Faculté d'Ingénieurs en Informatique, Multimédia, Systèmes, Télécommunication et Réseaux

Master en Génie Logiciel

VB.NET et ASP.NET

Préparé par Elie MATTA

Exemples:

1. Using ListBox:

```
Option Strict Off

Public Class Form1

    Sub incrementer()
        Static number As Integer = 0 'si on met Dim on aura comme resultat 0'
        Console.WriteLine(number)
        number += 1
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim V() As Integer
        V = New Integer(5) {0, 1, 2, 3, 4, 5}
        ReDim Preserve V(7)
        V(6) = 6
        V(7) = 7
        Dim c(,) As Integer = {{1, 2}, {3, 4}}

        For i As Integer = 0 To 5
            ListBox1.Items.Add(V(i).ToString)
        Next i

        For i As Integer = 0 To 7
            ListBox1.Items.Add(V(i).ToString)
        Next i

        For i As Integer = 0 To 1
            For j As Integer = 0 To 1
                ListBox1.Items.Add(c(i, j).ToString)
                'ou messagebox.show'
            Next j
        Next i

    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button3.Click
        For i As Integer = 1 To 5
            incrementer()
        Next i
    End Sub
End Class
```

2. Using Modules and creating classes:

In Form1:

```
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

    End Sub

    Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TextBox1.TextChanged

    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim add As Customer.FullAddress
        Dim cust As Customer
        cust = New Customer()
        add = New Customer.FullAddress()
        add.street = TextBox1.Text
        cust.name = TextBox2.Text
        cust.address = add

    End Sub

    Private Sub TextBox2_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles TextBox2.TextChanged

    End Sub
End Class
```

In Module1:

```
Module Module1
    Public Class Customer
        Class FullAddress
            Public street As String
        End Class
        Public name As String
        Public address As FullAddress
    End Class
End Module
```

3. Adding Values of a table using InputBox:

```
Private Sub Button1_Click_1(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim V(4) As Integer
        For i As Integer = 0 To V.Length - 1
            V(i) = CInt(InputBox("Entrer votre valeur", "Test"))
        Next
        MessageBox.Show(add(V))
    End Sub

    Public Function add(ByVal ParamArray V() As Integer) As Integer
        Dim R As Integer = 0
        For i As Integer = 0 To V.Length - 1
            R += V(i)
        Next
        Return R
    End Function
```

4. Get and Set:

```
    Class class1
        Private _Poids As Integer
        Public Property poids() As Integer
            Get
                Return _Poids
            End Get
            Set(ByVal value As Integer)
                If (Value < 20 Or Value > 90) Then
                    MessageBox.Show("Pas admise")
                Else
                    _Poids = value
                    MessageBox.Show("Admise")
                End If
            End Set
        End Property
    End Class



    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim c As New class1
        Dim a As String
        a = TextBox1.Text
        c.poids = a
    End Sub
End Class
```

5. Creating namespace:

We should put Imports before public class Form1

```
Imports ns1 = WindowsApplication1.topwa

Namespace top
    Public Class inside
    End Class
    Public Class sameclass
    End Class
End Namespace

Namespace topwa
    Public Class inside
    End Class
    Public Class sameclass
    End Class
End Namespace
```

In the button

```
        Dim cls1 As New top.inside() 'using namespace without alias
        Dim cls2 As New top.sameclass()

        Dim cls3 As New ns1.inside()'using namespace with alias
        Dim cls4 As New ns1.sameclass()
```

6. Example with Delegate

```
Imports ns1 = WindowsApplication1.topwa
Namespace topwa
Public Class inside
        Public Sub s1(ByVal x As Int16)
            MessageBox.Show("S1:" & x)
        End Sub
        Public Sub s2(ByVal y As Int16)
            MessageBox.Show("S2:" & y)
        End Sub
        Public Sub s3(ByVal z As Int16)
            MessageBox.Show("S3:" & z)
        End Sub
    End Class
End Namespace
```

In the button :

```
Dim x, y, z As String
        Dim s1, s2, s3 As dl
        x = TextBox1.Text
        y = TextBox2.Text
        z = TextBox3.Text
        Dim cls As New ns1.inside()
```

```
        s1 = New dl(AddressOf cls.s1)
        's1(x)
        s2 = New dl(AddressOf cls.s2)
        's2(y)
        s3 = [Delegate].Combine(s1, s2)
        s3(z)
```

In the Form:

```
Delegate Sub dl(ByVal x As Int16)
```

Exercices:

Exercice 1:

En utilisant la commande InputBox saisr une valeur i de type integer et puis effectue les test suivant :

- i<0
- 0<= i <= 10
- 11<= i <= 20
- i>20

En utilisant une structure if et une structure select case.
Affiche pour chaque cas un message indiquant le resultat

```
   Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim a As Integer
        a = InputBox("Test value", "Using select case")
        Select Case a
            Case Is < 0
                MessageBox.Show("a<0")

            Case Is <= 10
                MessageBox.Show("a between 0 and 10")

            Case Is <= 20
                MessageBox.Show("a between 11 and 20")

            Case Is > 20
                MessageBox.Show("a plus grand que 20")
        End Select
```

Exercices 2 :

1) Ecrire le code VB qui saisie les valeurs d'un vecteur A(5) en utilisant une boucle et le control inputbox et par suite additionner ce vecteur et afficher son résultat, on considère que c'est un vecteur d'entier. (En utilisant deux boucles : For et While)

Première Méthode :

```
'Dim R As Integer = 0
      'Dim V(4) As Integer
      'Dim i As Integer = 0
      'While (i < V.Length - 1)
      '    V(i) = CInt(InputBox("Entrer votre valeur", "Test"))
      '    R += V(i)
      '    i += 1
      'End While
      'MessageBox.Show(R)
```

Deuxième Méthode :

```
        Dim R As Integer = 0
        Dim V(4) As Integer
        For i As Integer = 0 To V.Length - 1
            V(i) = CInt(InputBox("Entrer votre valeur", "Test"))
            R += V(i)
        Next
        MessageBox.Show(R)
```

Troisieme Methode:

```
Dim x(5) as Integer
Dim s As Integer=0
For i as Integer=0 to 5
x(i)=CInt(inputbox("enrewe:"))
s+=x(i)
End for
MessageBox.show("somme="&s.ToString)
```

Quatrieme methode:

```
Dim x(5) as Integer
Dim s As Integer=0
Dim i as Integer=0
While i<=5
x(i)=CInt(InputBox("entrer.."))
s+= x(i)
i+=1
End While
MessageBox.Show(« somme est »&s)
```

2) Réécrira le même exercice en utilisant une classe qui contient deux fonctions qui s'appellent somme_for et somme_while

```
Public Class Class1
    Public Function sum_for(ByVal t() As Integer) As Integer
        Dim s As Integer = 0
        For i As Integer = 0 To t.Length - 1
            s += t(i)
        Next i
        Return s
    End Function

    Public Function sum_while(ByVal t() As Integer) As Integer
        Dim s As Integer = 0
        Dim i As Integer = 0
        While i < t.Length
            s += t(i)
            i += 1
        End While
        Return s
    End Function
End Class


    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Dim k(5) As Integer
        For i As Integer = 0 To k.Length - 1
            k(i) = CInt(InputBox("entre l'elemnet numero: " & i + 1))
        Next i
        Dim c As New Class1

        Dim somme As Integer = c.sum_for(k)
        MessageBox.Show("somme par methode for: " & somme)

        somme = c.sum_while(k)
        MessageBox.Show("somme par methode while: " & somme)
    End Sub
```

Exercice 3 :

En utilisant inputbox, upperbound(dimension courante) et ReDim preserve. Ecrire le code qui contient un bouton avec le code correspondant qui a chaque fois qu'on clique sur ce bouton affiche un inputbox

NB : int i = UBOUND(A)

```
    Dim A(0) As Int16
    Dim v As Int16
    Dim i As Int16


    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button3.Click
        v = InputBox("enter value")
        i = UBound(A)
        A(i) = v
        ReDim Preserve A(i + 1)
MessageBox.Show("La grandeur du vecteur A maintenant est " & i)
    End Sub
```

Exercice 4:

Form1:

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

    End Sub

    Public Structure customer
        Dim ID As Integer
        Dim FN As String
        Dim LN As String
        Dim DOB As Date
    End Structure

    Public acustomer(0) As customer

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim cust As customer
        cust.FN = TextBox1.Text
        cust.LN = TextBox2.Text
        cust.DOB = CDate(TextBox3.Text)
        cust.ID = UBound(acustomer)
        acustomer(UBound(acustomer)) = cust
        ReDim Preserve acustomer(UBound(acustomer) + 1)
        MessageBox.Show("Customer " & acustomer(cust.ID).FN.ToString & " has been added
SUCCESSFULLY!!!!")
        TextBox1.Clear()
        TextBox2.Clear()
        TextBox3.Clear()
```

```
    End Sub

    Public Function Rcust(ByVal ID As Integer) As customer
        If ID < acustomer.Length - 1 Then
            Return acustomer(ID)
        Else
            MessageBox.Show("ID INVALID")
            End
        End If

        ''Try
        ''    Return acustomer(ID)
        ''Catch ex As IndexOutOfRangeException
        ''    MessageBox.Show("Not included" & ex.Message.ToString)
        ''End Try
    End Function

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Form2.Show()

    End Sub
End Class
```

Form2:

```
Public Class Form2

    Private Sub Form2_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

    End Sub


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim cust1 As Form1.customer
        cust1 = Form1.Rcust(CInt(TextBox1.Text))

        ListBox1.Items.Add(cust1.ID.ToString)
        ListBox1.Items.Add(cust1.FN.ToString)
        ListBox1.Items.Add(cust1.LN.ToString)
        ListBox1.Items.Add(cust1.DOB.ToString)

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        ListBox1.Items.Clear()
    End Sub
End Class
```

Exercice 5 :

En utilisant cette classe ecrire le code qui cree une classe guest qui herite de la classe user et parsuite essayer d'utiliser les attributs ssn et pass dans la classe guest

```
  Class user
        Private ssn As String
        Protected pass As String
    End Class

    Class guest
        Inherits user
        Function f() As String
            pass = "ABCD" 'pass=works car c'est protected
            'ssn=doesnt work car c'est private
            Return pass
        End Function
    End Class
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim g As New guest
        Dim r As String
        r = g.f() 'on peut seulment acceder a la fonction dans la classe qui herite, on ne
peut pas accede a ces variables
        MessageBox.Show(r)
    End Sub
```

Exercice 6:

En utilisant une instance sur la classe guest et tout en modifiant les 2 classes guest et user tout en les ajoutant des constructeur proposez une methode pour initialiser ssn et pass et ceci sans changer l'accessibilite de ces 2 variables

```
    Class user
        Private ssn As String
        Protected pass As String

        Public Sub New(ByVal s As String, ByVal p As String)
            s = ssn
            p = pass
        End Sub
    End Class

    Class guest
        Inherits user
        Private ssn2 As String
        Sub New(ByVal s2 As String, ByVal p2 As String)
            MyBase.New(s2, p2) 'appel du constructeur de la classe parente
            ssn2 = s2
        End Sub
        Function f() As String
            Return ssn2 & pass
        End Function
    End Class
```

```
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Dim a As New guest("test", "password")
        Dim b As String
        b = a.f()
        MessageBox.Show(b)
    End Sub
End Class
```

Exercice 7

Modifier le code précédent tout en remplaçant les constructeurs par des fonctions

```
  Class user
        Private ssn As String
        Protected pass As String

        Sub s2(ByVal s As String, ByVal p As String)
            s = ssn
            p = pass
        End Sub
    End Class

    Class guest
        Inherits user
        Private ssn2 As String
        Sub s1(ByVal s2 As String, ByVal p2 As String)
            MyBase.s2(s2, p2) 'appel du constructeur de la classe parente
            ssn2 = s2
        End Sub
        Function f() As String
            Return ssn2 & pass
        End Function
    End Class

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Dim g As New guest()
        g.s1("aa", "bb")
        Dim r As String
        r = g.f()
        MessageBox.Show(r)
    End Sub
End Class
```

Exercices 8 :

Dans la solution précédent essayez de remplaces l'accesseur private dans la classe user par public ReadOnly ssn as string et commenter

Lorsqu'on déclare une variable readonly on peut seulement l'initialiser à l'intérieur d'un constructeur

Exercices 9 :

Reecriver la classe guest tout en considérant la visibilité de la question 8

```
    Class user
        Public ReadOnly ssn As String 'on ne peut pas la changer sauf si elle etait dans le
constructeur
        Protected pass As String


        Sub New(ByVal s As String, ByVal p As String)
            s = ssn
            p = pass
        End Sub
    End Class

    Class guest
        Inherits user
        Private ssn2 As String
        Public pass2 As String
        Sub New(ByVal s As String, ByVal p As String)
            MyBase.new(s, p)
            pass2 = p
        End Sub
    End Class

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        Dim g As New guest("12", "4")

        MessageBox.Show(g.ssn & g.pass2)
    End Sub
```

Avec l'utilisation du modificateur d'accès ReadOnly il est préférable que son modificateur soit de type prive :

```
Class user
        Private ReadOnly ssn As String 'on ne peut pas la changer sauf si elle etait dans le
constructeur
        Protected pass As String

        Sub s2(ByVal p As String)
            's = ssn code qui accede a la base de donne et incremente
            p = pass
        End Sub
    End Class

Rappel:
Dim p As New person
        Dim e As New Employee
        p = CType(e, p) 'parent=child
```

Exercices 10: Primary Key, Package et Sous Package, Overrides, Overridable, Shared Solution

Connection.vb

```
Imports System.Data.SqlClient
Imports System.Xml
Imports System.Data.OleDb



Public Class Connexion
    Dim s As String = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=|DataDirectory|\ADO.NET.accdb"

    Dim conn As OleDbConnection
    Dim mycommand As OleDbCommand

    Public Sub dispose()
        conn.Close()
    End Sub

    Public Sub usebuilder(ByVal ds As DataSet, ByVal adp As OleDbDataAdapter)
        Dim builder As New OleDbCommandBuilder(adp)
        adp.Update(ds)
    End Sub

End Class
```

Form1

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

    End Sub

    Class Package
        Protected pid As Int16 'protected car pid doit etre incrementer dans le constructeur
        Public pname As String
        Public is_sp As Boolean
        Public Shared packs As New ArrayList 'on va creer plusieurs instances c'est pour
quoi on a mis shared


        Sub New()
            pid = packs.Count 'pour incrementer le pid automatiquement
        End Sub

        Public Function Create_Package(ByVal name As String, ByVal is_sp As Boolean) As
Int16 'pour retrouner le pid
            Me.pname = name
            Me.is_sp = is_sp
            If is_sp = False Then
                packs.Add(Me)
            Else
                Dim p As New Package
```

```
                p.pname = name
                p.is_sp = is_sp
                packs.Add(p)
            End If
        End Function

        Public Overridable Sub GetDetails(ByVal id As Int16)
            For i As Integer = 0 To packs.Count - 1
                Dim pkg As Package = packs(i) 'ctype(packs(i),package)
                If pkg.pid = id Then
                    MsgBox(pkg.pname.ToString() & " - " & pkg.is_sp.ToString())
                    Exit For
                End If
            Next i
        End Sub

        Public Overridable Sub delete_pkg(ByVal id As Int16)
            packs.RemoveAt(id)
        End Sub
    End Class

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click

        Dim cls1 As New Package
        Dim id As Int16 = cls1.Create_Package("premiere", False)
        MessageBox.Show(id.ToString())

        Dim spcls As New sppackage
        id = spcls.Create_Package("deuxieme", True, "ok")
        MessageBox.Show(id.ToString())
        cls1.delete_pkg(1)
        cls1.GetDetails(0)
        cls1.GetDetails(1)
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click

    End Sub
End Class

Class sppackage
    Inherits Form1.Package
    Public status As String 'car on veut status
    Public Shared sppacks As New ArrayList


    'class create_package ici est overloads car on a fait redefinition de cette fonction et
elle contient 3 arguments
    Public Overloads Function Create_Package(ByVal name As String, ByVal is_sp As Boolean,
ByVal status As String) As Int16 'pour retrouner le pid 'elle utilise le constructeur de la
base clase
        MyBase.Create_Package(name, is_sp) 'appel create_package de la class package
        Me.status = status
        sppacks.Add(Me)
        Return pid
    End Function
```

```
    Public Overrides Sub GetDetails(ByVal id As Int16)
        For i As Integer = 0 To packs.Count - 1
            Dim pkg As sppackage = sppacks(i) 'ctype(packs(i),package)
            If pkg.pid = id Then
                MsgBox(pkg.pname.ToString() & " - " & pkg.is_sp.ToString() & "-" &
pkg.status.ToString())
                Exit For
            End If
        Next i
    End Sub

    Public Overrides Sub delete_pkg(ByVal id As Int16)
        sppacks.RemoveAt(id)
    End Sub

End Class
```

ADO.NET

Exercice 11:

1) Créer une table T1 contenant les colonnes C1, C2, C3 de type integer
2) En utilisant un contrôle datagridview insérer 2 lignes de donnes et utiliser comme entête les colonnes C1, C2 et C3
3) Dans une troisième ligne calculer la somme des deux premières lignes
4) Dans une 4eme ligne, multiplier la dernière ligne par un nombre aléatoire
5) Sauvegarder ce gridview dans la table T1
6) Recharger la table T1 dans un autre gridview
7) Essayer de modifier ce gridview tout en ajoutant une ligne quelconque a la fin de ce grid, commenter

Remarques:

- DataGridView.Columns.Add(« column name », « Column header »)
- Pour accéder à une cellule de datagridview on écrit le code suivant

Datagridview1 (column, rows)

Solution: WindowsApplication7

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load 'Question 1
        DataGridView1.Columns.Add("CN1", "C1")
        DataGridView1.Columns.Add("CN2", "C2")
        DataGridView1.Columns.Add("CN3", "C3")

    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click 'Question 2
        DataGridView1.Rows.Add(1, 2, 20)
        DataGridView1.Rows.Add(5, 4, 8)

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        MessageBox.Show(DataGridView1(2, 1).Value.ToString())
    End Sub

    Private Sub DataGridView1_CellContentClick_1(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles DataGridView1.CellContentClick

    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button3.Click 'Question 3
        If DataGridView1.Rows.Count > 0 Then
            Dim s1 As Int32 = 0
```

```
            Dim s2 As Int32 = 0
            Dim s3 As Int32 = 0

            For i As Integer = 0 To DataGridView1.Rows.Count - 1
                s1 = s1 + (DataGridView1(0, i).Value)
                s2 = s2 + (DataGridView1(1, i).Value)
                s3 = s3 + (DataGridView1(2, i).Value)
            Next
            DataGridView1.Rows.Add(s1, s2, s3)
        End If
    End Sub

    Dim a1 As New Random
    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button4.Click

        Dim v(2) As Integer
        Dim i1, i2, i3 As Int32
        Dim r0 As Integer
        For i As Integer = 0 To DataGridView1.Columns.Count - 1
            'r0 = a1.Next(10, 50)
            If i = 0 Then
                'i1 = a1.Next(10, 50) * DataGridView1(i, 2).Value
                v(0) = a1.Next(10, 50) * DataGridView1(i, 2).Value
            End If
            'r0 = a1.Next(10, 50)
            If i = 1 Then
                'i2 = a1.Next(10, 50) * DataGridView1(i, 2).Value
                v(1) = a1.Next(10, 50) * DataGridView1(i, 2).Value
            End If
            'r0 = a1.Next(10, 50)
            If i = 2 Then
                'i3 = a1.Next(10, 50) * DataGridView1(i, 2).Value
                v(2) = a1.Next(10, 50) * DataGridView1(i, 2).Value
            End If
        Next
        'DataGridView1.Rows.Add(i1, i2, i3)
        DataGridView1.Rows.Add(v(0), v(1), v(2))

    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button5.Click
        Dim r0 As New Random
        Dim r1 As Integer
        'r1 = r0.Next(10, 50)
        r1 = r0.Next()

        MessageBox.Show(r1.ToString())
    End Sub
End Class
```

Exercice 12:

1. Create a DataSet
2. Create a DataTable
    a. Add a primary key
3. Add Data (try to create duplication in the dataset)
4. Modify Data (utiliser Find)
5. Delete Data
    Dr.delete()
    Ds.tables(0).rows.Remove(dr)
        i. Ds.rejectchanges
        ii. Ds.acceptchanges

Solution:

1) dim ds as new DataSet
2) dim tbl as new DataTable("Authors")

```
tbl.columns.add("Auid",System.Type.GetType("System.Int32"))
tbl.columns.add("AuLName",System.Type.GetType("System.String"))
tbl.columns.add("AuFName",System.Type.GetType("System.String"))
ds.Tables.Add(tbl)
2- a. dim pk(0) as DataColumns
pk(0)= tbl.Columns("Auid")
tbl.Primarykey = pk
```

3) dim dr as DataRow

```
dr = ds.Table(0).NewRow 'ou tbl.NewRow
dr(0) = 1
dr(1) = "au_lname1"
dr(2) = "au_fname1"
tbl.Row.Add(dr)
dr= tbl.NewRow
dr(0) = 1
dr(1)= 2
dr(2)=2
tbl.Rows.Add(dr)
```

4) Modify

```
dr= tbl.NewRow
dr(0)= 2
dr(1)= "..."
dr(2)="..."
tbl.Rows.Add(dr)
```

Find the author AuID=1 by using rhe primaryKey
dim dr2 as DataRow
dr2 = tbl.Row.find(1)
dr(1)="New_auLname1"

5) fetch a row
dim dr3 as DataRow
dr3 = tbl.Rows.find(2)
dr3.Delete() ‘i. on peut le restaurer avec reject changes
tbl.Rows.Remove(dr3) ‘ii.

i) use of Acceptchanges
tbl.AcceptChanges//show in dataGrid View

ii) add/modify/delete new row
tbl.RejectChanges
tbl.Show in DataGridView

Components

1. **User control:**

- inherits from a usercontrol class, mais comment?
    A. New item: usercontrol
    B. Create a new class that inherits from usercontrol
    C. Create a class library that inherits from usercontrol
    D. Create a new project : windows librsary control
- Modify used in GUI

2. **Custom control**

- Inherits from control or any existing .NET control (Textbox, label…)
    a) New item: custome control
    b) Create class
    c) Create library
    d) Create windows library control
- Modify used by writing code, we can use GUI

PS: inherited control: inherits from : inherits from any existing usercontrol

We developed the user control (A. New item: usercontrol) in **windowsapplication2**
To develop the user control (B. Create a new class that inherits from usercontrol) we create a new Class and we put `Inherits System.Windows.Forms.UserControl` then we right click on the Class -> View designer and then we add the textbox and the buttons…

WindowsApplication2:
Form1:

```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        UserControl11.TextBox1.Text = "new"
    End Sub
End Class
```

UserControl1

```
Public Class UserControl1
    Public Property TextBoxText() As String
        Get
            Return TextBox1.Text
        End Get
        Set(ByVal value As String)
            TextBox1.Text = value
        End Set
    End Property
```

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        MessageBox.Show(TextBox1.Text)
    End Sub
End Class
```

Exercice 13: sur le Timer avec UserControl WindowsApplication8

UserControl :

```
Public Class UserControl1
    Private Bcolor = Nothing
    Private Fcolor = Nothing

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Timer1.Tick
        Label1.Text = Format(Now, "hh:mm:ss")
    End Sub

    Property ClockBackColor() As Color
        Get
            Return Bcolor
        End Get
        Set(ByVal value As Color)
            Bcolor = value
            Label1.BackColor = Bcolor
        End Set
    End Property

    Property ClockForeColor() As Color
        Get
            Return Fcolor
        End Get
        Set(ByVal value As Color)
            Fcolor = value
            Label1.ForeColor = Fcolor
        End Set
    End Property

    Private Sub UserControl1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub
End Class
```

Exemple 14 : sur le UserControl (Timer/ Enter) WindowsApplication5

Connection.vb

```
Imports System.Data.OleDb
Public Class Connection
    Dim con As OleDbConnection
    Dim mycommand As OleDbCommand
```

```
    Public Sub dispose()
        con.Close()
    End Sub


    Public Function GetReader(ByVal sql As String) As OleDbDataReader
        con = New OleDbConnection
        con.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;data Source=DataBase4.mdb"
        con.Open()
        mycommand = New OleDbCommand
        mycommand.Connection = con
        mycommand.CommandText = sql
        Dim datar As OleDbDataReader
        datar = mycommand.ExecuteReader()
        mycommand.Dispose()
        Return datar
    End Function
End Class
```

Form1.vb

```
Public Class Form1
    Dim WithEvents ctrl As WindowsApplication5.UserControl1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        ctrl = New WindowsApplication5.UserControl1
        ctrl.SetBounds(10, 10, 600, 25)
        ctrl.tva = 12
        Me.Controls.Add(ctrl)
    End Sub

    Public Sub hctrl() Handles ctrl.addControl
        Dim i1 As Integer = ctrl.Location.X
        MessageBox.Show(ctrl.Location.Y.ToString())
        Dim i2 As Integer = ctrl.Location.Y + ctrl.Size.Height
        Dim i3 As Integer = ctrl.Size.Width
        Dim i4 As Integer = ctrl.Size.Height

        ctrl = New WindowsApplication5.UserControl1
        ctrl.SetBounds(10, i2, 600, 25)
        Me.Controls.Add(ctrl)
        ctrl.Focus()
    End Sub
End Class
```

Usercontrol1.vb

```
Imports System.Data.OleDb
Public Class UserControl1

    Public Event addControl()
    Private years, principle, intRate, interest As Double
    Private _tva As Double

    Public Property tva()
        Get
            Return _tva
        End Get
        Set(ByVal value)
```

```
            _tva = value
        End Set
    End Property
    Private Sub TextBox5_KeyDown(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.KeyEventArgs) Handles TextBox5.KeyDown
        If (e.KeyData = Keys.Enter) Then
            RaiseEvent addControl()
        End If
    End Sub

    Private Sub TextBox3_Leave(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles TextBox3.Leave
        principle = Convert.ToDouble(TextBox1.Text)
        intRate = Convert.ToDouble(TextBox2.Text)
        years = Convert.ToDouble(TextBox3.Text)
        interest = principle * years * intRate / 100 - tva
        TextBox4.Text = interest.ToString()
    End Sub

    Private Sub UserControl1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        TextBox5.AutoCompleteMode = AutoCompleteMode.SuggestAppend
        TextBox5.AutoCompleteSource = AutoCompleteSource.CustomSource
        Dim datar As OleDbDataReader
        Dim cls As New Connection
        datar = cls.GetReader("SELECT * FROM customers WHERE custname like '" +
TextBox5.Text + "%'")
        Do Until datar.Read = False
            Dim str As String = datar(1).ToString().Trim()
            TextBox5.AutoCompleteCustomSource.Add(str)
        Loop
        datar.Close()
        cls.dispose()
    End Sub
End Class
```

Exemple 15: sur le WebUserControl WebApplication1

Defaults.apx.cs

```
Partial Public Class _Default
    Inherits System.Web.UI.Page

    Private Sub Logon1_SubmitPressed(ByVal Email As String, ByVal Password As String)
Handles WebUserControl11.SubmitPressed
        Try
            Label1.Text = Email
            Label2.Text = Password
        Catch ex As Exception
            WebUserControl11.displayMessage("No match was found.")
        End Try
    End Sub


End Class
```

WebUserControl

```
Public Partial Class WebUserControl1
    Inherits System.Web.UI.UserControl
```

```
    Public Event SubmitPressed(ByVal Email As String, ByVal Password As String)

    Private Sub btnSubmit_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        lblNotFound.Visible = False
        RaiseEvent SubmitPressed(TextBox1.Text, TextBox2.Text)
    End Sub

    Public Sub displayMessage(ByVal Message As String)
        lblNotFound.Text = Message
        lblNotFound.Visible = True
    End Sub
```

Exemple 16: avec ClassLirary(CustomeControl) et timer ClassLibrary2 et WindowsApplication12

Dans WindowsApplication12 (Form)

```
Public Class Form1

    Protected WithEvents tmr As ClassLibrary2.CustomControl1 'on a cree une instance

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        tmr = New ClassLibrary2.CustomControl1()
        tmr.input = TextBox1.Text
        tmr.enabled = True
    End Sub

    Private Sub f1(ByVal s As String) Handles tmr.finished
        MessageBox.Show("4 secondes elapsed 1" & " " & s)
    End Sub

    Private Sub f2(ByVal s As String) Handles tmr.finished
        MessageBox.Show("4 secondes elapsed 2" & s)
    End Sub



    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        tmr.Dispose()
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
        MessageBox.Show(TextBox1.Text)
    End Sub



    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
        TextBox1.Undo()
    End Sub
```

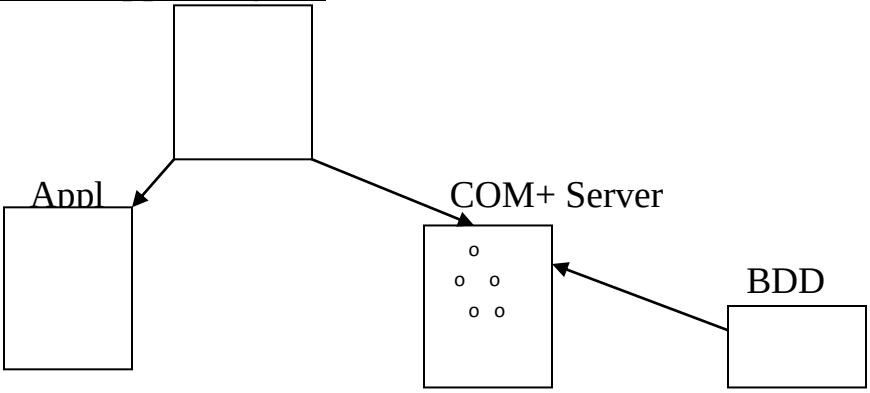
```
    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button5.Click
        CustomControl11.enabled = True
    End Sub


End Class
```

Dans ClassLibrary2 (CustomControl1.vb)

```
Public Class CustomControl1
    Inherits System.ComponentModel.Component

    Public input As String
    Public Event finished(ByVal str As String)

    Private WithEvents localtimer As System.Timers.Timer

    Public Property enabled() As Boolean
        Get
            Return localtimer.Enabled
        End Get
        Set(ByVal value As Boolean)
            localtimer.Enabled = value
        End Set
    End Property
    Private Sub localtimer_tick(ByVal s As Object, ByVal e As Timers.ElapsedEventArgs)
Handles localtimer.Elapsed
        RaiseEvent finished(input)
    End Sub
    Public Overloads Sub dispose()
        MyBase.Finalize()
        localtimer.Enabled = False
        localtimer.Dispose()
        MyBase.Dispose()

    End Sub
    Public Sub New()
        MyBase.new()
        InitializeComponent()
        localtimer = New SYstem.Timers.Timer
        localtimer.Enabled = False
        localtimer.Interval = 4000
    End Sub

End Class
```

.NET COM+

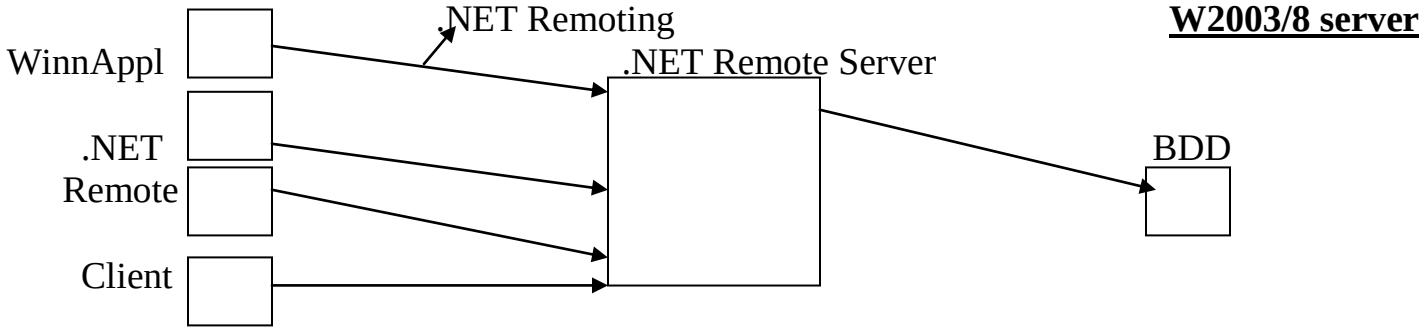
1) Server Appl + objects



2)

a) Windows
b) Web Server
c) Remote server

Exemple:



3) COM+ Features

a. Pooling
b. Transaction
c. Security
d. Manage ability
e. JIT (Just in time)(free the memory between methods calls -> scalability)

4) How to work with COM+ .NET

a. make the .NET assembly COM visible
b. Sign the .NET assembly
c. The .NET class must be created by using interface so it could be exposed with the methods to COM (unmanaged application in VS6) and to serviced components in .NET (COM+)
d. Use regsvcs.exe to register the .NET COM into the COM+ server and to create the .tlb proxyfile

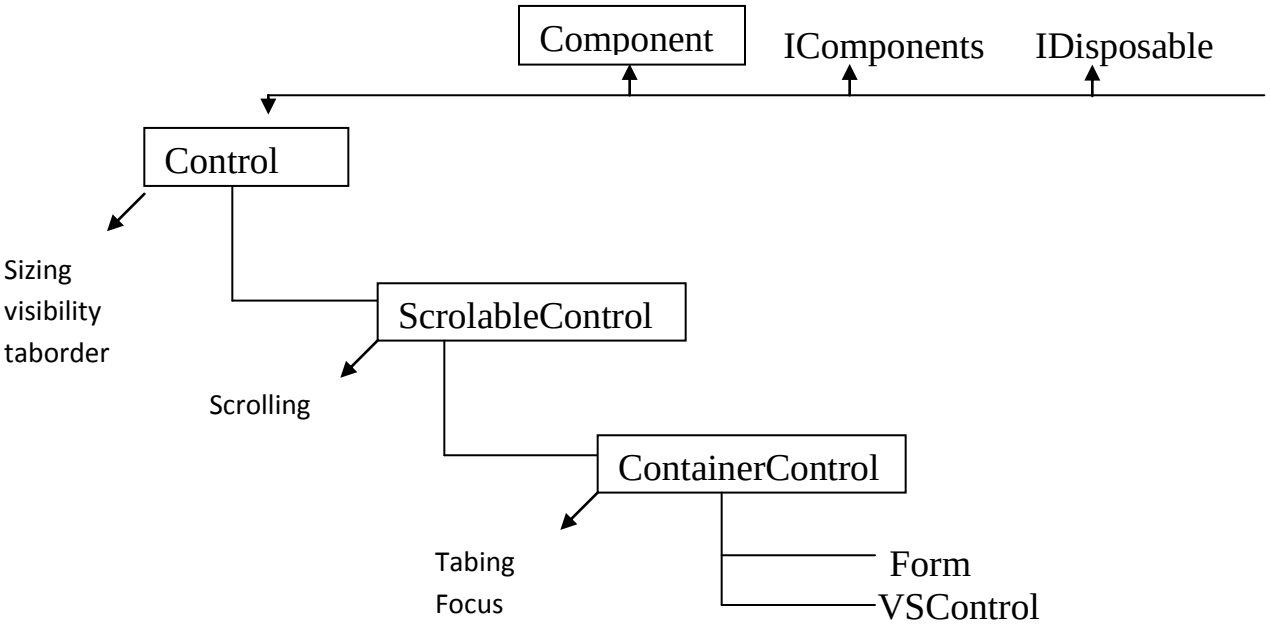
NOTE: Objects can only be returned to the pool when they are deactivated

Windows Forms

I. Why use windows form?

1. Rich of controls
2. Advanced printing support :
    a. Page setup dialog
    b. Print preview control
    c. Print preview dialog
    d. Print dialog
3. Advanced GDI+
    a. System.Drawing.Drawing2D
    b. System.Drawing.Imaging
    c. System.Drawing.Text
4. Visual inheritance a : WForm are classes -> we can benefit from inheritance
5. Extensible object model : We can extend the class library
6. Advanced forms design

II.



III. Application class:

.startuppath
.exit
.run

Methode 1: ConsoleApplication1

```
Module Module1

    Sub Main()
        Dim frm As New Form1
        Windows.Forms.application.run(frm)
    End Sub

End Module
```

Le code après celui la ne sera pas exécute car on a utilise .run(frm)

Methode 2 : ConsoleApplication1

Pour exécuter le code qui suit on utilise le code suivant :

```
   Dim frm As New Form1
        frm.Show()
        Windows.Forms.Application.Run()
        MsgBox("un")
        Dim frm2 As New form2
        frm2.show()
        Windows.Forms.Application.Run()
        MsgBox("deux")
        MsgBox("trois")
```

Mais on doit fermer l'application dans formClosed quand on utilise .Run():

```
Private Sub Form1_FormClosed(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosedEventArgs) Handles MyBase.FormClosed
        Windows.Forms.Application.Exit()
    End Sub
Public Class Form2

    Private Sub Form2_FormClosed(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosedEventArgs) Handles MyBase.FormClosed
        Windows.Forms.Application.Exit()
    End Sub
End Class
```

Dans ce cas les deux forms et les 3 msgbox seront affichés.

IV. Code Behind

Imports
Class
Sub new()
End Sub
InitializeComponent() : C'est le code créé par vb lorsqu'on fait drag drop

Destructeur : automatiquement appelle lorsqu'on ferme le formulaire
Non-Modal: the destructor is auto called when we close the form
Modal: we have to call the Dispose() manually

V. Form Proprieties

-Dialog Result
Button 17 dans le cours
-Opacity and focus
-AcceptButton and CancelButton

VI. Form Methods

-Close
-Show
-ShowDialog

VII. Form Events

-Activated, Deactivate

Exercices:

a. En utilisant deux méthodes différentes écrire le code correspondant pour manipuler les événements suivants : Form1_Activated, Form1_Closed, Form1_Desactivate, Form1_SizeChanged
b. En utilisant un formulaire modal traité les cas Ok, Cancel, Abort et Retry

Solution : WindowsApplication14

Form1:

```
Public Class Form1

    Private Sub Form1_Activated(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Activated
        TextBox1.Text = "Activated"

    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        AddHandler Me.FormClosed, AddressOf MyForm_ClickClosed

    End Sub

    Private Sub Form1_FormClosed(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosedEventArgs) Handles MyBase.FormClosed
        MsgBox("Closed")

    End Sub
```

```
    Private Sub Form1_Deactivate(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Deactivate
        TextBox1.Text = "Deactivated"

    End Sub

    Private Sub Form1_SizeChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.SizeChanged
        TextBox1.Text = "Size changed"
    End Sub

    Private Sub MyForm_ClickClosed(ByVal semder As Object, ByVal e As EventArgs)
        MsgBox("Closed handler")

    End Sub


    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        Dim frm2 As New Form2

        frm2.ShowDialog()
        If frm2.DialogResult = Windows.Forms.DialogResult.OK Then
            MsgBox("You have pressed ok")
        ElseIf frm2.DialogResult = Windows.Forms.DialogResult.Cancel Then
            MsgBox("You have pressed cancel")
        ElseIf frm2.DialogResult = Windows.Forms.DialogResult.Abort Then
            MsgBox("You have pressed Abort")
        ElseIf frm2.DialogResult = Windows.Forms.DialogResult.Retry Then
            MsgBox("You have pressed Retry")
        End If


    End Sub
End Class
```

Form2:

```
Private Sub Form2_FormClosed(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.FormClosedEventArgs) Handles MyBase.FormClosed
        Me.Dispose()
    End Sub
```

Exercies: WindowsApplication10CopyetMove

```
Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load
        Dim mnumain As New MainMenu

        Dim menuitem1 As New MenuItem

        menuitem1.Text = "File"
        mnumain.MenuItems.Add(menuitem1)

        Dim submenuitem11 As New MenuItem
        submenuitem11.Text = "Close"
        mnumain.MenuItems(0).MenuItems.Add(submenuitem11)

        Me.ListBox1.Items.Add("One")
        Me.ListBox1.Items.Add("Two")
        Me.ListBox1.Items.Add("Three")
        Me.ListBox1.Items.Add("Four")
        Me.ListBox1.Items.Add("Five")

        Me.TextBox1.AllowDrop = True
        Me.TextBox2.AllowDrop = True

        Me.Menu = mnumain
        AddHandler submenuitem11.Click, AddressOf exithandler
    End Sub

    Sub exithandler(ByVal sender As Object, ByVal e As EventArgs)
        If (MessageBox.Show("Are you sure that you want to exit ?", "Close",
MessageBoxButtons.YesNo) = Windows.Forms.DialogResult.Yes) Then
            Me.Close()
        End If
    End Sub

    Private Sub ListBox1_MouseDown(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles ListBox1.MouseDown
        Dim dragdropresult As DragDropEffects
        dragdropresult = Me.ListBox1.DoDragDrop(ListBox1.Text, DragDropEffects.All)
        If (dragdropresult = DragDropEffects.Move) Then
            Me.ListBox1.Items.RemoveAt(Me.ListBox1.SelectedIndex)
        End If
    End Sub

    Private Sub TextBox1_DragEnter(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.DragEventArgs) Handles TextBox1.DragEnter
        If e.Data.GetDataPresent(DataFormats.Text) Then
            e.Effect = DragDropEffects.Copy
        End If
    End Sub

    Private Sub TextBox1_DragDrop(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.DragEventArgs) Handles TextBox1.DragDrop
        Dim str As String = e.Data.GetData(DataFormats.Text).ToString()
        'Me.TextBox1.SelectedText = str + vbCrLf
        Me.TextBox1.SelectedText = str
```

```
    End Sub

    Private Sub TextBox2_DragEnter(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.DragEventArgs) Handles TextBox2.DragEnter
        If e.Data.GetDataPresent(DataFormats.Text) Then
            e.Effect = DragDropEffects.Move
        End If
    End Sub

    Private Sub TextBox2_DragDrop(ByVal sender As System.Object, ByVal e As
System.Windows.Forms.DragEventArgs) Handles TextBox2.DragDrop
        Dim str As String = e.Data.GetData(DataFormats.Text).ToString()
        Me.TextBox2.SelectedText = str + vbCrLf
    End Sub
End Class
```