VB.Net & C#.Net

Project 405

Name : Mohammed Ahmed Reyad Mahran

NickNameNew : Opreyad36333

NickNameOld : OpMrayed20953

Email : IT_M.Reyad@yahoo.com

2015/12/10

مقدمة

هذا الكتاب تم عمله لانه من احدى المشروعات المطلوبة منى من خلال أكاديمية المجموعة العربية للكمبيوتر.

واشكر هذة أكاديمية المجموعة للكمبيوتر على مستوى الذى وصلت اليه

المراجع

موقع ميكروسوفت

اكاديمية المجموعة العربية للكمبيوتر

اكاديمية بى سى لاب

الخاتمة

اهداء وشكر

اكاديمية المجموعة للكمبيوتر على مستوى الذى وصلت اليه

وخاصة الدكتور / عمرو موسى

والمهندس / احمد سمير

والعاملين بالفرع القبة

محتويات

C#.NET	VB.NET

الفصل الاول : مقدمة عن اللغتين :

أوجه التشابة :

هما من احدى اللغات التى تطورتها ميكروسوفت والاتنين من اللغات الموجودة داخل ViualStudio.Net و(المقطع دوت نت) يوضح لنا انهم يعتمدوا على اطار العمل NetFrameWork. والذى يعتبر بحر عميق يحتوى عى ثروة هائلة من الادوات والكلاسيس

مميزاتها : هى لغة بسيطة جدا وسهلة للتعلم تدعم للكائنات البرمجية هى اقرب ما تكون الى اللغة الانجليزية العادية ولاتحتوى على الرموز الكثيرة المملة التى تملا سى شاري وجى شارب مثل ++ و – و == و--- ----الخ التى تجعل احتمالات الخطأ اعلى عند كتابة الكود	**مميزاتها :** هى تعتبر تواما لفيجوال بيسيك الا انها تستخدم قواعد C++ فى كتابة الاوامر هى لغة قوية وسهلة التعلم وبرامجها سريعة التطوير تدعم البرمجة الكائنية تعتمد على مكتبات اطار الـ NET. مما يسهل عملية كتابة البرامج المعقدة دون مصادر خارجية.
عيوبها : لها حدود يجب ان تقف عندها فلا يمكن ان تصمم بها نظام تشغيل لان اوامرها لا تتعامل مع الالة بشكل مباشر لذا فهى بطيئة نسبيا عدم دعمها لكل اوجه البرمجة الكائنية	**عيوبها :** لاتعمل الا على بيئة الوينوز يعتبرها الكثيرون مجرد تقليد للغة الجافا

C#.NET	VB.NET

الفصل التانى : Data Structure:

المتغيرات Variables:

- نستعمل المتغيرات لكى نخزن بعض البيانات فى الذاكرة لنستعملها اثناء تنفيذ البرنامج.
- وكل متغير له اسم يميزه عن الباقى.
- وكل متغير له نوعا Data Type لتحديد طبيعة القيمة المراد تخزينها.
- طريقة الاعلان عن المتغيرات بحيث لا يقبل ان تبدأ اسماء المتغيرات بأرقام او رموز مع استثناءات قليلة او ان تكون اسماء المتغيرات منتمية الى الكلمات المحجوزة.

هى لاتهتم بحالة الاحرف	تهتم بحالة الاحرف
Dim X as String Dim X as String = "M"	string X ; string X ="M" ;

- يليه الاعلان عن متغير للقراءه فقط لايمكن تعديل قيمته ولكن ممكن اعطيه قيمه بعد انشاءه ليس شرطا نفس المكان

```vb
Public ReadOnly PI As Double = 3.14
```

```cs
public readonly double PI = 3.14;
```

الثوابت Constants:

- هى شبيهة جدا بالمتغيرات من حيث الدور المنوط بها.
- ولكنها تختلف فى كونها قيمة الثابت لاتتغير ابدا عند تنفيذ البرنامج وتبقى ثابتة دائما.

```vb
Public Const myNumber As Integer = 100
```

```cs
public const int myNumber = 100;
```

التعليقات Comments:

- هى عبارات نقوم بكتابتها فى برامجنا ويتجاهلها المترجم.
- هى دورها يكون فقط من اجل عنونة الكود لتسهيل قراءته او لتدوين بعض الملاحظات عليه من قبل المبرمج.

لسطر الواحد

```vb
' Dim X as String
```

لعده اسطر

```vb
' Dim X as String
' Dim y as String = "M"
```

لسطر الواحد

```cs
// public const int myNumber = 100;
```

لعده اسطر

```cs
/* public const int myNumber = 100;
public readonly double PI = 3.14; */
```

انواع البيانات DataType:

- هى طبيعة القيم التى تريد تخزينها فى المتغيرات او الثوابت.
- الارقام الصحيحة والعشرية والتاريخ والوقت والنصوص والمنطقية وكائن.

Long – Integer – Short – Byte	Long – Int – Short – Byte
Single – Double – Decimal	Float – Double – Decimal
Date	DateTime
String – Char	String – Char
Boolean	Bool
Object	Object

C#.NET	VB.NET

الروابط Operators:

الروابط (العمليات الحسابية او الرياضية) :

- هى الرموز نستخدمها لاجراء بعض العمليات.
- هى الجمع والطرح والضرب والقسمة والقسمة الصحيحة الطبيعية وباقى القسمة والاس

C#.NET	VB.NET
(+) – (-) – (*) - (/) – (/) – (%) – (غير موجودة)	(+) – (-) – (*) - (/) – (\) – (Mod) – (^)

الروابط اعطاء القيم :

(التساوى – الزيادة – النقصان – المضاعفة – الاختزال – الاختزال الصحيح الطبيعى – دمج النصوص)

x : Integer - y : String

C#.NET	VB.NET
X= 10	X= 10
X +=10	X +=10
X -=10	X -=10
X *=10	X *=10
X /=10	X /=10
X /=10	X \=10
y += "is a Student"	y & = "is a Student" او y + = "is a Student"

الروابط المقارنة :

(أكبر من – أكبر من او يساوى – أصغر من – أصغر من أو يساوى – يساوى – لايساوى)

C#.NET	VB.NET
(>) – (>=) – (<) – (<=) – (==) – (!>)	(>) – (>=) – (<) – (<=) – (=) – (<>)

الروابط الشرطية :

توجد منها رابط المعية (و) – رابط الاختيار (أو)

C#.NET	VB.NET
(&&) – (||)	(and) – (or)

الروابط الزيادة والنقصان بواحد :

الزيادة بواحد – النقصان بواحد

C#.NET	VB.NET
(Number++) – (Number--)	(غير موجودة) – (غير موجودة)

البنية الشرطية Flow Control:

هى تستخدم البنيات الشرطية من اجل التحقق من نتيجة شرط معين وبناء على هذا التحقق ننفذ شرط معين دون أخرى

وتوجد طريقتين(IF ..... Else) – (Select Case / Swith Condition) – (المعامل الشرطى)

IF ..... Else

```
if (x == 5
Console.WriteLine("five")
else
Console.WriteLine("notFive")
```

```
If x = 5 Then
Console.WriteLine("five")
Else
Console.WriteLine("notFive")
End If
```

Select Case / Swith Condition

```
switch (x)
{
case 90:
Console.WriteLine("Pass");
break;
case 50
Console.WriteLine("Fail");
};break
```

```
Select Case x
Case 90
Console.WriteLine(" ممتاز")
Case 50
Console.WriteLine(" راسب")
End Select
```

المعامل الشرطى

يستخدم للقيام بعملية من بين عمليتين بعد تحقق شرط معين

```
Dim State As String = IIF(Pwd = "123" , "Hi" , "No"
```

```
String State= Pwd=="123" ? "Hi","No";
```

C#.NET	VB.NET

البنية التكرارية Loops:

- نستخدم لتكرار جزء معين من عدة مرات طول لم يتحقق الشرط.

For ..... Next

```
for (int i = 0; i < 10; i++)
{
Console.WriteLine(i);
}
```

```
For i As Integer = 0 To 9
Console.WriteLine(i)
Next
```

Do ..... Loop

```
int x = 0;
Do
{
Console.WriteLine(x);
x++;
}
While (x < 10)
```

```
Dim x As Integer = 0
Do Until x > 10
Console.WriteLine(x)
X+=1
Loop
```

While ..... End While

```
int x = 0;
while (x < 10)
{
Console.WriteLine(x);
x++;
}
```

```
Dim x As Integer = 0
While x < 10
Console.WriteLine(x)
X+=1
End While
```

For each

```
int[] arr = { 10, 20, 30, 40 };
foreach (int i in arr)
Console.WriteLine(i);
```

```
Dim arr As Integer() = {10, 20, 30, 40}
For Each i As Integer In arr
Console.WriteLine(i)
Next
```

الانتقالات:

- نستخدم لتكرار جزء معين من عدة مرات طول لم يتحقق الشرط.

GoTo : تستخدم الانتقال الى سطر كود اخر	**goto : تستخدم الانتقال الى سطر كود اخر**
Exit : تستخدم للخروج من الشرط مع تنفيذ باقى الكود اللى خارج الشرط	**break : تستخدم للخروج من الشرط مع تنفيذ باقى الكود اللى خارج الشرط**
Countine : تستخدم الخروج من الشرط المحقق مع تنفيذ باقى الشرط وتنفيذ باقى الكود اللى خارج الشرط	**Countine : تستخدم الخروج من الشرط المحقق مع تنفيذ باقى الشرط وتنفيذ باقى الكود اللى خارج الشرط**

الوظائف Method:

- هى مجموعة من الاوامر المجمعة تحت اسم معين وعند النداء عليها بهذا الاسم يتم تنفيذها.
- من الممكن ان تاخذ تمرير البرامترات
- وهى بها حالتين (Sub لاترجع بقيمة ولكنها تنفذ مجموعة من الاوامر) – (Function ترجع بقيمة بعد تنفيذ مجموعة من الاوامر)

Sub

```
void printmsg(string msg)
{
Console.WriteLine(msg);
}
```

```
Private Sub printmsg(ByVal msg As
String)
Console.WriteLine(msg)
End Sub
```

Function

```
Public int sum(int number1, int number2)
{
int total = number1 + number2;
return total;
}
```

```
Public Function sum(ByVal number1 As
Integer, ByVal number2 As Integer) As
Integer
Dim total As Integer = number1 + number2
Return total
End Function
```

C#.NET	VB.NET

المصفوفات Arrays:

- هى مجموعة من المتغيرات التى تحتوى على نفس نوع البيانات

```
int[] intarray = new int[5];
for (int i = 0; i < 5; i++)
Console.WriteLine(intarray[i]);
متعدده الابعاد
int matrix = new int[3, 3];
matrix[1, 2] = 20;
```

```
Dim intarray As Integer() = New Integer(4)
For i As Integer = 0 To 4
Console.WriteLine(intarray(i))
Next
متعدده الابعاد
Dim matrix As Integer = New Integer(2, 2)
matrix(1, 2) = 20
```

التراكيب Structure:

- هى قريب من Class بس على شكل مصغر.
- نستخدم التراكيب من اجل انشاء انواع مركبة قابلة لاحتواء اكثر من نوع بيانات.
- مثلا : تستطيع حفظ معلومات طالب معين (الاسم – العمر – العنوان - .....) فى نفس التركيب
- بدل من الاعلان عن متغيرات متفرقة
- ويمكن التركيب ان يضم كذلك وظائف وخصائص بالاضاف الى الحقول.
- اهم مايميزها انه من الممكن انشاء دوال وفويد او سب والاعتماد على متغيراتها على مستوى الاستراكتشر ككل.

```
struct Car
{
public int carNumber;
public int year;
public string factory;};
Car ahmedcar = new Car();
ahmedcar.carNumber = 1000;
ahmedcar.factory = "Nissan";
ahmedcar.year = 2007;
```

```
Structure Car
Public carNumber As Integer
Public year As Integer
Public factory As String
End Structure
Dim ahmedcar As New Car()
ahmedcar.carNumber = 1000
ahmedcar.factory = "Nissan"
ahmedcar.year = 2007
```

المعدات Enumerations:

- تستخدم اذا كنا نريد تحديد قيم محددة ليتم تخزينها فى متغير ما
- مثلا حينما نريد حفظ ايام الاسبوع فى برنامجنا فنحن نعلم مسبقا ان مجال ايام الاسبوع محدد والقيم معروفة

```
Enum cars As Byte
toyota = 0
nissan = 1
fiat = 2
End Enum
```

```
enum cars : byte
{
toyota = 0,
nissan = 1,
fiat = 2
ليس شرطا تحديد النوع هنا وهنا}
```

معالجة الاستثناءات Handling Exceptions:

- هى اخطاء تحدث عند اشتغال البرنامج اى فى مرحلة التنفيذ
- ويكون ذلك لاسباب عديدة من بينها ادخال قيمة غير مناسبة كمحاولة تخزين قيمة نصية فى متغير رقمى او محاولة القيام بعملية القسمة على صفر او محاولة حذف ملف غير موجود اساسا وغير ذلك
- حينما يحدث استثناء فى البرنامج فانه يتوقف فورا عن الاشتغال لذلك وجب ادارة هذة الاخطاء عبر استخدام Try .....Catch

```
int age;
Console.writeline("Enter your age:");
Try
{
  Age = int.parse(console.Readliine());
}
Catch (Exception err)
{
  Console.writeline(err.Message);
}
Finally
  Console.writeline("press any key to
leave …");
}
Console.ReadKey()
```

```
Dim age as Integer
Console.writeline("Enter your age:")
Try
Age= console.Readliine
Catch ex as Exception
Console.writeline(err.description)
Finally
Console.writeline("press any key to
leave …")
End Try
Console.ReadKey()
```

C#.NET	VB.NET

مجلات الاسماء NameSpace :

- هى انواع نقوم بانشائها من اجل تمثيل شامل لكائن معين.
- وهى شبيهة بالتراكيب

```
using System.Data.SqlClient;
SqlConnection sql1 = new
SqlConnection();
```

```
Imports System.Data.SqlClient
Dim sql1 As SqlConnection = New
SqlConnection()
```

الفئات والكائنات Classes & Objects:

- هى انواع نقوم بانشائها من اجل تمثيل شامل لكائن معين.
- وهى شبيهة بالتراكيب
- اما الكائنات فهى نسخ نقوم بانشائها من فئة معينة لكى نستعملها فى برنامجنا.
- للاعلان عن الفئات فى لغتى الفيجوال بيسيك والسى شارب فاننا نستعمل الكلمة المحجوزة Class.

```
Public Class Car
{
   // code
}
Car object1 = new Car();
```

```
Public Class Car
   ' code
End Class
Dim object1 As New Car
```

This & Me:

- لكى نتمكن من الدخول الى عناصر الفئة الحالية
- ويمكننا الاستغناء عنهما وكتابة اسم العنصر مباشرة (المقصود بالعنصر حقول ووظائف وخصائص الفئة)

```
Public Class Employee
{
   Private String Name;

   Public String Work ()
   {
         Return  "I am a Man"
    }

    Public void initialize (sName As String)
   {
          this.name = sName
          this.Work()
   }
}
```

```
Public Class Employee
   Public Name As String

   Public Function Work () As String
         Return  "I am a Man"
    End Function

    Public sub initialize (sName As String)
          Me.name = sName
          Me.Work()
    End Sub
End Class
```

Access Modifier :

Private تعنى الوصول من داخل الكلاس الحالى او السب فقط	Private تعنى الوصول من داخل الكلاس الحالى او السب فقط
Public تعنى الوصول من اى حته كلاس مشروع حالى او خارجى	Public تعنى الوصول من اى حته كلاس مشروع حالى او خارجى
Protected تعنى الوصول من الكلاس الحاليه او المشتقه لها فقط	Protected تعنى الوصول من الكلاس الحاليه او المشتقه لها فقط
Internal الوصول من الكلاس او المشروع الحالى فقط	Friend الوصول من الكلاس او المشروع الحالى فقط

C#.NET	VB.NET

الوراثة Inheritance:

- هى عملية نقل عناصر فئة معينة تسمى الفئة الرئيسية Main Class الى فئة اخرى او اكثر وتسمى الفئة البنت او الفئة Derived Class .
- مثلا استطيع ان انشئ فئة اسميها Animal ثم اشتق منها فئات اخرى (اسد – نمر - .....) وبمجرد القيام بعملية الوراثة فان عناصر الفئة الام (الفئة Animal) تنتقل الى الفئات البنات.
- للاعلان عن كلاس وهو عموما يتم النداء عليه من اضافه داخل ال Solution Explorer وماهو بالظل الاحمر واسفله خط هو طريقه الوراثه
- تاتى مع الفويد/ السب او الداله وهى تعنى انه يتم النداء على الفويد/ السب او الداله مباشرا دون عمل Object من الكلاس

Class Car:General
{
}
	Public Class Car
	Inherits General
	End Class

Class Car { Public Class Car { }}	Public Class Car Public Sub New End Sub End Class

السطر الاول تاتى تلك الكلمه مصاحبه للكلاس (MustInherit / Abstract) وتعنى انه يجب وراثه تلك الكلاس ولا يمكن نهائيا عمل اوبجكت من الكلاس دا ودا يعنى بالتبعيه ان كل البروبرتى والسب والدوال يطبق عليها ذلك المبدا
وعلى العكس ان اردنا ان لايورث الكلاس نهائيا نستخدم السطر الثانى (SeaLed /NotInheritable)

Abstract X SeaLed	MustInherit X NotInheritable

تعنى ان السب الحالى يمكن استخدام الاسم للسب مع تغير جسم الكود كاملا
وهذا بخلاف ال OverLoad اللذى ياخذ نفس اسم السب ونفس الكود مع اضافات فى جسم الكود او النقصان وهذا يتطلب بالضروره

- اختلاف البراميتر فى السب / الفويد / الداله
- اختلاف ترتيب البراميتر فى حاله تساوى العدد
- تساوى العدد مع اختلاف نوع المتغيرات

Virtual In Drive Class/ `override`	Overridable In Drive Class/ `Overloads Overrides`

الواجهات Interfaces:

- فى لغتى الفيجوال بسيك والسى شارب لايوجد مفهوم الوراثة المتعددة التى تمكننا من وراثة اكثر من فئة مرة واحدة.
- وهى عبارة عن هيكل تقوم الفئات باستعماله بحيث لاتحتوى الواجهات على شفرة معينة وانما تحتوى على هيكل يضم تعريف لوظائف وخصائص وحقول من دون كود.
- وحينما تتم عملية استعمال هذه الواجهة من طرف فئة معينة نقوم باعطاء محتوى لعناصرها.
- تسمى عملية استعمال الواجهات من طرف الفئات بـ Implementation.

```
Public Interface ILocation
{
        void SetLocation (int X);
}
```

```
Public Class MyForm : ILocation
{
        Private int x as ;

        void SetLocation (int X) Implements
ILocation.SetLocation
{
this.x = x;
 }
}
```

```
Public Interface ILocation
        Sub SetLocation (Byval X as Integer)
End Interface
```

```
Public Class MyForm
        Implements ILocation
        Private x as Integer

        Sub SetLocation (Byval X as Integer)
Implements ILocation.SetLocation
Me.x = x
        End sub
End Class
```

C#.NET	VB.NET

المشيدات Destractor & Constructors :

- هى عبارة عن وظيفة خاصة لاتعيد لنا شيئا وليس لها نوع ودروه يأتى فى اعداد الكائنات التى سيتم استنساخها من الفئة كإعطاء قيم بدائية لحقول الكائن بمجرد انشائه.
- يعنى لعمل Constractor للكلاس والكنستراكتور هو اول حدث يتم تنفيذه عند اخذ نسخه من الكلاس او وراثته

```
Public Class Emp
{
        Private string name;
        Private int age ;
Public Emp ()
{
      this.New ("Not found name" , 0);
}
'or
Public Emp (string sName, int sage)
      this.name = sName;
      this.age = sage;
}
```

```
Public Class Emp
        Private name as string
        Private age as integer
Sub New ()
      Me.New ("Not found name" , 0)
End sub
'or
Sub New (Byval sName as string , Byval sage as
integer)
      Me.name = sName
      Me.age = sage
End sub
```

- الكود السالف عمل ال Destractor والدستراكتور ده بيستخدم غالبا لتدمير اوبجكت او الكلاس الحالى او ملفات.
- لانهاء الكائنات ونحتاج احيانا ووضع قيمة "لاشئ" فيها.

```
~ Car ( )
{      }
```

```
protected override void Finalize()
{
     try
     {
          // Cleanup statements...
     }
     finally
     {
          base.Finalize();
     }
}
```

```
Protected Overrides Sub Finalize()
    Dispose(False)
    MyBase.Finalize()
End Sub
```

```
Employee Emp =  New Employee();
Emp = Null;
```

```
Dim Emp As New Employee
Emp = Nothing
```

الخصائص Properties :

- هى عبارة عن وظائف نستخدمها للوصو الى حقول الفئة الحالية من خلال فئات اخرى وكان هذه الحقول معرفة بـ Public.
- هى تتكون من جزءين جزء يسمح بقراءة قيمة الحقل ويسمى Getter
- وجزء يسمح بتعديل القيمة الحقل ويسمى Setter .

```
Class Example
{
  Private Int _count

  Public Int Number()
     {
       Get { Return _count ; }
       Set { _count = value ; }
     }
}


Module Module1
{
  Sub Main()
```

```
Class Example
  Private _count As Integer

  Public Property Number() As Integer
          Get
            Return _count
          End Get
          Set(ByVal value As Integer)
            _count = value
          End Set
  End Property
End Class

Module Module1
```

C#.NET	VB.NET

```
(
        Example e = New Example();
        ' Set property.
        e.Number = 1;
        ' Get property.
        Console.WriteLine(e.Number);
  }
}
```

```
  Sub Main()
        Dim e As Example = New Example()
        ' Set property.
        e.Number = 1
        ' Get property.
        Console.WriteLine(e.Number)
  End Sub
End Module
```

اظهار علبة الرسائل :

- هى عبارة عن وظائف نستخدمها للوصو الى حقول الفئة الحالية من خلال فئات اخرى وكان هذه الحقول معرفة بـ **Public**.

```
MessageBox.Show("Message ", "title" ,
MessageBoxButton.YesNo, MessageBoxImage)
```

```
MsgBox("Message",
MsgbosStyle.Information , "title")
```