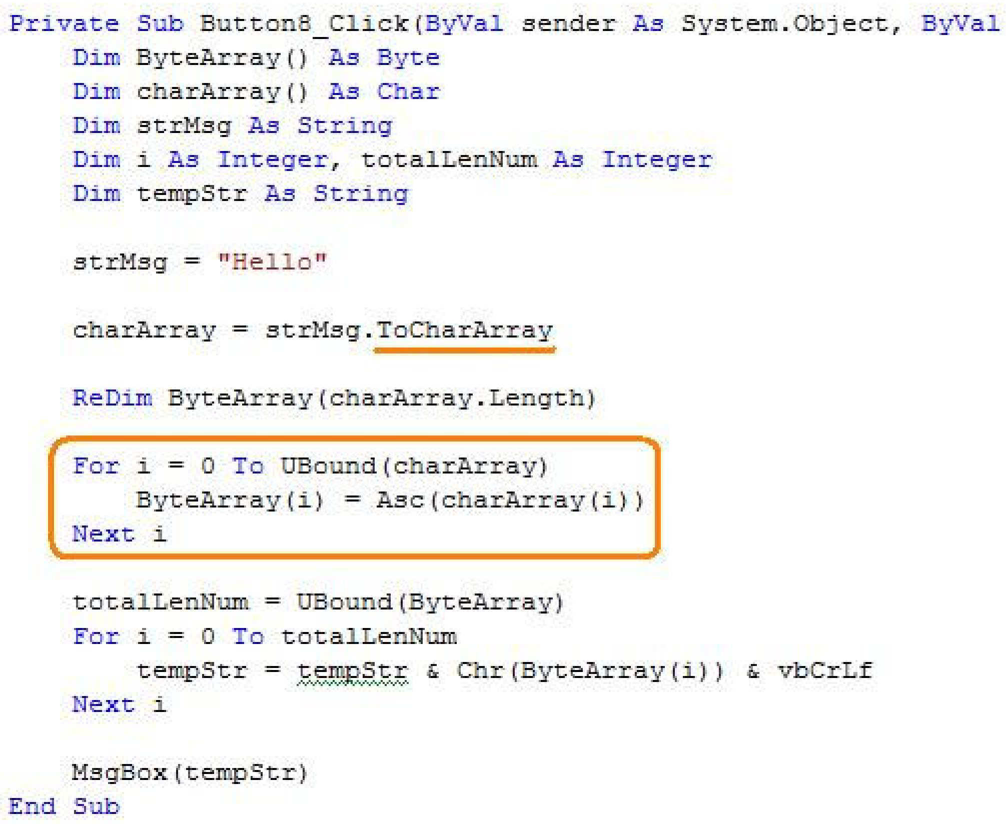
http://social.msdn.microsoft.com/Forums/en-US/vbgeneral/thread/f8bd2f71-9753-4f08-b14c-49c534f76d97

No need to convert, because VB.NET string is in Unicode. This is the same in VB6.

http://www.example-code.com/charset101_4.asp

But you still want to convert into a byte array.

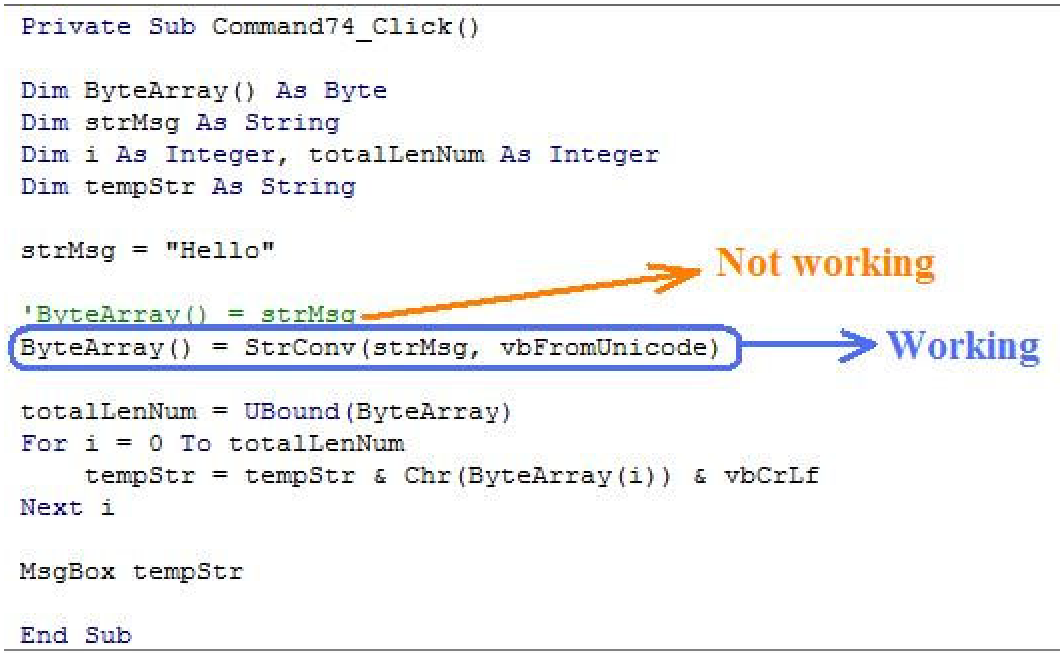
See the example below



```
Private Sub Button8_Click(ByVal sender As System.Object, ByVal
    Dim ByteArray() As Byte
    Dim charArray() As Char
    Dim strMsg As String
    Dim i As Integer, totalLenNum As Integer
    Dim tempStr As String

    strMsg = "Hello"

    charArray = strMsg.ToCharArray

    ReDim ByteArray(charArray.Length)

    For i = 0 To UBound(charArray)
        ByteArray(i) = Asc(charArray(i))
    Next i

    totalLenNum = UBound(ByteArray)
    For i = 0 To totalLenNum
        tempStr = tempStr & Chr(ByteArray(i)) & vbCrLf
    Next i

    MsgBox(tempStr)
End Sub
```

26.0 Creating an array out of a string

VB6

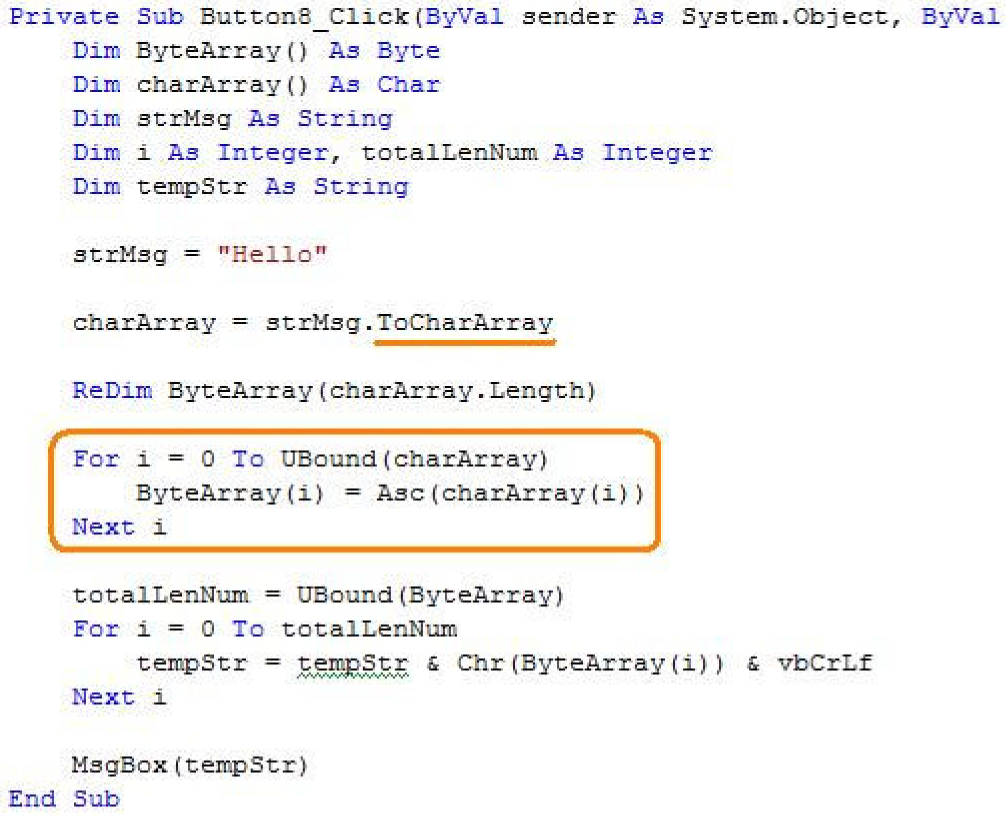


```
Private Sub Command74_Click()

Dim ByteArray() As Byte
Dim strMsg As String
Dim i As Integer, totalLenNum As Integer
Dim tempStr As String

strMsg = "Hello"

'ByteArray() = strMsg
ByteArray() = StrConv(strMsg, vbFromUnicode)

totalLenNum = UBound(ByteArray)
For i = 0 To totalLenNum
    tempStr = tempStr & Chr(ByteArray(i)) & vbCrLf
Next i

MsgBox tempStr

End Sub
```

Not working

Working

VB.NET

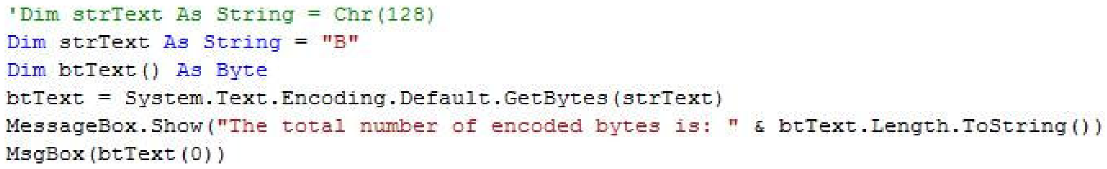


```
Private Sub Button8_Click(ByVal sender As System.Object, ByVal
    Dim ByteArray() As Byte
    Dim charArray() As Char
    Dim strMsg As String
    Dim i As Integer, totalLenNum As Integer
    Dim tempStr As String

    strMsg = "Hello"

    charArray = strMsg.ToCharArray

    ReDim ByteArray(charArray.Length)

    For i = 0 To UBound(charArray)
        ByteArray(i) = Asc(charArray(i))
    Next i

    totalLenNum = UBound(ByteArray)
    For i = 0 To totalLenNum
        tempStr = tempStr & Chr(ByteArray(i)) & vbCrLf
    Next i

    MsgBox(tempStr)
End Sub
```

Take note that you can only assign a value to a byte; a byte is a value.

Remember that a byte which represent a number is the number itself. Means that a byte which is = 40 is not the numerical 40, it is binary 40 (or 00101000 in binary) and it is representing “B” in ANSI. Whereas for the numerical 1, the byte is actually “40”

Alternatively…



```
'Dim strText As String = Chr(128)
Dim strText As String = "B"
Dim btText() As Byte
btText = System.Text.Encoding.Default.GetBytes(strText)
MessageBox.Show("The total number of encoded bytes is: " & btText.Length.ToString())
MsgBox(btText(0))
```

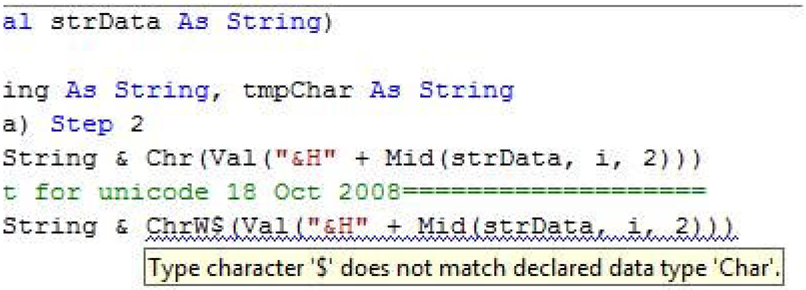
Take note that in VB6, you have to use () for the array assigning.

ByteArray() = …

However, in VB.NET you need to drop the usage of () during array assigning charArray = …

The parentheses is only used during declaration..

27.0 ChrW$



```
al strData As String)

ing As String, tmpChar As String
a) Step 2
String & Chr(Val("&H" + Mid(strData, i, 2)))
t for unicode 18 Oct 2008====================
String & ChrW$(Val("&H" + Mid(strData, i, 2)))
```

Type character '$' does not match declared data type 'Char'.

ChrW$ is no longer exist.

Use ‘ChrW’

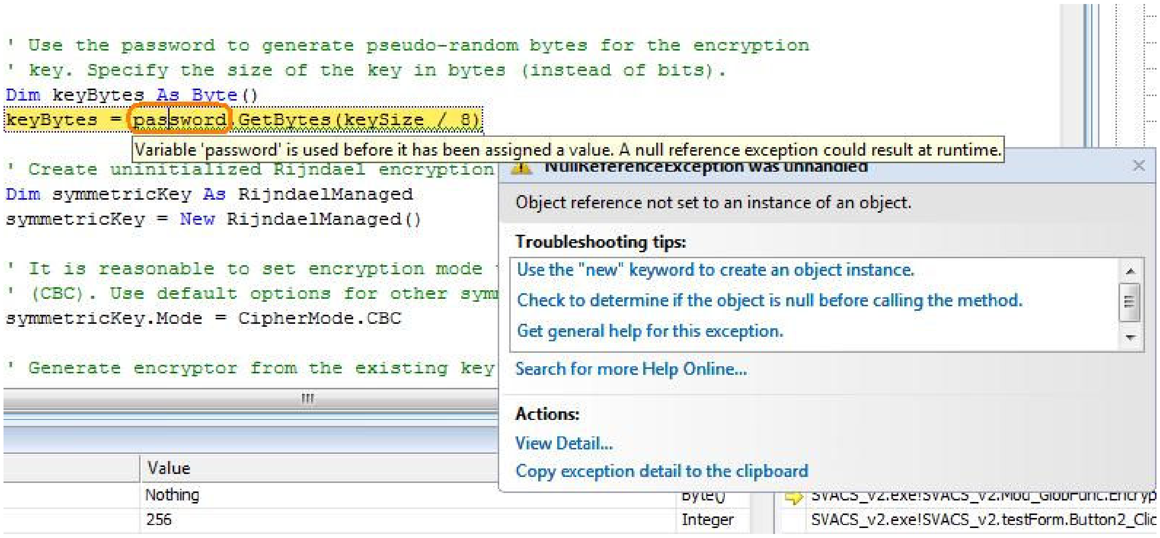
28.0 Unload (Me)

Becomes ‘me.dispose()’ or ‘me.close()’

29.0 Object Null Exception

In VB6, it doesn’t inform you which object is ‘null’ or ‘nothing’ in the line which triggered ‘Object reference is not set to an instance of an object’

However, in VB.NET, it does.



```
' Use the password to generate pseudo-random bytes for the encryption
' key. Specify the size of the key in bytes (instead of bits).
Dim keyBytes As Byte()
keyBytes = password.GetBytes(keySize / 8)

' Create uninitialized Rijndael encryption
Dim symmetricKey As RijndaelManaged
symmetricKey = New RijndaelManaged()

' It is reasonable to set encryption mode
' (CBC). Use default options for other sym
symmetricKey.Mode = CipherMode.CBC

' Generate encryptor from the existing key
```

Variable 'password' is used before it has been assigned a value. A null reference exception could result at runtime.

NullReferenceException was unhandled

Object reference not set to an instance of an object.

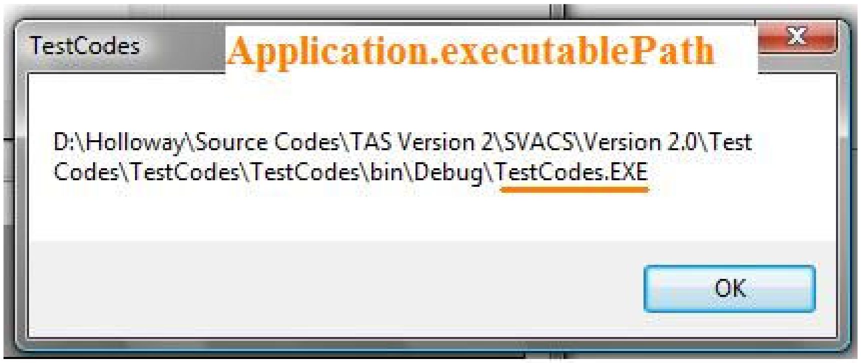
Troubleshooting tips:
Use the "new" keyword to create an object instance.
Check to determine if the object is null before calling the method.
Get general help for this exception.
Search for more Help Online...
Actions:
View Detail...
Copy exception detail to the clipboard

	Value			
	Nothing	Byte()		SVACS_v2.exe!SVACS_v2.Mod_GlobFunc.Encryp
	256	Integer		SVACS_v2.exe!SVACS_v2.testForm.Button2_Clic

30.0 App.path

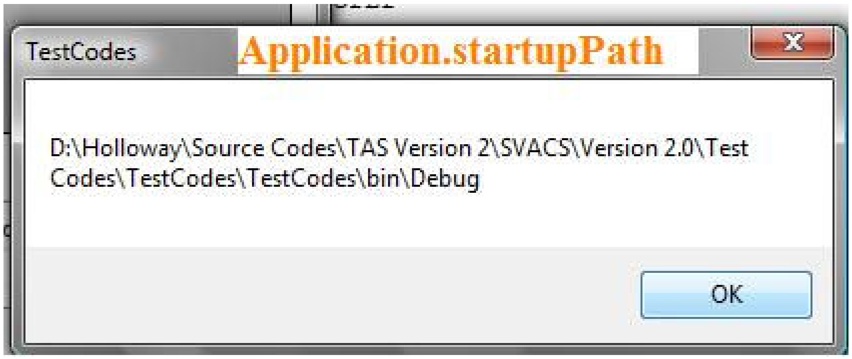
In VB.NET, use either

Application.ExecutablePath



D:\Holloway\Source Codes\TAS Version 2\SVACS\Version 2.0\Test Codes\TestCodes\TestCodes\bin\Debug\TestCodes.EXE

Application.StartupPath

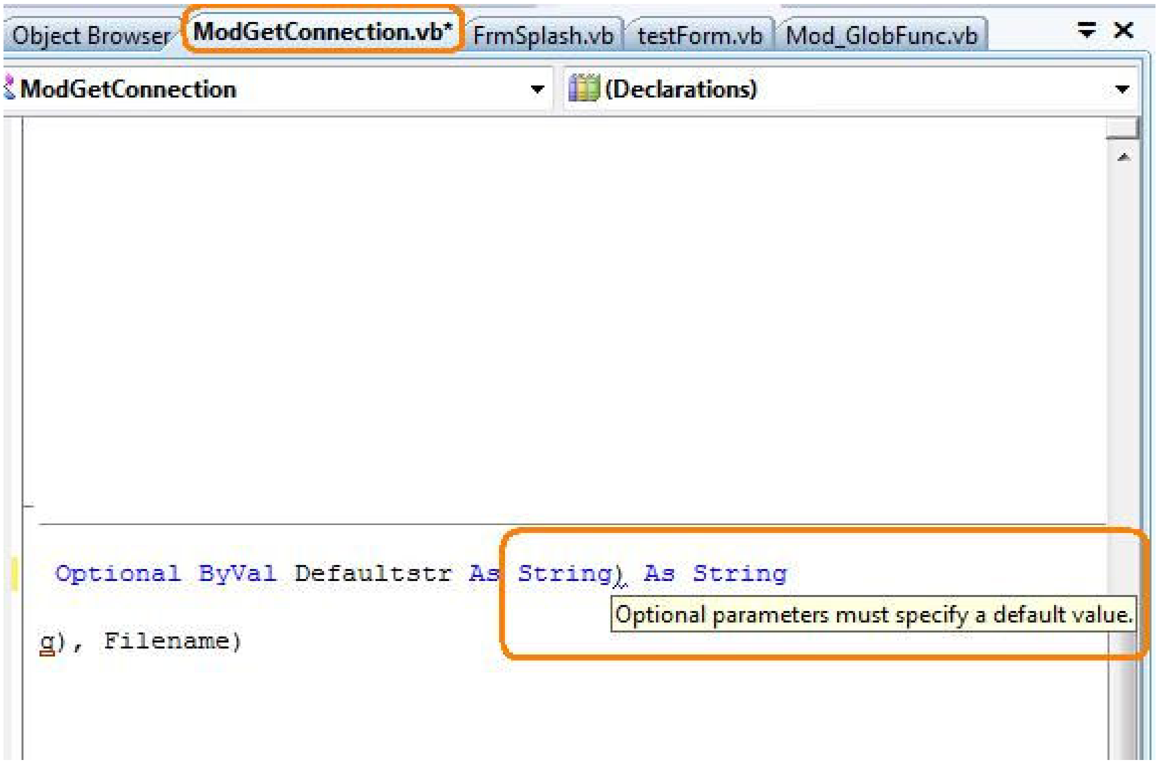


D:\Holloway\Source Codes\TAS Version 2\SVACS\Version 2.0\Test Codes\TestCodes\TestCodes\bin\Debug

Thus, config files must be placed inside ‘bin’ directory.

31.0 Optional Parameter

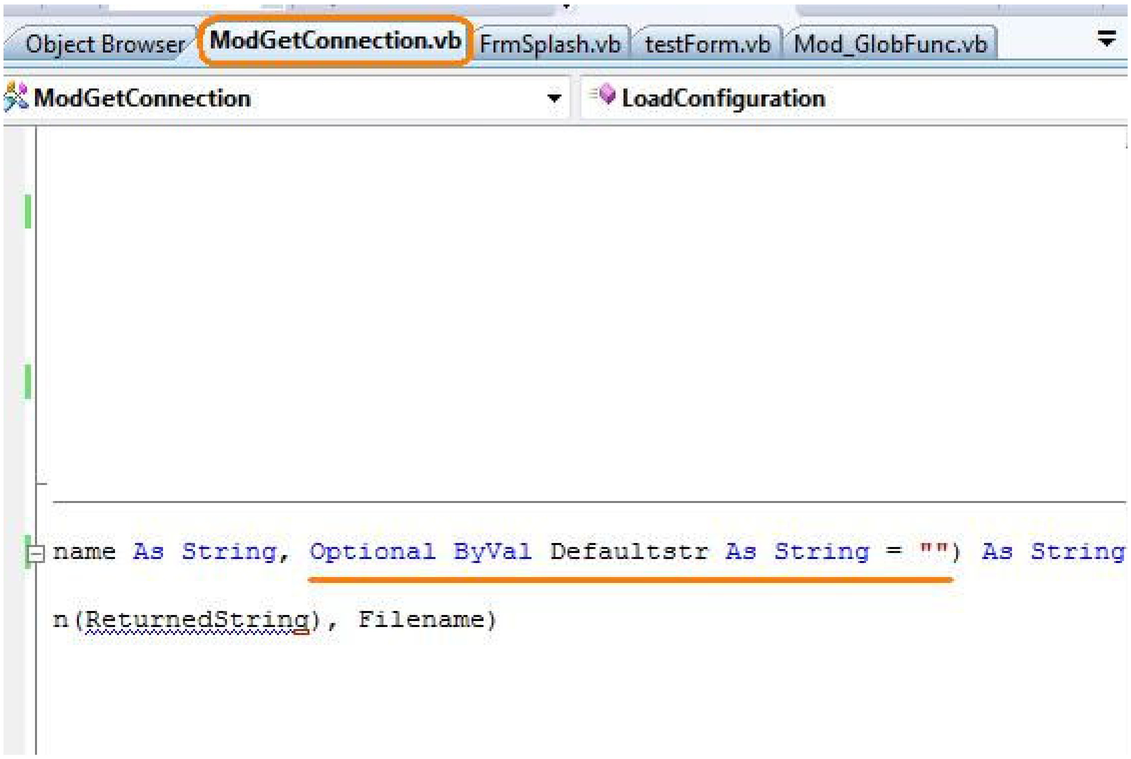
Must have a default value



```
 Optional ByVal Defaultstr As String) As String
```

Optional parameters must specify a default value.

```
g), Filename)
```



```
name As String, Optional ByVal Defaultstr As String = "") As String

n(ReturnedString), Filename)
```

What about for objects ?

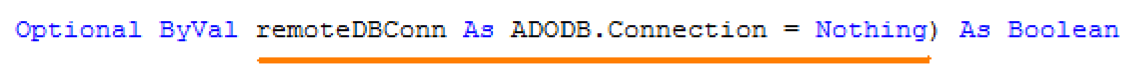
According to the following, VB.NET discourages optional parameters.

http://jelle.druyts.net/2003/05/17/OptionalParametersVBNET.aspx
http://www.codeguru.com/csharp/.net/net_general/tipstricks/article.php/c13505

Ideally, we should use ‘overload’ to have different definition for the same method.

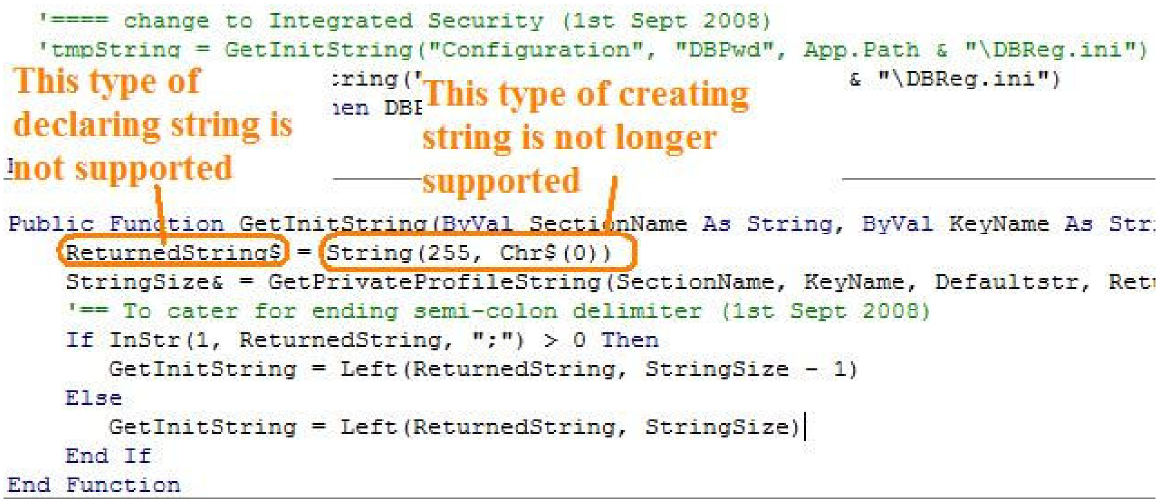
C# doesn’t support ‘optional’ parameter.

But for object, the default value is ‘nothing’.



```
Optional ByVal remoteDBConn As ADODB.Connection = Nothing) As Boolean
```

32.0 String(count, char)

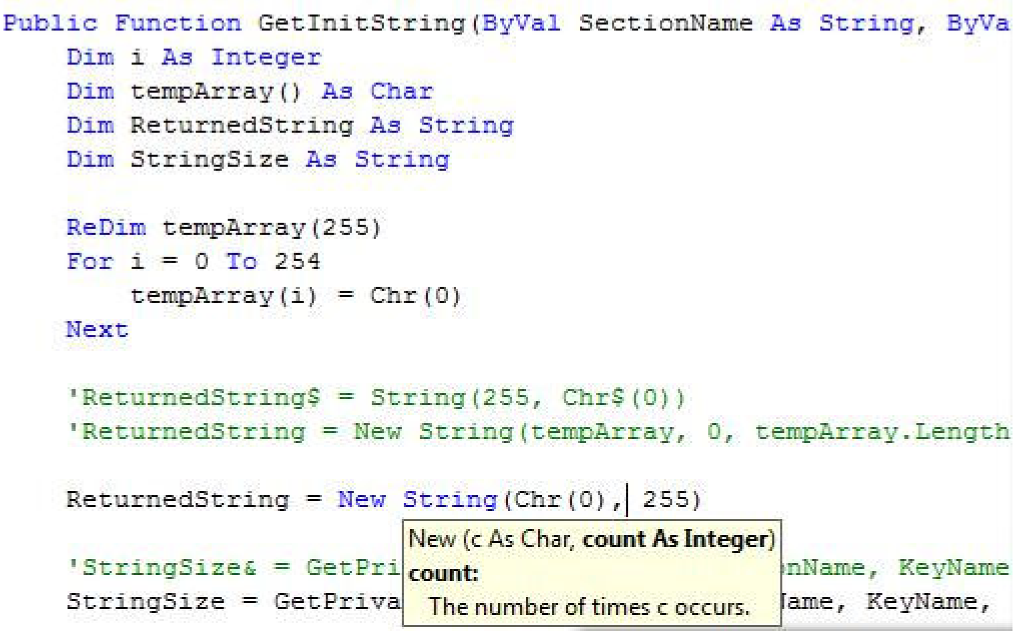


```
  '==== change to Integrated Security (1st Sept 2008)
  'tmpString = GetInitString("Configuration", "DBPwd", App.Path & "\DBReg.ini")
                  :ring("                                   & "\DBReg.ini")
                  ıen DBI

Public Function GetInitString(ByVal SectionName As String, ByVal KeyName As Str:
    ReturnedString$ = String(255, Chr$(0))
    StringSize& = GetPrivateProfileString(SectionName, KeyName, Defaultstr, Ret
    '== To cater for ending semi-colon delimiter (1st Sept 2008)
    If InStr(1, ReturnedString, ";") > 0 Then
       GetInitString = Left(ReturnedString, StringSize - 1)
    Else
       GetInitString = Left(ReturnedString, StringSize)
    End If
End Function
```

This type of declaring string is not supported

This type of creating string is not longer supported

http://bytes.com/groups/net-vb/387284-what-vb-net-equiv-vb6-string-number-character-function

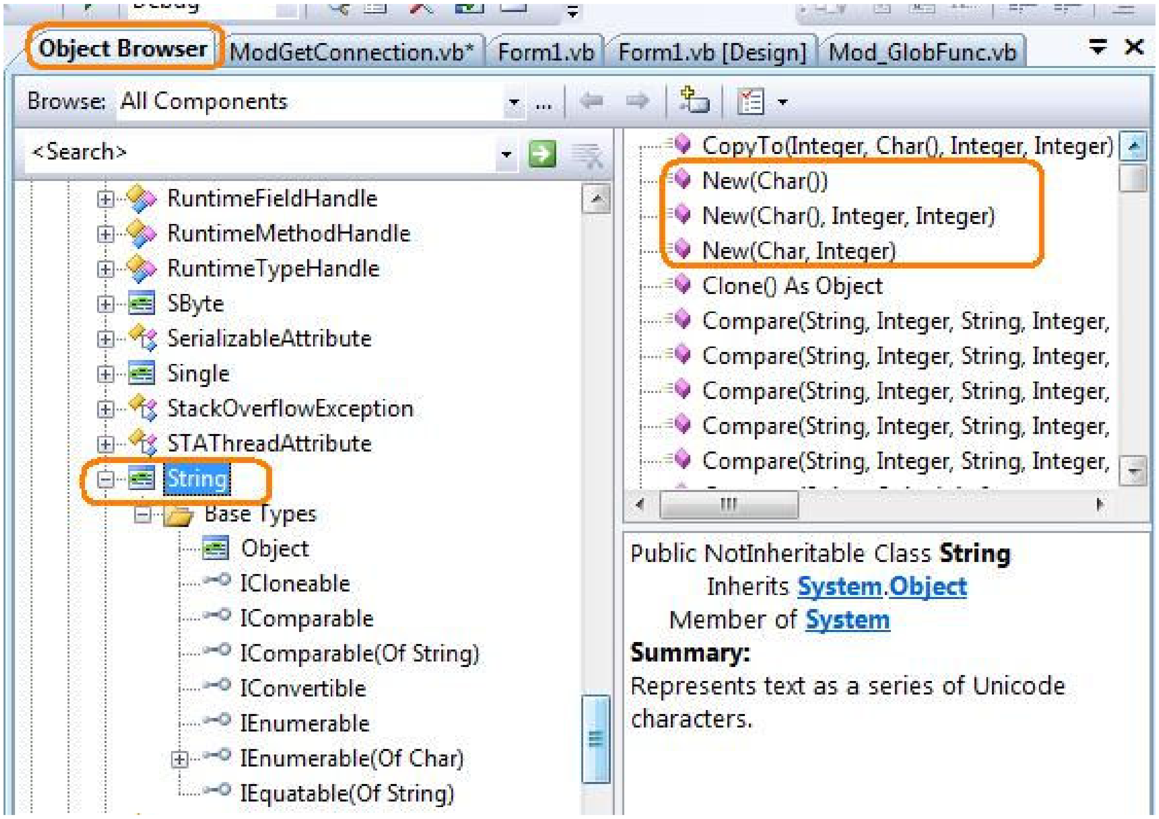


```
Public Function GetInitString(ByVal SectionName As String, ByVa
    Dim i As Integer
    Dim tempArray() As Char
    Dim ReturnedString As String
    Dim StringSize As String

    ReDim tempArray(255)
    For i = 0 To 254
        tempArray(i) = Chr(0)
    Next

    'ReturnedString$ = String(255, Chr$(0))
    'ReturnedString = New String(tempArray, 0, tempArray.Length

    ReturnedString = New String(Chr(0), 255)

    'StringSize& = GetPri                          nName, KeyName
    StringSize = GetPriva                          Name, KeyName,
```

New (c As Char, **count As Integer**)
count:
The number of times c occurs.

There are three overloaded new methods for string.

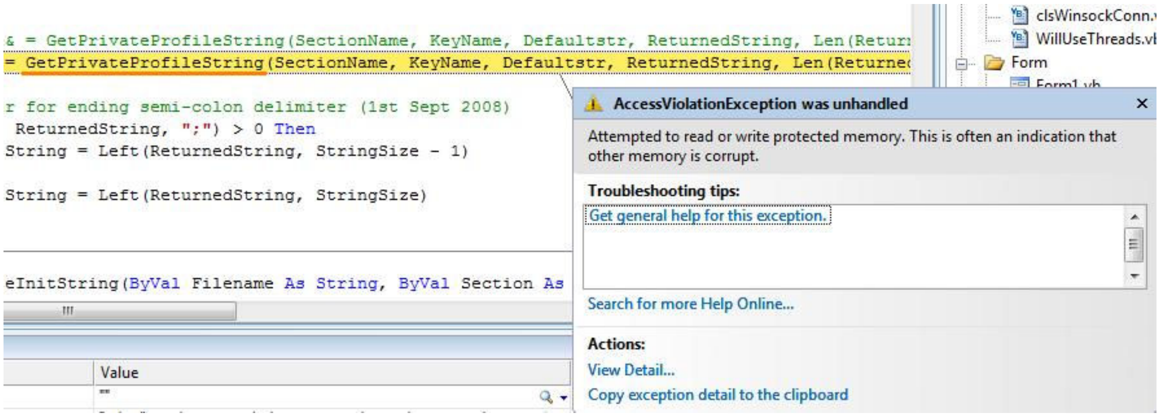
Check out the string definition.



Public NotInheritable Class **String**
Inherits System.Object
Member of System
Summary:
Represents text as a series of Unicode characters.

Win32 and COM Development:

Calling lib such as ‘GetPrivateProfileString’ may not work well with .NET.



```
& = GetPrivateProfileString(SectionName, KeyName, Defaultstr, ReturnedString, Len(Retur
= GetPrivateProfileString(SectionName, KeyName, Defaultstr, ReturnedString, Len(Returne

r for ending semi-colon delimiter (1st Sept 2008)
 ReturnedString, ";") > 0 Then
String = Left(ReturnedString, StringSize - 1)

String = Left(ReturnedString, StringSize)


eInitString(ByVal Filename As String, ByVal Section As
```

AccessViolationException was unhandled

Attempted to read or write protected memory. This is often an indication that other memory is corrupt.

Troubleshooting tips:

Get general help for this exception.

Search for more Help Online...

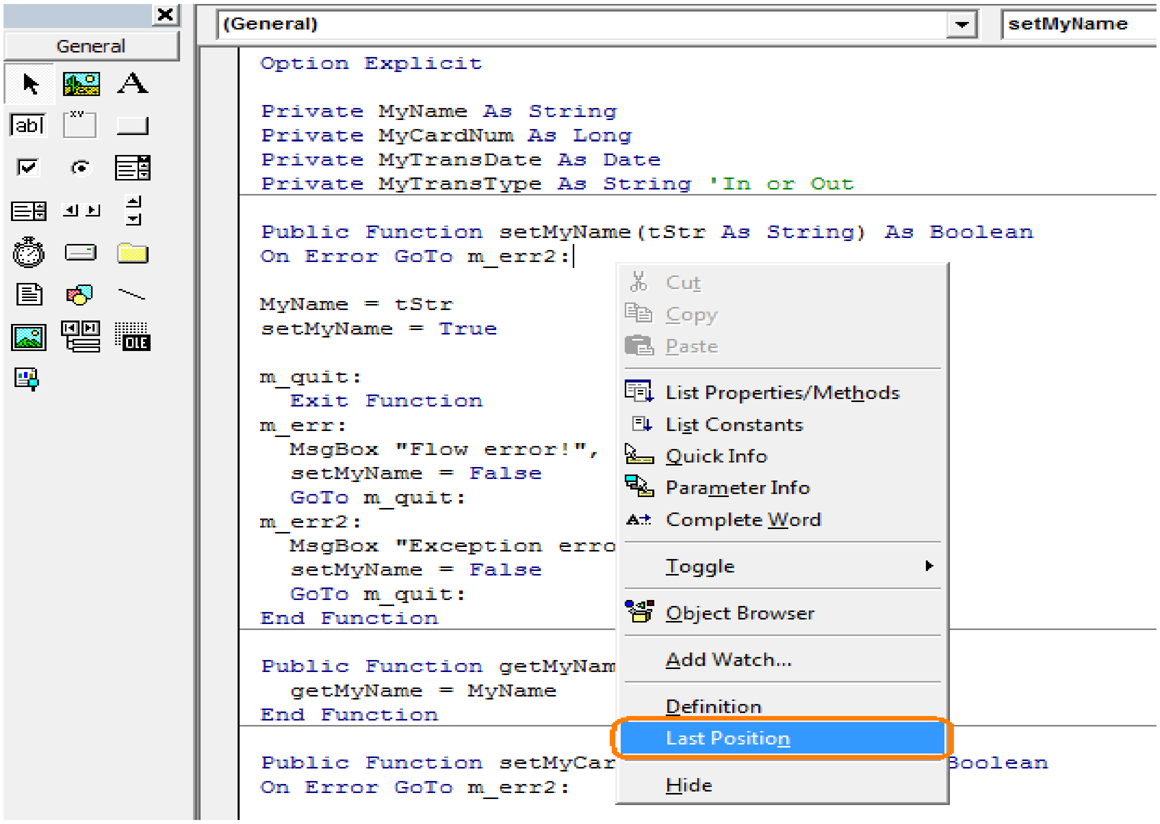
Actions:

View Detail...

Copy exception detail to the clipboard

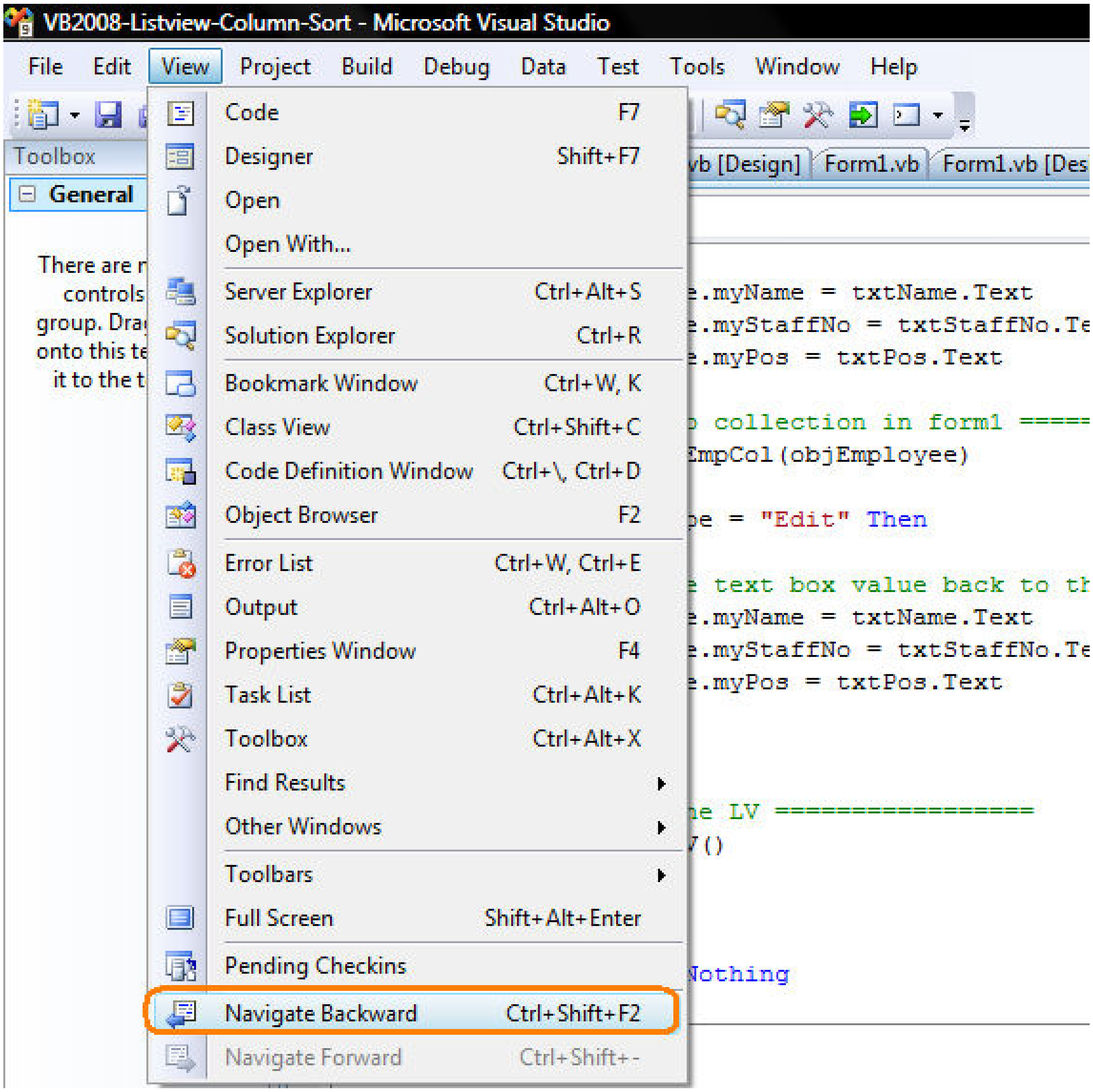
32.0 Last Position:

VB6:



```
Option Explicit

Private MyName As String
Private MyCardNum As Long
Private MyTransDate As Date
Private MyTransType As String 'In or Out

Public Function setMyName(tStr As String) As Boolean
On Error GoTo m_err2:

MyName = tStr
setMyName = True

m_quit:
  Exit Function
m_err:
  MsgBox "Flow error!",
  setMyName = False
  GoTo m_quit:
m_err2:
  MsgBox "Exception erro
  setMyName = False
  GoTo m_quit:
End Function

Public Function getMyNam
  getMyName = MyName
End Function

Public Function setMyCar                                         Boolean
On Error GoTo m_err2:
```

VB.NET:



```
e.myName = txtName.Text
e.myStaffNo = txtStaffNo.Te
e.myPos = txtPos.Text

o collection in form1 =====
EmpCol(objEmployee)

pe = "Edit" Then

e text box value back to th
e.myName = txtName.Text
e.myStaffNo = txtStaffNo.Te
e.myPos = txtPos.Text



he LV =================
V()



Nothing
```

33.0 Left

In VB.NET, you have to invoke the full path of Microsoft.Visualbasic.Left

```
dataBuffer = Microsoft.VisualBasic.Left(dataBuffer, totalDataLen - 5)
```

34.0 Format Date Time

34.1 Date Time Format:

Item	VB6	VB.NET

Year	yyyy	yyyy
Month	mm	MM (big capital)
Day	dd	dd
Hour	hh	hh
Minutes	nn	mm (small capital)
Seconds	ss	ss

34.2 VB.NET Format Date

Format a date to string is important for example when inserting records to MSSQL server which requires the date format to be in MM/dd/yyyy hh:mm:ss

VB6:

1. You can create a date using format string.

VB.NET

1. You can't create a date using format string.
2. You can only create a date using function such as new date(year, month, day, hour, minute, second).
3. When you format a date, don't assign the output to a date variable (because when assign a string to a date variable, the format must be based on the regional setting) or perform CDate(), assign it to a string variable instead

Refer to 'Format-Date-Sample-1'

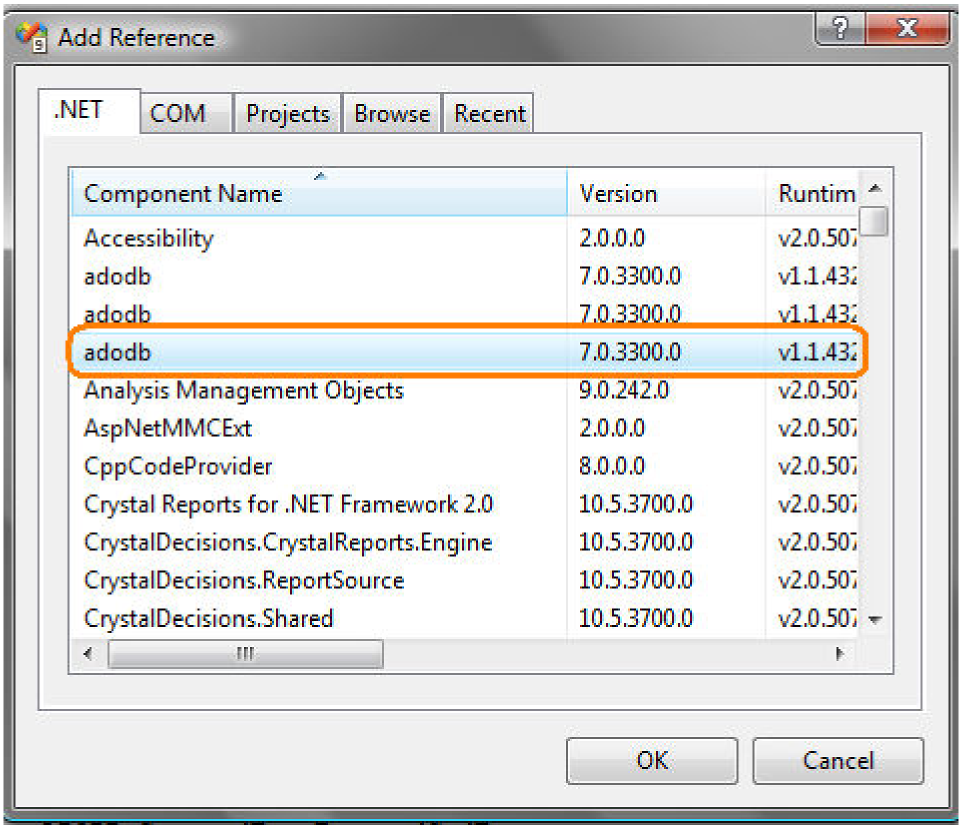
http://rapidshare.com/files/304427621/Format-Date-Sample-1.zip

35.0 ADODB

Adodb is supported in VB.NET

35.1 Add Reference

You just have to add reference.



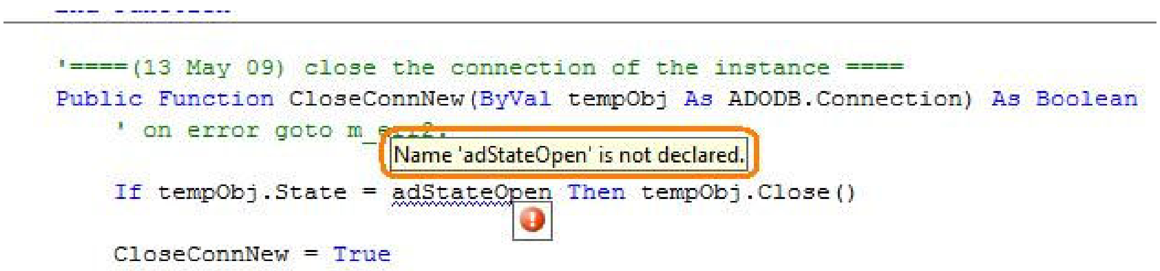
Component Name	Version	Runtim
Accessibility	2.0.0.0	v2.0.507
adodb	7.0.3300.0	v1.1.432
adodb	7.0.3300.0	v1.1.432
adodb	7.0.3300.0	v1.1.432
Analysis Management Objects	9.0.242.0	v2.0.507
AspNetMMCExt	2.0.0.0	v2.0.507
CppCodeProvider	8.0.0.0	v2.0.507
Crystal Reports for .NET Framework 2.0	10.5.3700.0	v2.0.507
CrystalDecisions.CrystalReports.Engine	10.5.3700.0	v2.0.507
CrystalDecisions.ReportSource	10.5.3700.0	v2.0.507
CrystalDecisions.Shared	10.5.3700.0	v2.0.507

35.2 CommandType

And there are changes to the 'CommandType'

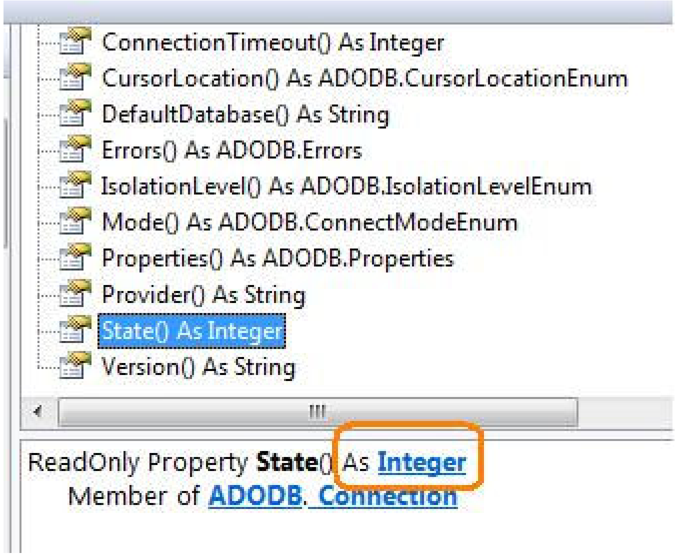
VB6	VB.NET
AdCmdFile	ADODB.CommandTypeEnum.adCmdFile
AdCmdStoredProc	ADODB.CommandTypeEnum.adCmdStoredProc
AdCmdTable	ADODB.CommandTypeEnum.adCmdTable
AdCmdTableDirect	ADODB.CommandTypeEnum.adCmdTableDirect
AdCmdText	ADODB.CommandTypeEnum.adCmdText
AdCmdUnknown	ADODB.CommandTypeEnum.adCmdUnknown
AdCmdUnspecified	ADODB.CommandTypeEnum.adCmdUnspecified

35.3 Adodb.State:



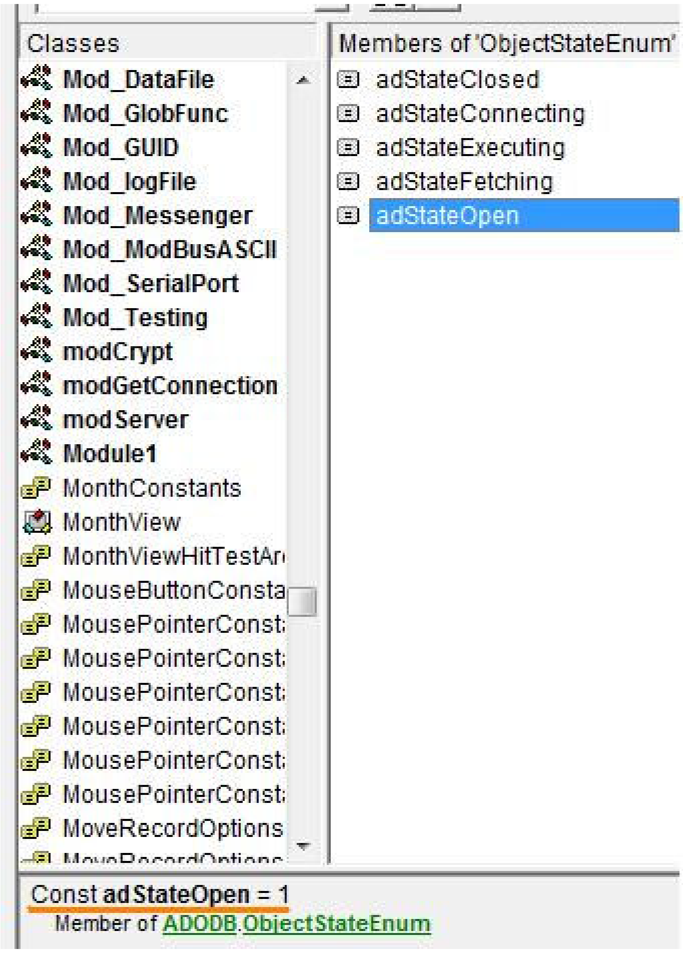
```
End Function

'=====(13 May 09) close the connection of the instance ====
Public Function CloseConnNew(ByVal tempObj As ADODB.Connection) As Boolean
    ' on error goto m_err2.

    If tempObj.State = adStateOpen Then tempObj.Close()

    CloseConnNew = True
```

Name 'adStateOpen' is not declared.



```
ConnectionTimeout() As Integer
CursorLocation() As ADODB.CursorLocationEnum
DefaultDatabase() As String
Errors() As ADODB.Errors
IsolationLevel() As ADODB.IsolationLevelEnum
Mode() As ADODB.ConnectModeEnum
Properties() As ADODB.Properties
Provider() As String
State() As Integer
Version() As String
```

```
ReadOnly Property State() As Integer
    Member of ADODB.Connection
```



Classes	Members of 'ObjectStateEnum'
Mod_DataFile	adStateClosed
Mod_GlobFunc	adStateConnecting
Mod_GUID	adStateExecuting
Mod_logFile	adStateFetching
Mod_Messenger	adStateOpen
Mod_ModBusASCII	
Mod_SerialPort	
Mod_Testing	
modCrypt	
modGetConnection	
modServer	
Module1	
MonthConstants	
MonthView	
MonthViewHitTestAr	
MouseButtonConsta	
MousePointerConst	
MousePointerConst	
MousePointerConst	
MousePointerConst	
MousePointerConst	
MousePointerConst	
MoveRecordOptions	

```
Const adStateOpen = 1
    Member of ADODB.ObjectStateEnum
```

Just use number the number.

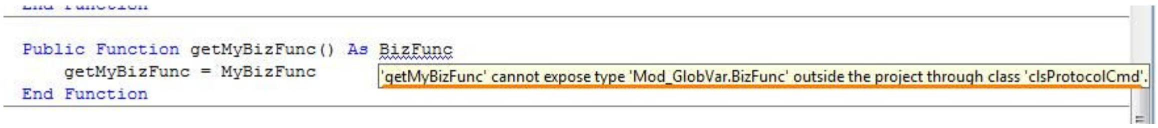
- adStateOpen = 1
- adStateFetching = 8
- adStateExecuting = 4
- adStateConnecting = 2
- adStateClosed = 0

35.4 Saving and Retrieving Image from MSSQL

Refer to the following samples

http://www.it-sideways.com/2009/11/save-and-retrieve-image-from-database.html

36.0 Enum



```
End Function

Public Function getMyBizFunc() As BizFunc
    getMyBizFunc = MyBizFunc
End Function
```

'getMyBizFunc' cannot expose type 'Mod_GlobVar.BizFunc' outside the project through class 'clsProtocolCmd'.

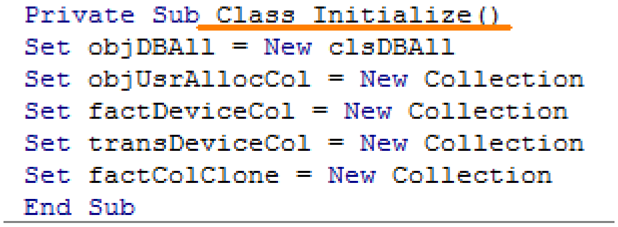
http://forum.strataframe.net/Topic5232-14-1.aspx

Change the Module to 'Public'

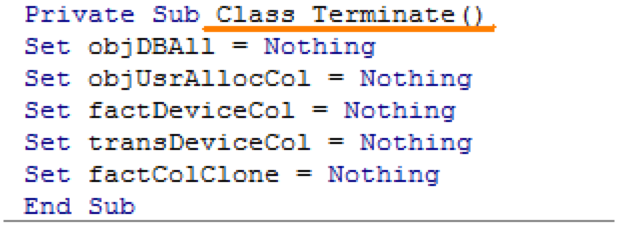
It suggested not to use 'Module', use 'shared' class instead.

37.0 Class_Initialize

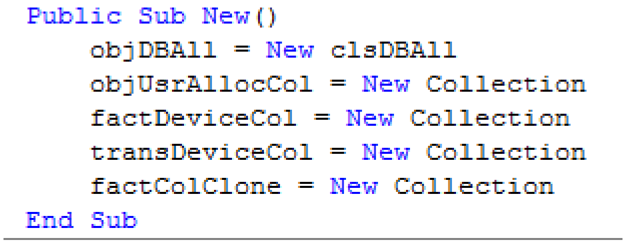
Item	VB6	VB.NET
	Class_Initialize	New()
	Class_Terminate	Finalize()



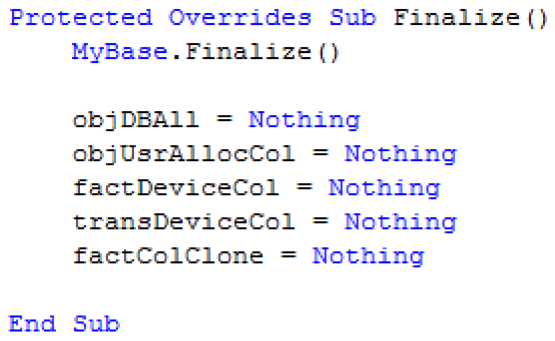
```
Private Sub Class_Initialize()
Set objDBAll = New clsDBAll
Set objUsrAllocCol = New Collection
Set factDeviceCol = New Collection
Set transDeviceCol = New Collection
Set factColClone = New Collection
End Sub
```



```
Private Sub Class_Terminate()
Set objDBAll = Nothing
Set objUsrAllocCol = Nothing
Set factDeviceCol = Nothing
Set transDeviceCol = Nothing
Set factColClone = Nothing
End Sub
```



```
Public Sub New()
    objDBAll = New clsDBAll
    objUsrAllocCol = New Collection
    factDeviceCol = New Collection
    transDeviceCol = New Collection
    factColClone = New Collection
End Sub
```



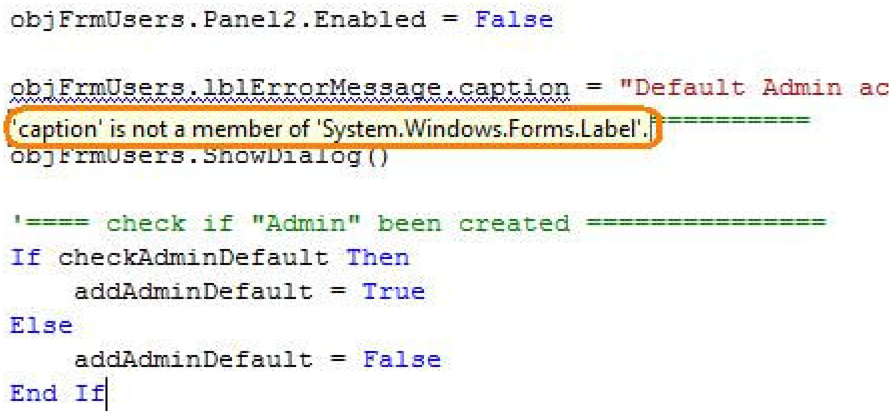
```
Protected Overrides Sub Finalize()
    MyBase.Finalize()

    objDBAll = Nothing
    objUsrAllocCol = Nothing
    factDeviceCol = Nothing
    transDeviceCol = Nothing
    factColClone = Nothing

End Sub
```

38.0 Label

38.1 Caption:

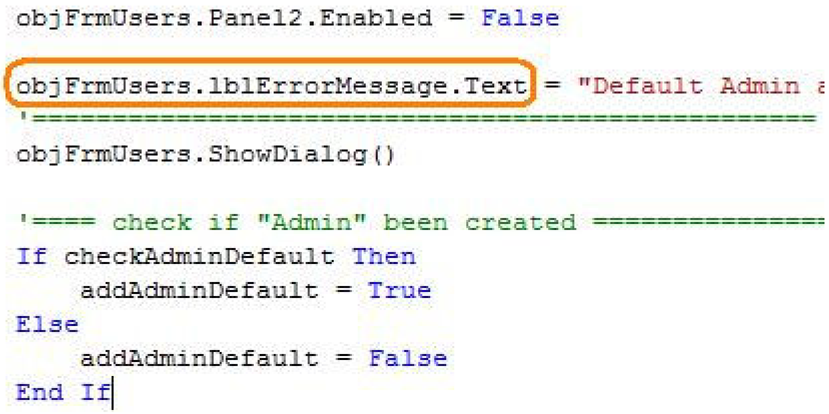
There is no more caption property in VB6.

You have to switch to the text property.



```
objFrmUsers.Panel2.Enabled = False

objFrmUsers.lblErrorMessage.caption = "Default Admin ac
'caption' is not a member of 'System.Windows.Forms.Label'.
objFrmUsers.ShowDialog()

'==== check if "Admin" been created ===============
If checkAdminDefault Then
    addAdminDefault = True
Else
    addAdminDefault = False
End If
```

Not Working

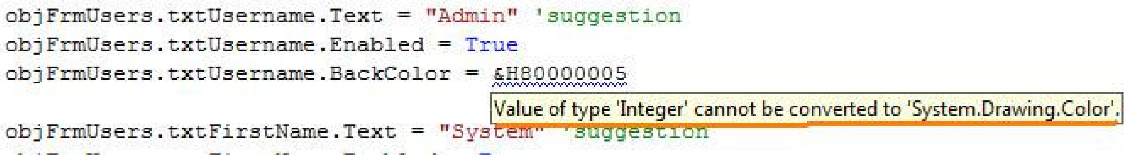


```
objFrmUsers.Panel2.Enabled = False

objFrmUsers.lblErrorMessage.Text = "Default Admin a
'==================================================
objFrmUsers.ShowDialog()

'==== check if "Admin" been created ===============
If checkAdminDefault Then
    addAdminDefault = True
Else
    addAdminDefault = False
End If
```

Working

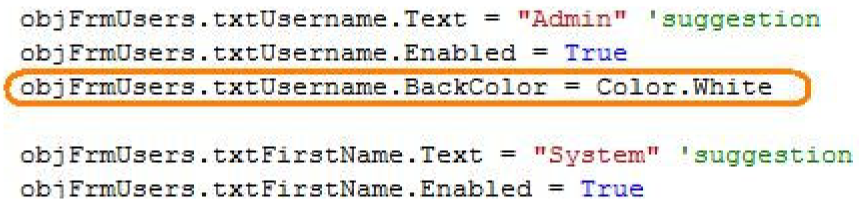
39.0 Color Code:

You can’t use the hex code anymore.



```
objFrmUsers.txtUsername.Text = "Admin" 'suggestion
objFrmUsers.txtUsername.Enabled = True
objFrmUsers.txtUsername.BackColor = &H80000005
Value of type 'Integer' cannot be converted to 'System.Drawing.Color'.
objFrmUsers.txtFirstName.Text = "System" 'suggestion
```

Not working

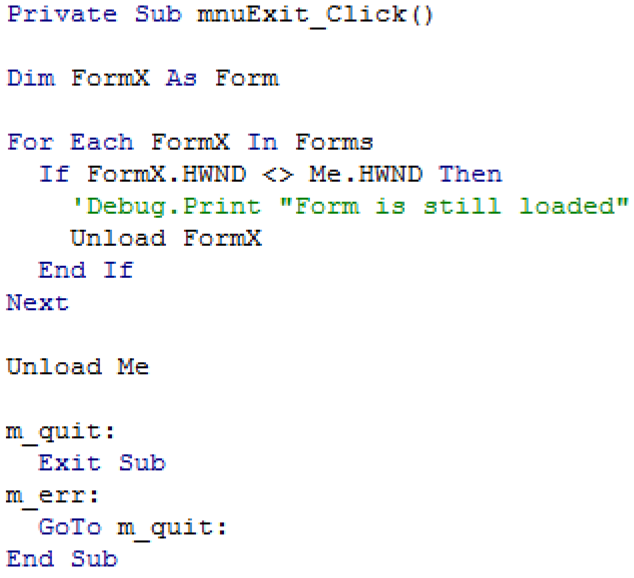


```
objFrmUsers.txtUsername.Text = "Admin" 'suggestion
objFrmUsers.txtUsername.Enabled = True
objFrmUsers.txtUsername.BackColor = Color.White

objFrmUsers.txtFirstName.Text = "System" 'suggestion
objFrmUsers.txtFirstName.Enabled = True
```

Working

40. Exit Application

VB6:



```
Private Sub mnuExit_Click()

Dim FormX As Form

For Each FormX In Forms
  If FormX.HWND <> Me.HWND Then
    'Debug.Print "Form is still loaded"
    Unload FormX
  End If
Next

Unload Me

m_quit:
  Exit Sub
m_err:
  GoTo m_quit:
End Sub
```

VB.NET

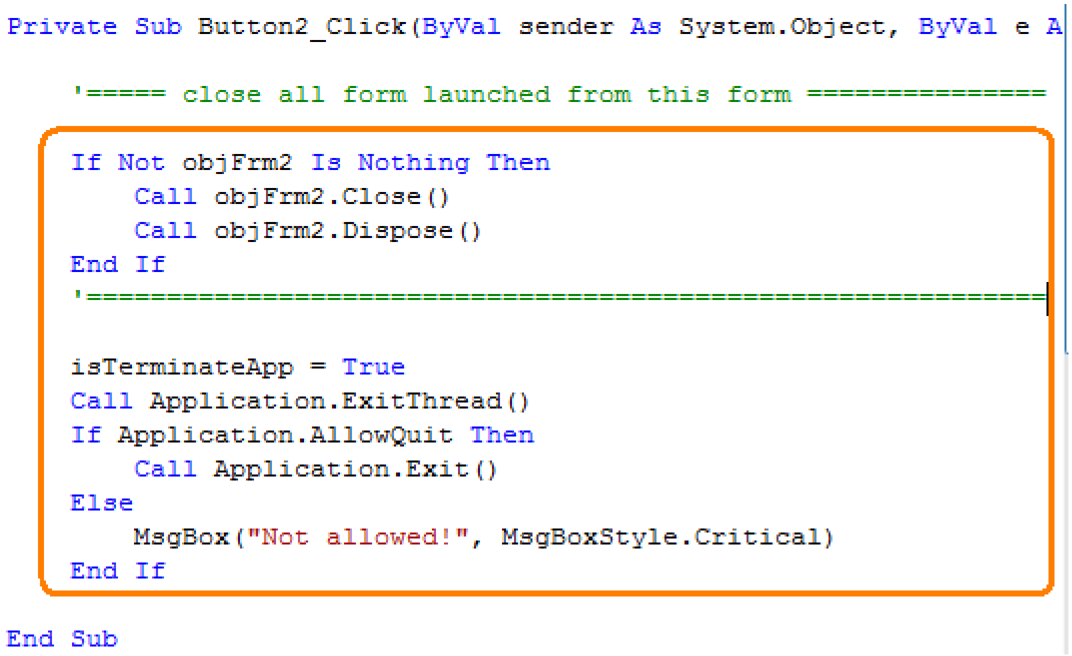
Make use of the combination of:

1. Application.AllowQuit
2. Application.ExitThread
3. Application.Exit

Tips:

1. Make use of ‘Application.AllowQuit’ property to see if you can quit the application safely.
2. For every form to shut down, must first invoke ‘Application.ExitThread()’ before calling the close() method. Otherwise, you may get the ‘Application.AllowQuit’ return as false.

3. For the main form, first of all close all other forms launched from the main form, then perform ‘Application.ExitThread()’, check for ‘Application.AllowQuit’ and then only perform ‘Application.Exit()’



```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e A

    '===== close all form launched from this form ===============

    If Not objFrm2 Is Nothing Then
        Call objFrm2.Close()
        Call objFrm2.Dispose()
    End If
    '============================================================

    isTerminateApp = True
    Call Application.ExitThread()
    If Application.AllowQuit Then
        Call Application.Exit()
    Else
        MsgBox("Not allowed!", MsgBoxStyle.Critical)
    End If

End Sub
```

Refer to the following sample:

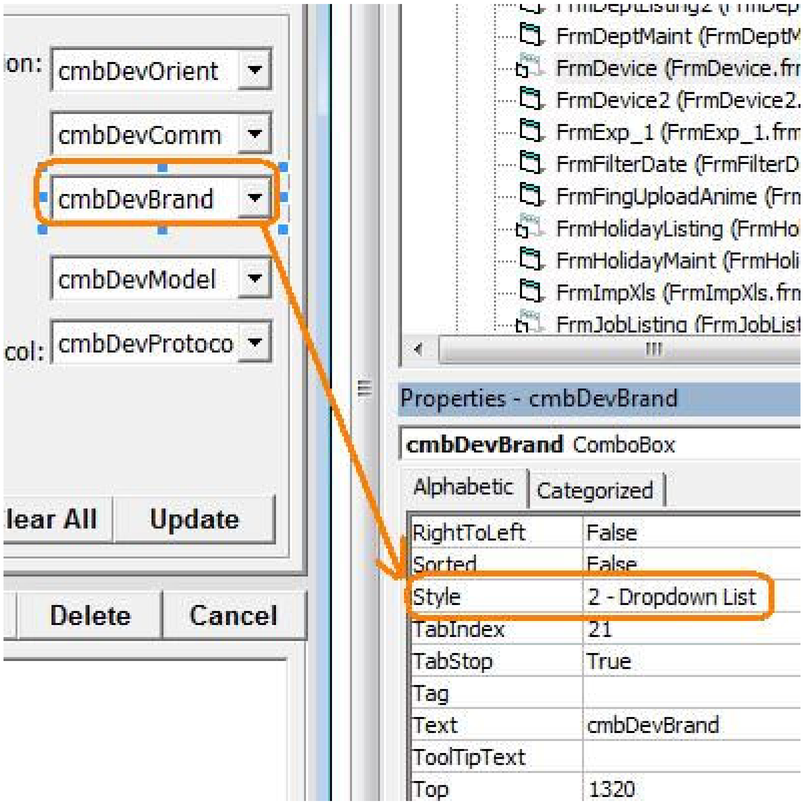
http://rapidshare.com/files/306805704/AllowQuit-Sample-1.zip

41.0 Combo Box:

41.1 Combo Box Style:

VB6:

Use ‘Style’



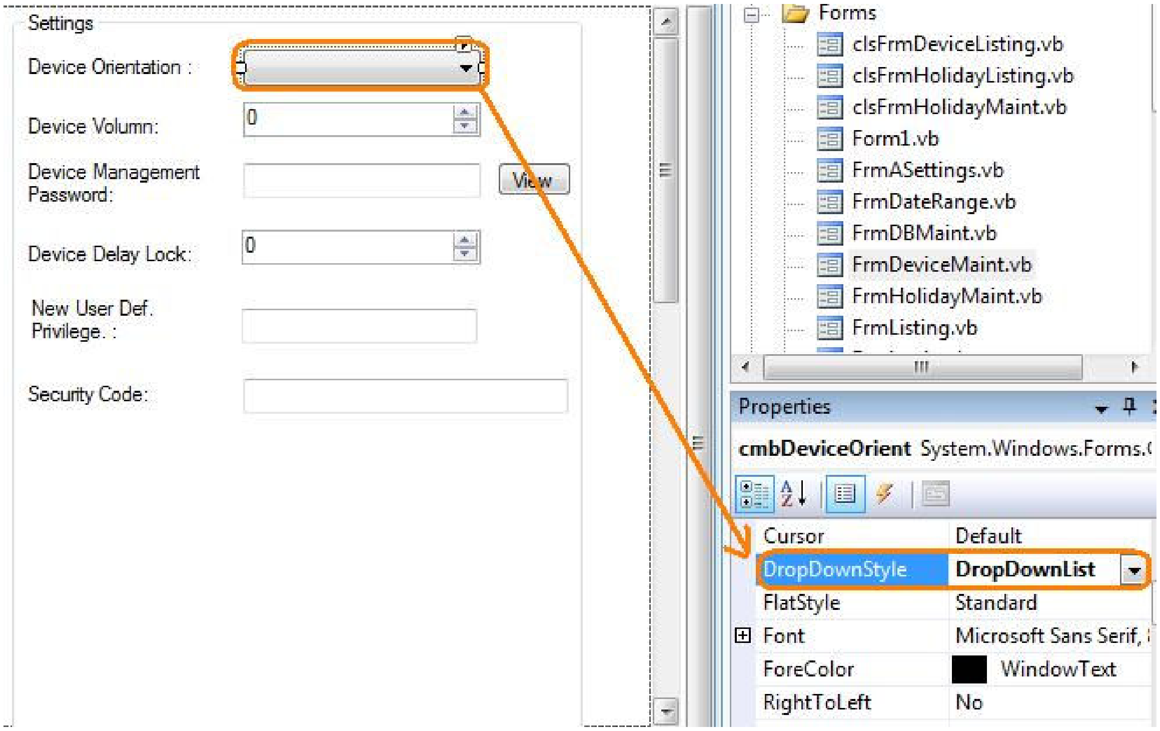
Properties - cmbDevBrand

cmbDevBrand ComboBox

RightToLeft	False
Sorted	False
Style	2 - Dropdown List
TabIndex	21
TabStop	True
Tag	
Text	cmbDevBrand
ToolTipText	
Top	1320

VB.NET

Use 'DropDownStyle'



Properties

cmbDeviceOrient System.Windows.Forms.(

Cursor	Default
DropDownStyle	**DropDownList**
FlatStyle	Standard
Font	Microsoft Sans Serif,
ForeColor	■ WindowText
RightToLeft	No

42. ByVal & ByRef

http://www.it-sideways.com/2009/05/byval-vs-byref-visual-basic.html

43. Create Leading Zeros

VB6:

For String or Integer

→ Use format

VB.NET

For Integer

→ Use format

For String

→ Use padleft.

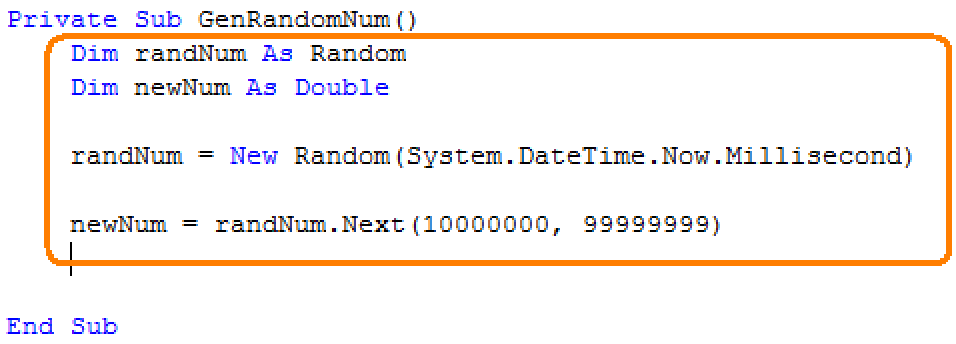
44.0 Generate Random Numbers:

VB6:

http://rapidshare.com/files/306685739/Random-Number-Sample-1.zip

VB.NET:

Use 'Random' class.



```
Private Sub GenRandomNum()
    Dim randNum As Random
    Dim newNum As Double

    randNum = New Random(System.DateTime.Now.Millisecond)

    newNum = randNum.Next(10000000, 99999999)


End Sub
```

45.0 StdPicture

In VB.Net, use ‘bitmap’ object variable.

Refer to the samples for saving and retrieving image from database from

http://www.it-sideways.com/2009/11/save-and-retrieve-image-from-database.html

46.0 DCOM

Do away DCOM and use ‘Remoting’

Refer to the following two samples.

http://www.it-sideways.com/2009/09/vbnet-remoting-aspnet-client.html
http://www.it-sideways.com/2009/08/vbnet-remoting-sample-code.html

47.0 ‘Set’ Keyword

In VB6, when you create a new object such as collection, ADODB object or etc, you have to use the ‘set’ keyword.

For instance,

```
‘ To Create New Collection
Dim tempCol as collection
Set tempCol = new collection

‘To Set Object to Nothing
Set tempCol = nothing
```

In VB.NET, the ‘set’ keyword will not be required.

```
‘ To Create New Collection
Dim tempCol as collection
tempCol = new collection

‘To Set Object to Nothing
tempCol = nothing
```

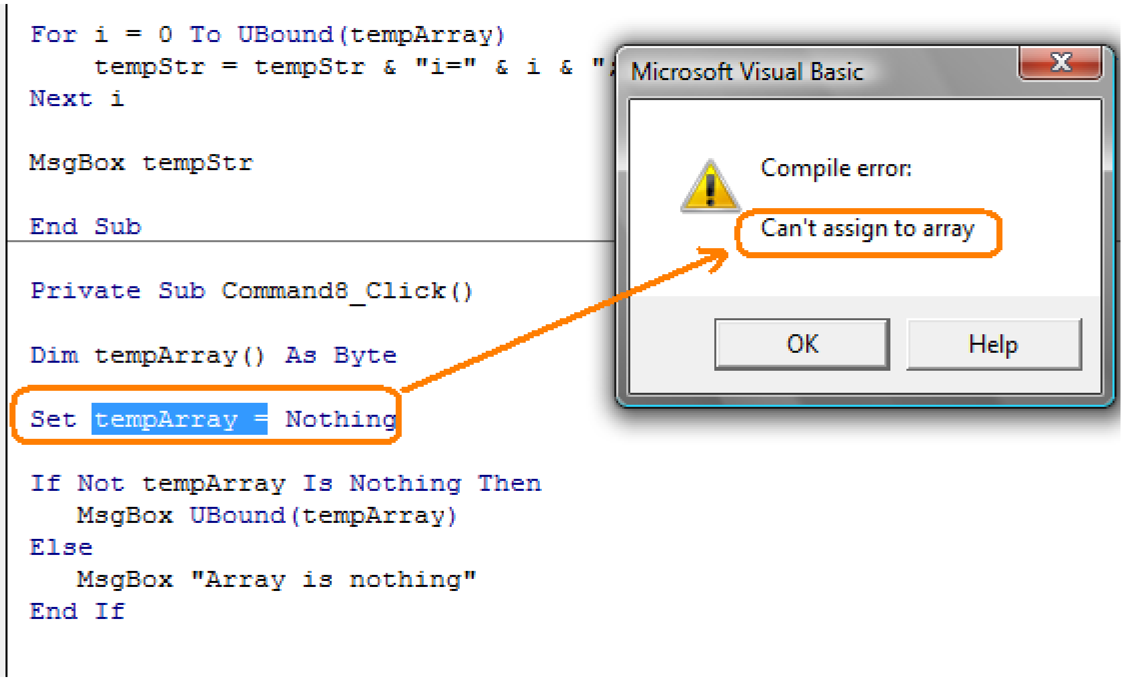
48. Array

VB6:

1. You can't pass an array parameter byVal.

Refer to ''

2. You can't set an array to nothing (or null).

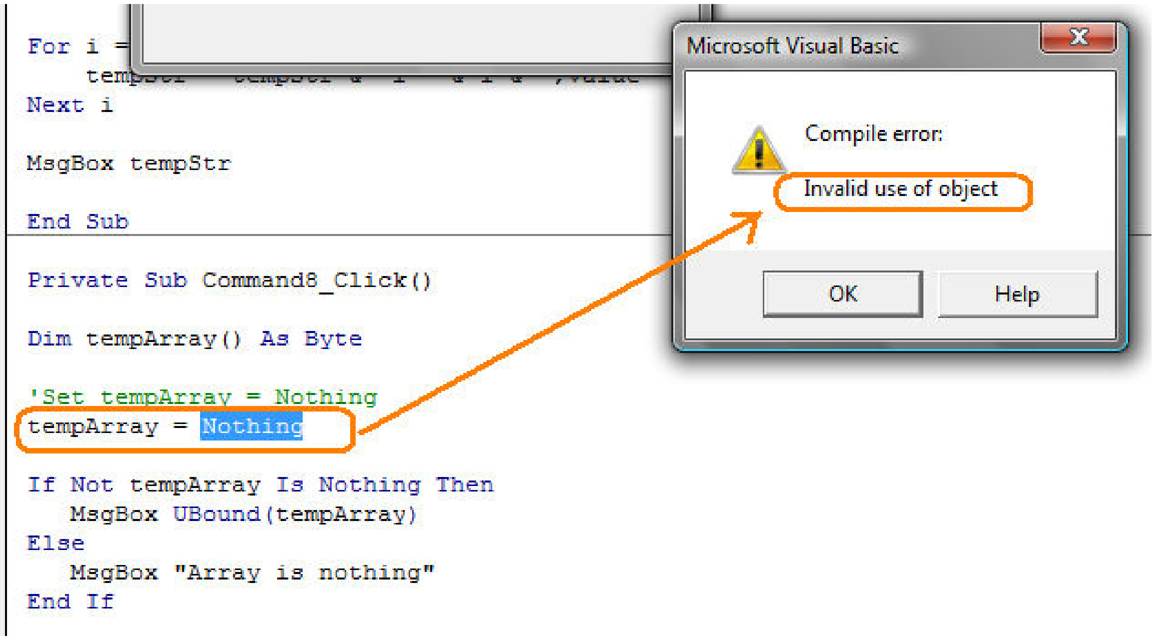


```
For i = 0 To UBound(tempArray)
    tempStr = tempStr & "i=" & i & ";
Next i

MsgBox tempStr

End Sub

Private Sub Command8_Click()

Dim tempArray() As Byte

Set tempArray = Nothing

If Not tempArray Is Nothing Then
   MsgBox UBound(tempArray)
Else
   MsgBox "Array is nothing"
End If
```

Microsoft Visual Basic

Compile error:

Can't assign to array

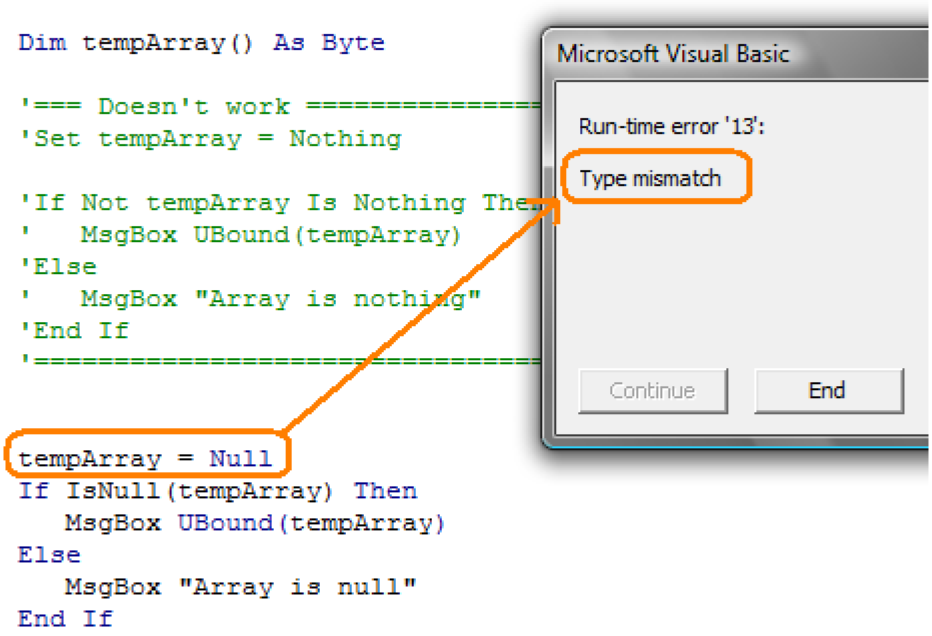


```
For i =
    tempStr = tempStr & "i=" & i & ";value
Next i

MsgBox tempStr

End Sub

Private Sub Command8_Click()

Dim tempArray() As Byte

'Set tempArray = Nothing
tempArray = Nothing

If Not tempArray Is Nothing Then
   MsgBox UBound(tempArray)
Else
   MsgBox "Array is nothing"
End If
```

Microsoft Visual Basic

Compile error:

Invalid use of object



```
Dim tempArray() As Byte

'=== Doesn't work ===============
'Set tempArray = Nothing

'If Not tempArray Is Nothing The
'    MsgBox UBound(tempArray)
'Else
'    MsgBox "Array is nothing"
'End If
'===================================


tempArray = Null
If IsNull(tempArray) Then
   MsgBox UBound(tempArray)
Else
   MsgBox "Array is null"
End If
```

Microsoft Visual Basic

Run-time error '13':

Type mismatch

<u>VB.NET:</u>

1. You CAN pass an array parameter byVal.

This is actually important for thread-concurrency.

Refer to ‘’

2. You CAN set an array to nothing.

```
Dim tempArray() As Byte

tempArray = Nothing

If Not tempArray Is Nothing Then
   MsgBox(UBound(tempArray))
Else
   MsgBox("Array is nothing")
End If
```

This is good to reinitialize an array variable.

3. You CAN redim an array as to -1

```
Dim tempArray() As Byte

ReDim tempArray(-1)

MsgBox(UBound(tempArray))
```

This is also another alternative to reinitialize an array variable.

49.0 Printers (Getting a listing of installed printers)

VB6:

```
Dim p As Printer

 For Each p In Printers
   Call cmbPrinter.AddItem(p.DeviceName)
 Next p
```

VB.NET:

Using System.Management namespace.

Refer to ‘http://msdn.microsoft.com/en-us/library/system.management.aspx’

http://www.dotnetcurry.com/ShowArticle.aspx?ID=148&AspxAutoDetectCookieSupport=1

Right Click Project > Add Reference > Select .NET Tab > System.Management. Also add the following namespace in the form.

```
Dim oquery As System.Management.ObjectQuery = New
System.Management.ObjectQuery("SELECT * FROM Win32_Printer")

Dim mosearcher As System.Management.ManagementObjectSearcher = New
System.Management.ManagementObjectSearcher(oquery)

Dim moc As System.Management.ManagementObjectCollection =
mosearcher.Get()

For Each tPrinter As ManagementObject In moc

     cmbPrinter.Items.Add(tPrinter("DeviceID"))

Next tPrinter
```

You only need to use these four properties.

DeviceID
DriverName
Name (same as DeviceID)
PortName

Get a full listing of available properties from the appendix or using the following codes to generate.

```
Dim oquery As System.Management.ObjectQuery = New
System.Management.ObjectQuery("SELECT * FROM Win32_Printer")

Dim mosearcher As System.Management.ManagementObjectSearcher = New
System.Management.ManagementObjectSearcher(oquery)

Dim moc As System.Management.ManagementObjectCollection =
mosearcher.Get()

For Each tPrinter As ManagementObject In moc

     Dim pdc As System.Management.PropertyDataCollection =
tPrinter.Properties

     For Each pd As System.Management.PropertyData In pdc

                cmbPrinterProperty.Items.Add(pd.Name)

      Next pd

Next tPrinter
```

Refer to sample code here.

http://www.it-sideways.com/2010/01/printer-listing-vbnet.html

50.0 Crystal Report:

If you are using Visual Studio 2008, you do not need to have Crystal Report (as separate software).

- Change 'CRAXDRT.Report' to 'CrystalDecisions.CrystalReports.Engine.ReportDocument '
- Change 'CRAXDRT.Section' to 'CrystalDecisions.CrystalReports.Engine.Section'
- Change 'CRAXDRT.DatabaseTable' to 'CrystalDecisions.CrystalReports.Engine.Table'

There is no such thing as CRAXDRT.Application. Use 'CrystalDecisions.CrystalReports.Engine.ReportDocument' to load document.

Overall, the VB.NET offers a much simpler implementation for reporting using Crystal Reports.

VB6:

Download the sample from.

http://rapidshare.com/files/337247438/VB6_CR85_BestPractice.zip

VB.NET:

Download the sample from

http://www.it-sideways.com/2010/01/crystal-report-in-vbnet-sample.html

Top 10 Coolest Things about VB.NET:

1.0 Threading and Async.

Basically, asynchronous processing means executing a process in a new thread so that you can carry on what you doing without waiting for the process to finish. If you do not execute a process in a new thread, then you are executing a synchronous process, which is the main thread (GUI thread) will wait for that process to finish.

Therefore, asynchronous is actually multi-threading.

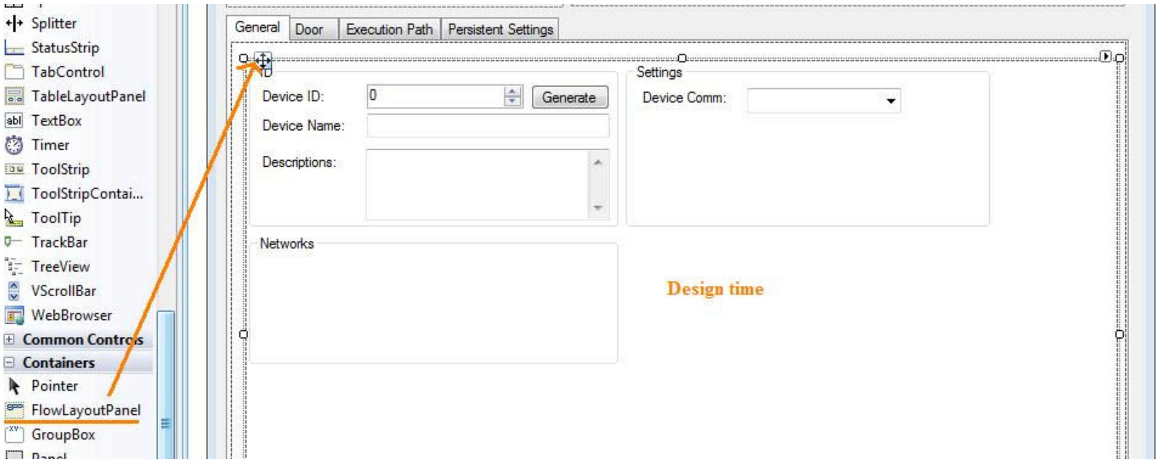
Refer to the following three articles and samples on multi-threading.

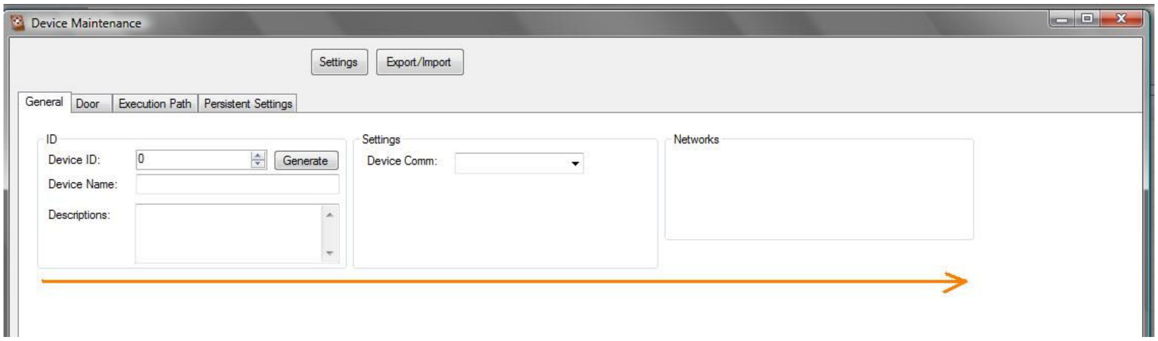
http://www.it-sideways.com/2009/07/vbnet-systemthreadingreadwritelock.html

http://www.it-sideways.com/2009/08/vbnet-systemthreadinginterlocked.html

http://www.it-sideways.com/2009/08/vbnet-thread-concurrency-for-module.html

2.0 Flow Layout

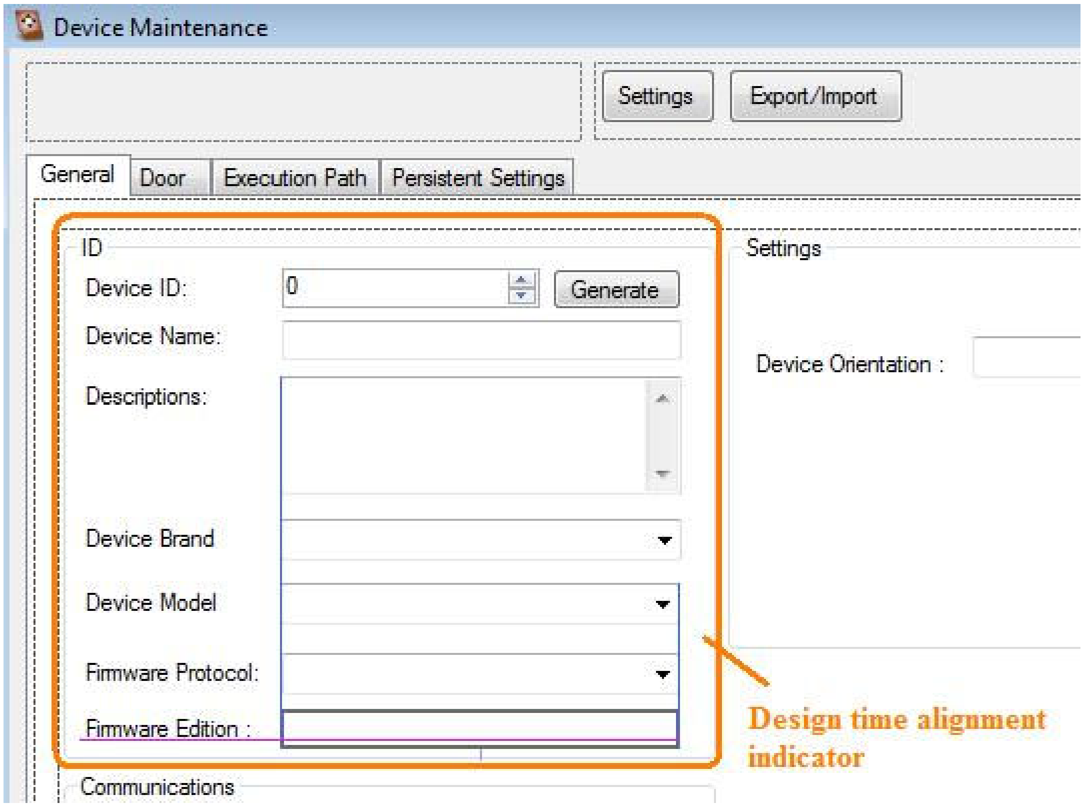




Refer to sample from ..

http://www.it-sideways.com/2009/07/how-to-make-form-controls-grow.html

3.0 Design Time Alignment Indicators



Device ID: 0

Design time alignment indicator

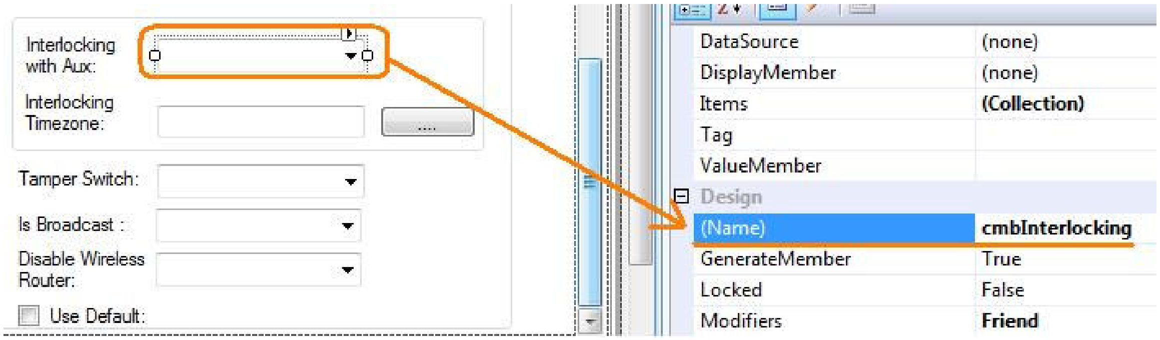
Using Docking, Autosize, AutoScroll for Auto Resizing

Refer to sample from.

http://www.it-sideways.com/2009/07/how-to-make-form-controls-grow.html

4.0 Auto resolve changes to control’s name

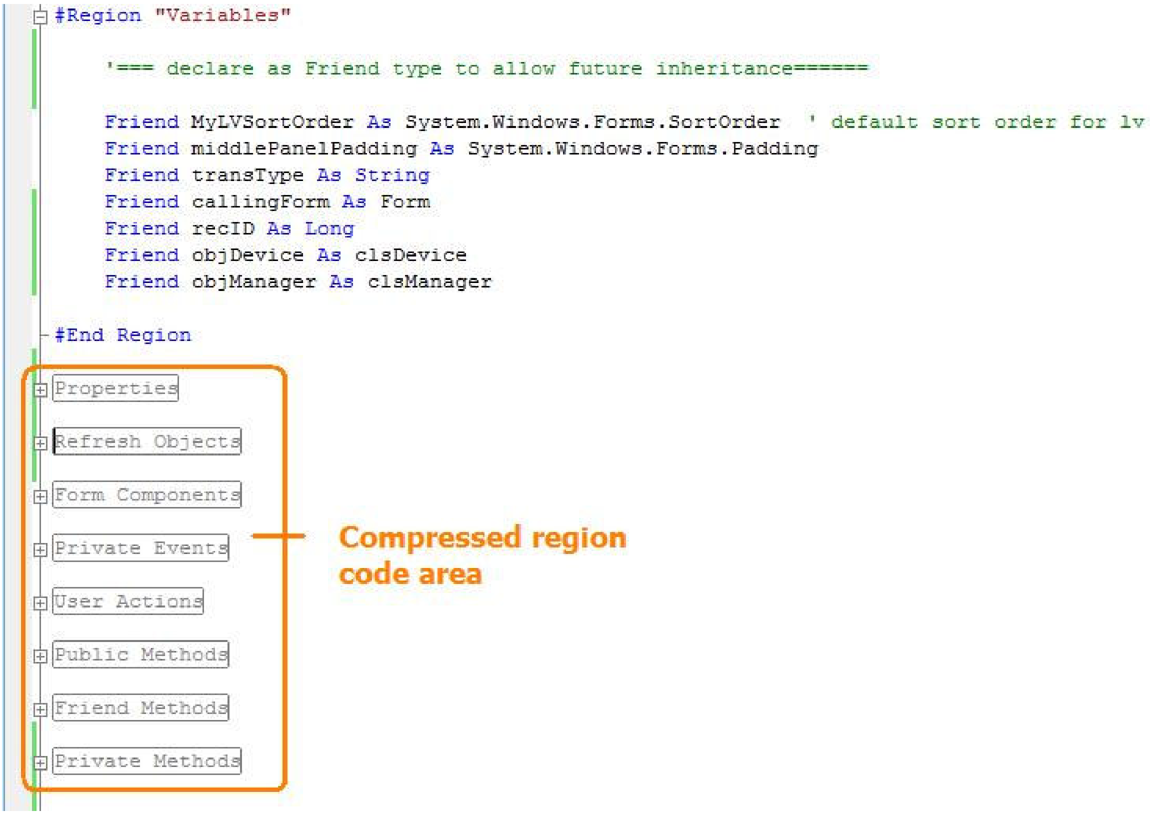
If you use the IDE to change control’s name, it will automatically resolve all dependecies of the codes for you with respect to the change.



DataSource	(none)
DisplayMember	(none)
Items	**(Collection)**
Tag	
ValueMember	
(Name)	**cmbInterlocking**
GenerateMember	True
Locked	False
Modifiers	**Friend**

5.0 Using Region to hide your codes

This is a good feature to manage your codes properly.



```
#Region "Variables"

    '=== declare as Friend type to allow future inheritance======

    Friend MyLVSortOrder As System.Windows.Forms.SortOrder  ' default sort order for lv
    Friend middlePanelPadding As System.Windows.Forms.Padding
    Friend transType As String
    Friend callingForm As Form
    Friend recID As Long
    Friend objDevice As clsDevice
    Friend objManager As clsManager

#End Region

Properties

Refresh Objects

Form Components

Private Events

User Actions

Public Methods

Friend Methods

Private Methods
```

6.0 Do Away Direct API Call

In .NET, you can do away with API calls such as Win32 and etc.

Check out the sample for resizing image in VB6 and VB.NET.

The codes for VB.NET is much simpler and straight forward.

Download sample codes from

http://www.it-sideways.com/2009/11/save-and-retrieve-image-from-database.html

7.0 Object Oriented

Fully use overloading and inherits.

8.0 Windows Service

You can now create software running as service easily and do away with SRVANY

Refer to the following sample.

http://www.it-sideways.com/2009/04/windows-service-sample-code-vbnet.html

9.0 GUI

Nicer and better GUI

10.0 Inter-switching with C#.NET and C++.NET

Once your codes are in VB.NET, you can easily switch to C#.NET and C++.NET because you will be using the same frameworks for each of these different languages.

Top 3 Difficulties of Migrating to VB.Net

1.0 Implementing the preferred way.

VB.Net is a new platform and it comes with new best practices such as using inheritance and etc. Often, one would to decide between switching totally to new best practices (which can be time consuming) or stick to old practices. These include:

- OOP – inheritance, override, overlap
- Multi-threading
- ADO.NET
- Exception handling

2.0 Multi-Threaded

Multi-threading is not as easy as 123. You have to take care of.

- Threadsafe
- Thread-concurrency

http://www.it-sideways.com/2009/07/vbnet-systemthreadingreadwritelock.html

http://www.it-sideways.com/2009/08/vbnet-systemthreadinginterlocked.html

http://www.it-sideways.com/2009/08/vbnet-thread-concurrency-for-module.html

3.0 Lower Level Implementation

Doing things like socket programming is harder than when it is in VB6. Because .NET frameworks doesn't come with build-in winsock object. You have to built-it from a lower level point of view.

Refer to the following articles.

http://www.it-sideways.com/2009/06/default-encoding-for-microsoft-winsock.html

Appendix:

Migrates from VBScript Functions to VB.NET

ASP2ASPX can convert VBScript Functions to VB.NET automatically. Please see following table that lists VBScript and VB.NET functions:

VBScript 5.6 Functions	VB.NET Functions or managed code
Abs	Math.Abs
Array	New Object() { }
Asc, AscB, AscW	Asc
Atn	Math.Atan
CBool	CBool
CByte	CByte
CCur	CDec
CDate	CDate
CDbl	CDbl
Chr	Chr
CInt	CInt
CLng	CLng
Cos	Math.Cos
CreateObject	CreateObject or New Instance in .NET
Date	Today
DateAdd	DateAdd
DateDiff	DateDiff
DatePart	DatePart
DateSerial	DateSerial
DateValue	DateValue
Day	Day
Eval	(Does not support)
Exp	Math.Exp
Filter	Filter
Fix	Fix
FormatCurrency	FormatCurrency
FormatDateTime	FormatDateTime
FormatNumber	FormatNumber

GetLocale	Session.LCID
GetObject	GetObject
GetRef	(Does not support)
Hex	Hex
Hour	Hour
InputBox	InputBox
InStr, InstrB	InStr
InStrRev	InStrRev
Int	Int
IsArray	IsArray
IsDate	IsDate
IsEmpty	IsEmpty
IsNull	IsDBNull
IsNumeric	IsNumeric
IsObject	*
Join	Join
LBound	LBound
LCase	LCase
Left, LeftB	Left
Len, LenB	Len
LoadPicture	LoadPicture
Log	Math.Log
LTrim	LTrim
Mid, MidB	Mid
Minute	Minute
Month	Month
MonthName	MonthName
MsgBox	MsgBox
Now	Now
Oct	Oct
Replace	Replace
RGB	RGB
Right, RightB	Right
Rnd	Rnd
Round	Math.Round
RTrim	RTrim
ScriptEngine	*
ScriptEngineBuildVersion	*

ScriptEngineMajorVersion	*
ScriptEngineMinorVersion	*
Second	Second
SetLocale	*
Sgn	Math.Sign
Sin	Math.Sin
Space	Space
Split	Split
Sqr	Math.Sqrt
StrComp	StrComp
String	New String()
StrReverse	StrReverse
Tan	Math.Tan
Time	TimeOfDay
Timer	VB.Timer
TimeSerial	TimeSerial
TimeValue	TimeValue
Trim	Trim
TypeName	TypeName
UBound	UBound
UCase	UCase
VarType	VarType
Weekday	Weekday
WeekdayName	WeekdayName
Year	Year

http://www.netcoole.com/asp2aspx/vbhtml/vbfuncs.htm

Printer’s Management Object Properties

Attributes	MarkingTechnology	ConfigManagerUserConfig
Availability	MaxCopies	CreationClassName
AvailableJobSheets	MaxNumberUp	CurrentCapabilities
AveragePagesPerMinute	MaxSizeSupported	CurrentCharSet
Capabilities	MimeTypesSupported	CurrentLanguage
CapabilityDescriptions	Name	CurrentMimeType
Caption	NaturalLanguagesSupported	CurrentNaturalLanguage
CharSetsSupported	Network	CurrentPaperType
Comment	PaperSizesSupported	Default
ConfigManagerErrorCode	PaperTypesAvailable	DefaultCapabilities
ConfigManagerUserConfig	Parameters	DefaultCopies
CreationClassName	PNPDeviceID	DefaultLanguage
CurrentCapabilities	PortName	DefaultMimeType
CurrentCharSet	PowerManagementCapabilities	DefaultNumberUp
CurrentLanguage	PowerManagementSupported	DefaultPaperType
CurrentMimeType	PrinterPaperNames	DefaultPriority
CurrentNaturalLanguage	PrinterState	Description
CurrentPaperType	PrinterStatus	DetectedErrorState
Default	PrintJobDataType	DeviceID
DefaultCapabilities	PrintProcessor	Direct
DefaultCopies	Priority	DoCompleteFirst
DefaultLanguage	Published	DriverName
DefaultMimeType	Queued	EnableBIDI
DefaultNumberUp	RawOnly	EnableDevQueryPrint
DefaultPaperType	SeparatorFile	ErrorCleared
DefaultPriority	ServerName	ErrorDescription
Description	Shared	ErrorInformation
DetectedErrorState	ShareName	ExtendedDetectedErrorState
DeviceID	SpoolEnabled	ExtendedPrinterStatus
Direct	StartTime	Hidden
DoCompleteFirst	Status	HorizontalResolution
DriverName	StatusInfo	InstallDate
EnableBIDI	SystemCreationClassName	JobCountSinceLastReset
EnableDevQueryPrint	SystemName	KeepPrintedJobs
ErrorCleared	TimeOfLastReset	LanguagesSupported
ErrorDescription	UntilTime	LastErrorCode
ErrorInformation	VerticalResolution	Local
ExtendedDetectedErrorState	WorkOffline	Location
ExtendedPrinterStatus	Attributes	MarkingTechnology
Hidden	Availability	MaxCopies
HorizontalResolution	AvailableJobSheets	MaxNumberUp
InstallDate	AveragePagesPerMinute	MaxSizeSupported
JobCountSinceLastReset	Capabilities	MimeTypesSupported
KeepPrintedJobs	CapabilityDescriptions	Name
LanguagesSupported	Caption	NaturalLanguagesSupported
LastErrorCode	CharSetsSupported	Network
Local	Comment	PaperSizesSupported
Location	ConfigManagerErrorCode	PaperTypesAvailable

Parameters	CurrentNaturalLanguage	PortName
PNPDeviceID	CurrentPaperType	PowerManagementCapabilities
PortName	Default	PowerManagementSupported
PowerManagementCapabilities	DefaultCapabilities	PrinterPaperNames
PowerManagementSupported	DefaultCopies	PrinterState
PrinterPaperNames	DefaultLanguage	PrinterStatus
PrinterState	DefaultMimeType	PrintJobDataType
PrinterStatus	DefaultNumberUp	PrintProcessor
PrintJobDataType	DefaultPaperType	Priority
PrintProcessor	DefaultPriority	Published
Priority	Description	Queued
Published	DetectedErrorState	RawOnly
Queued	DeviceID	SeparatorFile
RawOnly	Direct	ServerName
SeparatorFile	DoCompleteFirst	Shared
ServerName	DriverName	ShareName
Shared	EnableBIDI	SpoolEnabled
ShareName	EnableDevQueryPrint	StartTime
SpoolEnabled	ErrorCleared	Status
StartTime	ErrorDescription	StatusInfo
Status	ErrorInformation	SystemCreationClassName
StatusInfo	ExtendedDetectedErrorState	SystemName
SystemCreationClassName	ExtendedPrinterStatus	TimeOfLastReset
SystemName	Hidden	UntilTime
TimeOfLastReset	HorizontalResolution	VerticalResolution
UntilTime	InstallDate	WorkOffline
VerticalResolution	JobCountSinceLastReset	
WorkOffline	KeepPrintedJobs	
Attributes	LanguagesSupported	
Availability	LastErrorCode	
AvailableJobSheets	Local	
AveragePagesPerMinute	Location	
Capabilities	MarkingTechnology	
CapabilityDescriptions	MaxCopies	
Caption	MaxNumberUp	
CharSetsSupported	MaxSizeSupported	
Comment	MimeTypesSupported	
ConfigManagerErrorCode	Name	
ConfigManagerUserConfig	NaturalLanguagesSupported	
CreationClassName	Network	
CurrentCapabilities	PaperSizesSupported	
CurrentCharSet	PaperTypesAvailable	
CurrentLanguage	Parameters	
CurrentMimeType	PNPDeviceID	