المحاضرة الأولي

أنواع الاجراءات

- يعود بقيمة (دالة function).
- لا يعود بقيمة (روتين فرعي subroutine).

اجراءات لا تعود بقيمة

```
Public/private/...  Sub  subname([parameters])
    ..............
End sub
```

مثال:

اجراء sum تقوم بجمع عددين (حيث نمرر العددين)و طباعة الناتج.

```
Public Sub sum(byval x as integer, byref y as integer)
    dim z as integer
X=x+10 : y=y+30
    z=x+y
     msgbox(z)
End sub
```

مثال:

دالة sum1 تقوم بجمع عددين (حيث نمرر العددين) و الرجوع بالناتج.

```
Public function sum1(byval x as integer, byref y as integer)
as integer
sum1=x+y
      dim z as integer
     z=x+y
      return z
End sub
```

استدعاء الاجراءات

- الاجراء الذي لا يعيد قيمة نستدعيه بذكر اسمه أو مسبوقا بـ call.

```
X=10 : y=20
Sum (x , y)
Msgbox(x)   //10
msgbox(y)   // 50
```

- الاجراء الذي يعيد قيمة نستدعيه بذكر اسمه مسبوقا بالمتغير الذي يستقبل القيمة الراجعة.

```
Call  sum (30,40)
```

```
Dim m as integer
M=sum1(5,200)
...
If(sum1(33,70))>=100 then  ...
```

- كود الاجراء يكتب في module أو في الform نفسها.

تمرير الوسائط

- بالقيمة.
- بالعنوان (الاشارة).

الفرق بين التمرير بالقيمة أو بالإشارة ، في التمرير بالقيمة يقوم الاجراء بتلقي نسخة من البيانات و اذا قام الاجراء بتعديلها فان ذلك لن يؤثر علي المتغير الاصلي على عكس التمرير بالإشارة.

* الافتراضي هو التمرير بالقيمة.

انواع البيانات

- Built – in (byte, integer, ...).
- User – defined (class,...).

البرمجة بالكائنات OOP

كل شئ في برنامجك عبارة عن شيء / كائن له خصائصه و وظائفه.

اشبه بالعالم الحقيقي الذي تراه يوميا، فالانسان كائن له صفات معينة (خصائص Properties) كالاسم، العمر، اللون، الطول، الوزن، .... الخ، وله افعال يقوم بها (طرق Methods) كالمشي، الكتابة، الضحك، البكاء، النوم، .... الخ، كما ان الانسان تحدث عليه وقائع (احداث Events) تؤثر فيه وينتج عنها ردود فعل كاستقبال رسالة مفرحة او محزنة، التعرض لجلطة في المخ، وصول لكمة خطافية في الفك الايمن، صفعة قوية في الخد الايسر، .... الخ.

كذلك الحال مع كائنات Visual Basic، فهي تحتوي على خصائص تحوي بيانات خاصة بها مثل: Left، Height، BackColor .... الخ، وطرق لتفعل افعال خاصة بها مثل: Move، Refresh، ZOrder .... الخ، واحداث تقع عليها كـ Click، MouseMove، KeyPress، .... الخ تنتج عنها ردود فعل خاصة.

- يتم التعامل مع اى كيان (شئ) على انه فصيله لها خصائصها (صفاتها) + سلوكها (وظائفها).
- الفكرة مبنية على دمج البيانات و الدالات التي تعمل على تلك البيانات في كينونة واحدة.
  - البيانات تسمى بـ member variables
  - الدوال تعرف بـ member functions / methods
- مثل فصيلة الطالب لها صفاتها (اسم الطالب ، رقمه الجامعى ...) و وظائفها (حساب تقدير الطالب ، حساب معدله ، ....).
- لا نتعامل مع الفصيلة لكن نتعامل مع كائن object مشتق منها.
- الفصيلة هي قالب نشتق منه كائن.

```
int    x;
```

لماذا نحتاج البرمجة بالكائنات OOP

١. اللغات الاجرائية procedural lang.

البرنامج عبارة عن لائحة من التعليمات مثل C, Pascal, ... و يقسم البرنامج الكبير الى اجزاء اصغر (دالات) كل دالة لها هدف معين لجعل البرنامج اسهل فهما و قراءة.

٢. البرمجة البنيوية structured prog.

يتم تقسيم البرنامج الى دالات يتم تجميعها معا في وحدة واحدة (module).

عيوب اللغات الاجرائية ، البرمجة البنيوية

١. بعض الدوال تغير في البيانات التي قد تستخدمها دوال اخرى (متغيرات عامة).
٢. طريقة تخزين البيانات (مصفوفة مثلا) اذا تم تغيير ترتيب البيانات فلابد من تغيير الدوال التي تستخدمها.
٣. لا يوجد اجراء لإخفاء البيانات عن بعض الدالات واظهارها لدالات اخرى.
٤. صعوبة تصميم البرامج الاجرائية.
٥. صعوبة انشاء انواع بيانات جديدة.

سمات OOP

من الضروري ان اوضح الفرق بين الفئة Class والكائن Object، فالفئة –بشكل مبسط- هي مجرد وصف لخصائص، طرق واحداث الكائن، بينما الكائن هو وحدة تحتوي على بيانات واكواد معرفة في الفئة. اعود للمثال السابق، فالانسان هو فئة خلقها الله عز وجل واصفة لخصائص، طرق واحداث كائنات مشتقة منها، فأنا –واعوذ بالله من كلمة انا- كائن لدي خصائص من فئة الانسان كالاسم تركي، العمر 99 .... الخ، وانت ايضا كائن لديك خصائص من نفس الفئة "الانسان" كاسمك س، عمرك ص .... الخ. كذلك الحال مع Visual Basic، فادوات النص Text1 و Text2 هي كائنات من الفئة TextBox، وادوات العنوان Label1، Label2 و Labe3 هي كائنات من الفئة Label.

مميزات البرمجة بالكائنات OOP

١. كبسلة البيانات (الاخفاء) encapsulation
٢. انشاء انواع بيانات جديدة
٣. التجريد abstraction
٤. التوارث inheritance
٥. التعددية الشكلية polymorphism

التغليف:

يقصد بالتغليف Encapsulation في لغات OOP بوضع جميع الاشياء معا Putting everything together، بحيث تحقق استقلالية الكائن المطلقة ببياناته الخاصة به وحتى اكواده، من المزايا التي يقدمها لك التغليف هو امكانية تطوير البنية التحتية للكائن بدون ان يتأثر تركيب برنامجك ودون الحاجة الى تعديل سطر واحد من اكواد البرنامج، مثلا لو قمت بتصميم فئة للبحث عن الملفات واعتمدت عليه بدرجة كبيرة في برنامجك، وبعد فترة من الاختبارات والتجارب القوية لاحظت بطء في عملية التنفيذ، فكل ما ستفعله هو تعديل البنية التحتية للفئة الخاصة بالبحث وتطوير خوارزميات اكوادها دون تغيير سطر واحد من سطور البرنامج الاخرى والتي تستعمل هذه الفئة بالتحديد.

كلما زادت استقلالية الفئة، كلما زادت كفاءة اعادة استخدامها في برنامج آخر وتطبيق اسلوب اعادة استخدام الاكواد Code Reusability. مبدأ اعادة استخدام الاكواد من احد المبادئ الضرورية التي يتوجب عليك محاولة والتعود على تطبيقها دائما في برامجك ومشاريعك اليومية، بحيث تتمكن من الاستفادة من الفئة التي صممتها في اكثر من مشروع واكثر من برنامج. وحتى تنشئ فئة قابلة لاعادة الاستخدام، حاول دائما وقبل ان تبدأ بكتابة سطر واحد من الفئة باخذ احتياطاتك

كبسلة البيانات ، التغليف (الاخفاء) encapsulation

- تحزيم كل معلومات وقدرات كيان ما فى كائن واحد ليسهل الرجوع اليه وقتما نشاء.
- اشتقاق كائن من الطبقة للتعامل مع بيانات الطبقة ووظائفها من خلال الكائن.
- تسهل عملية الصيانة لوجود البيانات والاوامر في مكان واحد (الطبقة).
- التغليف يتيح للمستخدم استعمال الصنف دون الاهتمام بتفاصيله او كيفية عمله المهم معرفة ما يفعله.

التجريد abstraction

- عند حل مشكلة لا ننظر الى تفاصيلها الغير مفيدة او التفاصيل التى لا تتعلق بالمشكلة بل نركز على البيانات المتعلقة بالمشكلة.

التعددية الشكلية polymorphism

- الاستعمال المتعدد الاغراض للدوال و الادوات مثل +،-،*
- مثل :
- "+" تستخدم لتمثل جمع عددين صحيحين او حقيقين او جمع حرفين او جملتين حسب الغرض من الاستخدام.
- فصيلة تتوارث صفات ووظائف فصيلة اخرى و كلاهما يستخدم نفس الوظائف لكن بطرق تطبيق مختلفة.

تعدد الواجهات:

بباسطة مبدأ تعدد الواجهات Polymorphism هو قدرة الفئة على احتوائها اكثر من واجهة بحيث تمكنك من توحيد عدة فئات مختلفة باسماء اعضاء متشابهه، فلو امعنت النظر قليلاً في ادوات Visual Basic ستجد انها مختلفة المهام والانجازات الا انها تحتوي على خصائص، طرق واحداث مشتركة كـ Left، Moveو Click مما يسهل عليك كمبرمج حفظها وتوحيد الاجراءات التي تستخدم هذه الاعضاء. الفصل القادم يناقش مبدأ تعدد الواجهات بالتفصيل.

التوارث inheritance

انشاء فصيلة بها صفاتها + دوالها بالاضافة الى صفات و دوال الفصيلة المشتقة منها.

الوراثة:

الوراثة Inheritance هي قدرة فئة على اشتقاق اعضاء من فئة ام بحيث تزيد من قوة الفئة الوارثة وتضيف اعضاء جديدة للفئة الام، فلو كان لديك فئة قوية واردت اضافة طريقة او خاصية لها، فلا يوجد داعي لاعادة بناء الفئة من جديد واضافة الخاصية او الطريقة المطلوبة، فكل ما ستقوم به هي عملية انشاء فئة خالية تضيف اليها الخاصية او الطريقة التي تريدها ومن ثم تشتقها من الفئة التي تريد تطويرها واضافة الخاصية او الطريقة لها. الفصل القادم يناقش مبدأ الوراثة بالتفصيل.

٤-٢-٥ مفهوم التصنيف أو الفئة (Classes)

هو عبارة عن قالب أو مخطط يتم منه إنشاء كائن معين، وهو يمثل جميع الخصائص والوظائف التي سوف يحتويها الكائن بعد ذلك. أما الكائن (Object) فهو يمثل وحدة مستقلة تم إنشاؤها من التصنيف (Class) وهي التي تستخدم فعلاً لأداء الوظائف المختلفة للتصنيف(Class) .

الفصيلة class

هي الفكرة المجردة لشيء معين، تحتوي علي مجموعة من الصفات و السلوكيات التي يقوم بها النوع.

انشاء الفصيلة

Project ثم add new item ثم class ثم نحدد اسم الفصيلة.

```
Public class class name
...
End class
```

مثال:

انشئ فصيلة student تحتوي علي اسم الطالب ، عمره ، السنة الدراسية، الوظيفة sayhello للترحيب بالطالب.

```
Public class student
   public stname as string
   public stage, styear as integer
   public sub sayhello()
        msgbox("hi "&stname)
   end sub
End class
```

الكائن object

لا يمكن التعامل مع الفصيلة مباشرة بل نشتق منها الكائن للتعامل معه.

Dim object-name as new class-name

Or

Dim object-name as class-name

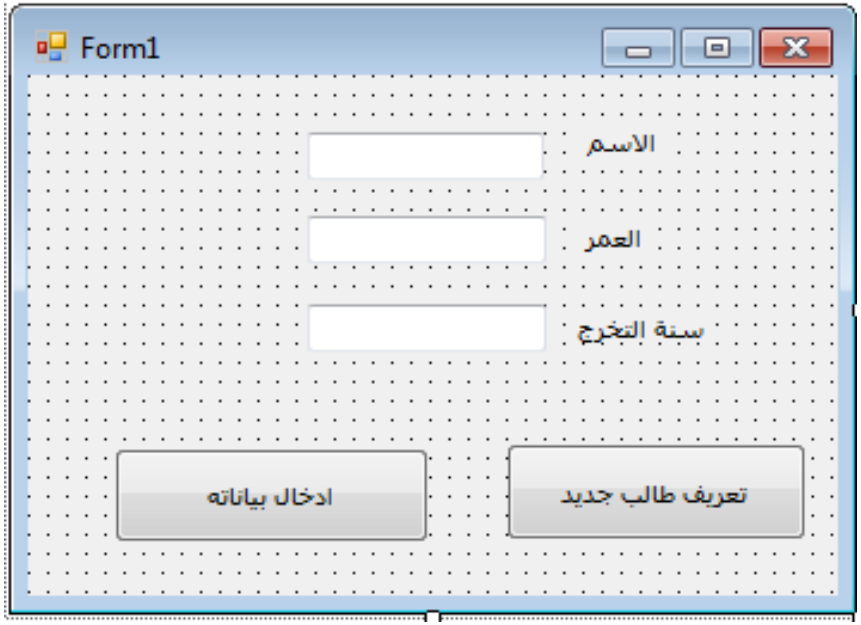
object-name=new class-name

- New : لانشاء نسخة جديدة من الفصيلة ووضعها في الذاكرة.
- يمكن انشاء أي عدد من الكائنات .
- الاعلان عن الكائن يتم في الـform لا في الـclass.

مثال

في المثال السابق student انشي كائن ثم تعاملي معه من خلال الـ form التالية.

```
' في منطقة الاعلان
Dim studi as student
' كود زر تعريف طالب جديد
Studi=new student
' كود زر ادخال البيانات
Studi.stname=t1.text
Studi.stage=t2.text
Studi.styear=t3.text
studi.sayhello()
```



مثال

```
Class Person
    Public FirstName As String
    Public LastName As String
    Public Age As Integer
End Class
```

```
Dim Ahmed As New Person()
Ahmed.Age = 15
Ahmed.FirstName = "Ahmed"
Ahmed.LastName = "Gamal"
Dim Ali As New Person()
Ali.Age = 15
Ali.FirstName = "Ahmed"
Ali.LastName = "Gamal"
```

مثال

دالة تعيد لنا الاسم الكامل لشخص معين

```
Public Function getFullName() As String
    Return FirstName + LastName
End Function
```