

أساسيات لغة Visual Basic

- خصائص الأدوات
- الأحداث ونافذة الشفرة
- المتحولات
- أنواع المعطيات
- بني التحكم

تم تحميل هذا الكتاب من موقع كتب
www.kutub.info
للمزيد من الكتب في جميع مجالات التقنية ، تفضلوا بزيارتنا

خصائص الأدوات:

تعريف الخصائص:

الخصائص هي مجموعة من المواصفات التي تغيير من سلوك ومظهر الأدوات، لكل أداة في Visual Basic - بما في ذلك نافذة البرنامج Form - مجموعة محددة من الخصائص Properties مثل: لون الأداة، عنوان الأداة، حجم الأداة، موقع الأداة الخ.

عندما تقوم بإضافة أداة ما إلى نافذة البرنامج فإن Visual Basic تقوم بضبط خصائص هذه الأداة على قيم افتراضية، وبعد ذلك تستطيع تعديل هذه الخصائص كيفما تريد.

تغيير (ضبط) الخصائص:

تتم عملية ضبط خصائص الأدوات أثناء تصميم البرنامج فقط باستخدام نافذة الخصائص Properties Window ، وهناك ثلاث خطوات تمر بها عملية تغيير الخصائص وهي :

- 1- اختيار الأداة التي نريد ضبط خصائصها من نافذة البرنامج.
- 2- اختيار الخاصية التي نريد تغييرها من نافذة الخصائص.
- 3- إدخال القيمة الجديدة.

ولإظهار نافذة الخصائص لأداة ما ، نقوم أولاً بتحديد الأداة ثم نضغط F4 فتظهر كما الشكل (1-2).

The screenshot shows the 'Properties - Form1' window with the following properties listed:

(Name)	Form1
Appearance	1 - 3D
AutoRedraw	False
BackColor	FF& (highlighted)
BorderStyle	2 - Sizable
Caption	Components
ClipControls	True
ControlBox	True
DrawMode	13 - Copy Pen
DrawStyle	0 - Solid
DrawWidth	1
Enabled	True
FillColor	&H000000
FillStyle	1 - Transparer
Font	MS Sans Serif
FontTransparen	True
ForeColor	&H8000001
Height	3600
HelpContextID	0

Annotations in the image:

- Top left: الاسم البرمجي للأداة المختارة (The program name for the selected tool)
- Top right: الاسم الشائع للأداة المختارة (The common name for the selected tool)
- Middle left: عرض الخصائص مرتبة أبجدياً (Display properties alphabetically)
- Middle right: عرض الخصائص مرتبة حسب النوع (Display properties by type)
- Bottom left: عمود الخصائص : وهو يعرض أسماء الخصائص ، واسم الخاصية مفيد جداً بالنسبة لنا لأننا سنستعمله عند تغيير الخاصية برمجياً بعد التنفيذ (Properties column: It displays the names of the properties, and the property name is very useful for us because we will use it when changing the property programmatically after execution)
- Bottom right: عمود القيم : وفيه تعرض قيم الخصائص ومنه نستطيع تغيير قيمة أي خاصية من الخصائص (Value column: It displays the values of the properties, and from here we can change the value of any property from the properties)

الشكل (1-2) نافذة الخصائص



خانة اسم الأداة : وهي تعرض اسم الأداة النشطة ونوعها، فإذا أردت أن تعدل خصائص أداة أخرى غير تلك المعروضة في هذه الخانة، فما عليك سوى أن تنقر فوق هذه الأداة من نافذة البرنامج فتتحدث محتويات نافذة الخصائص لتعرض خصائص الأداة الجديدة المختارة. أو تختار هذه الأداة من خانة اسم الأداة فتصبح هي النشطة وتُعرض خصائصها.

تغيير قيمة خاصة ما:

لتغيير قيمة خاصة ما نضغط فوق اسم الخاصية مما يؤدي إلى تظليلها هي وقيمتها، وبعد ذلك نغير قيمة هذه الخاصية من عمود القيم، وعند تغيير قيمة خاصة ما يجب التمييز بين الأنواع التالية من الخصائص وهي:

1. **الخاصية النصية :** وهي الخاصية التي قيمتها عبارة عن سلسلة محارف كخاصية الـ Name, Caption.

2. **الخاصية الرقمية :** وهي الخاصية التي قيمتها عبارة عن رقم مثل خصائص الأبعاد Width, Height.

3. **الخاصية المنطقية :** وهي الخاصية التي قيمتها إما **True** (والتي تدل على أن الخاصية فعالة)

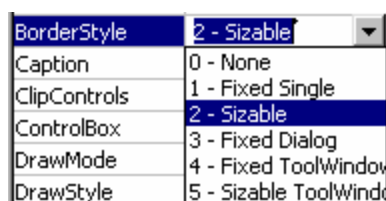
أو **False** (والتي تدل على أن الخاصية غير فعالة).

ومن الخصائص المنطقية خاصية Visible التي تتحكم بإظهار الأداة عندما تكون قيمتها True وإخفاء الأداة عندما تكون قيمتها False.

4. **الخاصية اللونية :** وهي خاصية قيمتها عبارة عن لون محدد بإحدى توابع الألوان Qbcolor أو RGB أو عدد

ست عشري، ومن أهم الخصائص اللونية خاصية BackColor.

5. **الخاصية التي قيمتها عبارة عن ملف:** مثل خاصية Picture و Icon و ...



6. **الخاصية واحد من مجموعة:** وفيها علينا منح الخاصية إحدى قيم

القائمة التي تنسدل أمامنا ولا يمكننا إعطاء قيمة من خارجها.

وكمثال على هذه الأنواع من الخصائص، خاصية BorderStyle التابعة للنافذة والتي تأخذ قيمة من ست قيم فقط موجودة في القائمة التي تنسدل

عند الضغط على السهم الصغير بجوارها.

تغيير الخصائص لأكثر من أداة:

إذا أردت ضبط الخصائص لأكثر من أداة في آن واحد، قم بتحديد هذه الأدوات معاً، ثم اضغط المفتاح F4 لإظهار نافذة الخصائص التي ستعرض الخصائص المشتركة فقط بين الأدوات المحددة، غير الخاصية التي تريد وستلاحظ أن هذا التغيير سيؤثر على جميع الأدوات المحددة (جرب ذلك على الخاصية Caption لثلاثة أزرار مثلاً).

هناك بعض الخصائص التي لا يظهر تأثيرها عند التصميم ولكن يظهر فقط عند التنفيذ مثل الخاصية Mouse icon التي تتحكم بشكل المشيرة عند مرورها فوق الأداة .

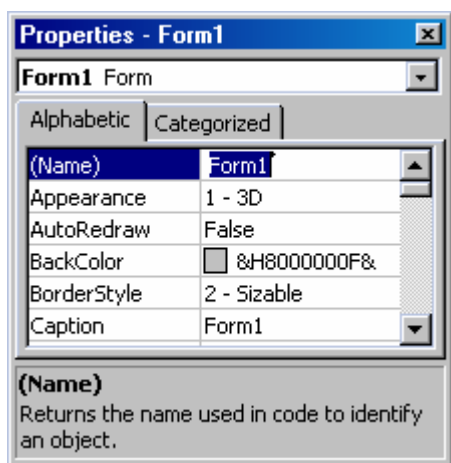
لتنفيذ البرنامج نضغط على المفتاح F5، ولإيقاف تنفيذ البرنامج نضغط على زر الأغلاق الخاص بنافذة البرنامج

الخصائص الشائعة:

هناك مجموعة من الخصائص الشائعة الاستخدام والمتوفرة لمعظم الأدوات، سنقوم الآن بشرح أهم هذه الخصائص وسنؤجل الحديث عن باقي الخصائص لوقت آخر.

1- الخاصية Name:

تعتبر هذه الخاصية من أهم الخصائص على الإطلاق، وهي متوفرة لجميع الأدوات دون استثناء، وهذه الخاصية تحدد الاسم البرمجي للأداة، وهو الاسم الذي يستخدم عند كتابة شفرة تخص هذه الأداة مثل:



`Form1.Caption="Test"`

حيث Form1 تمثل اسم النافذة.

عندما تضع أداة جديدة أو تضيف نافذة جديدة يتم وضع الخاصية Name افتراضياً لهذه الأداة وذلك بذكر اسم الأداة يليها رقم مثل Form1 و Form2 و Label1 و Label2.. الخ.

والآن إذا كنت ترغب في تغيير هذا الاسم الافتراضي فعليك تذكر ما يلي:

- 1- يجب أن يبدأ الاسم بحرف ولا يجوز أن يبدأ برقم، ويجوز أن يتخلله أرقام.
- 2- يفضل أن يكون الاسم باللغة الإنكليزية، وذلك لتجنب المشاكل التي يمكن أن تحدث عند استخدام الأسماء العربية.

3- يجب أن لا يتجاوز الاسم 40 حرفاً.

4- لا يجوز استخدام بعض الحروف مثل النقطة و الفراغ و الفاصلة و ...

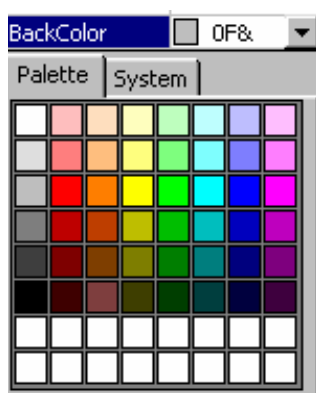
5- لا يجوز استخدام الكلمات المحجوزة مثل : FOR و WHILE و FUNCTION و ...

6- يفضل استخدام الأسماء التي تدل على وظيفة الأداة، وتجنب الأسماء العشوائية.

مثلاً: يمكنك تسمية النافذة "F" بدلاً من "Form1" و لكن عندها ستصبح الشفرة على الشكل:

`F.Caption="Test"`

الخاصية Name متوفرة أثناء التصميم فقط، أي من المستحيل تغيير الخاصية Name ضمن الشيفرة وهناك العديد من الخصائص الأخرى تشترك معها بهذه الصفة.



2- الخاصية BackColor (لون الأرضية):

تحدد هذه الخاصية لون أرضية الأداة، وعند محاولة تغيير هذه الخاصية يظهر مربع صغير يحوي سهم ، عند الضغط على هذا المربع يظهر لوح الألوان الذي يمكننا من اختيار اللون الذي نريد .

ونلاحظ في مربع الألوان وجود بوابتين الأولى Palette ومنها نختار ألوان ثابتة ، والثانية System ومنها نختار ألوان يستخدمها النظام Windows .

3- الخاصية Caption (العنوان):

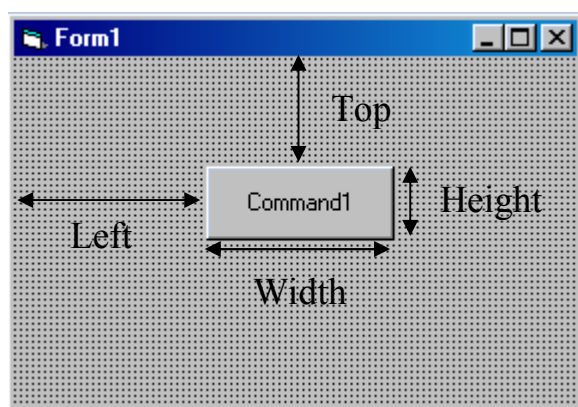
وهي تحدد النص الذي سيظهر على الأداة كعنوان لها، ويجب أن لا يتجاوز النص 255 حرفاً بما في ذلك الفراغات.

4- الخاصية Enabled (التمكين):

تحدد هذه الخاصية فيما إذا كانت الأداة ستتأثر بالأحداث (النقر أو حركة الماوس) أم لا، حيث تأخذ القيمتين True تتأثر أو False لا تتأثر. لن يظهر تأثير هذه الخاصية إلا بعد تنفيذ البرنامج.

5- الخاصية Font (الخط):

تستخدم من أجل تحديد شكل ونوع وحجم الخط الذي سيظهر به عنوان الأداة.



6- الخاصية ForeColor (لون الخط):

وهي تحدد لون الخط الذي سيكتب به عنوان الأداة.

7- الخاصية **Height**: تحدد ارتفاع الأداة مقدراً بـ **Twip**.

Twip=1/15 Picxel

8- الخاصية **Width**: تحدد عرض الأداة.

9- الخاصية **Left**: تحدد مقدار بعد الطرف الأيسر للأداة عن الطرف الأيسر للنافذة.

10- الخاصية **Top**: تحدد مقدار بعد الطرف العلوي للأداة عن الطرف العلوي للنافذة.

11- خاصية **Picture**: وتستخدم لتحميل صورة ووضعها كخلفية للأداة، ومن الجدير بالذكر أن Visual Basic

تستطيع التعامل مع عدد كبير من أنواع الصور أشهرها صور: **BMP**، **WMF**، **DIB**، **ICO**، **JPG**، و **JPG**.

لإزالة الصورة من على النافذة نقوم بتحديد قيمة الخاصية **Picture** ثم نضغط على المفتاح **Delete**.

12- خاصية **Visible**: وتستخدم لإظهار أو إخفاء الأداة أثناء التنفيذ. ففي بعض الأحيان نضطر لإخفاء الأداة

لسبب معين. لن يظهر تأثير هذه الخاصية إلا بعد تنفيذ البرنامج.

من المهم الآن أن تقوم بإضافة الأدوات وتجريب الخصائص السابقة، وتصميم واجهات مختلفة لتخليها.

تغيير الخصائص أثناء التنفيذ:

قبل أن نتعرف على الأحداث الشائعة لابد من أن تعلم أن الشيفرة الأكثر استخداماً في Visual Basic هي الشيفرة التي تُستخدم لتغيير خصائص الأدوات بعد التنفيذ، وهذه الشيفرة تملك الشكل العام التالي:

```
القيمة الجديدة = اسم الخاصية . اسم الأداة . اسم النافذة
FormName.ToolName.PropertyName = NewValue
```

حيث :

اسم النافذة: هو اسم النافذة الموجودة عليها الأداة التي نريد تغيير خصائصها، ونحصل عليه من خاصية **Name** التابعة للنافذة.

اسم الأداة: وهو اسم الأداة التي نريد تغيير خصائصها، ونحصل عليه من نافذة الخصائص التابعة للأداة ومن الخاصية **Name** بالتحديد.

اسم الخاصية: وهو اسم الخاصية التي نريد تغيير قيمتها، ونحصل عليه -في حال لم نكن نحفظه- من نافذة الخصائص التابعة لهذه الأداة.

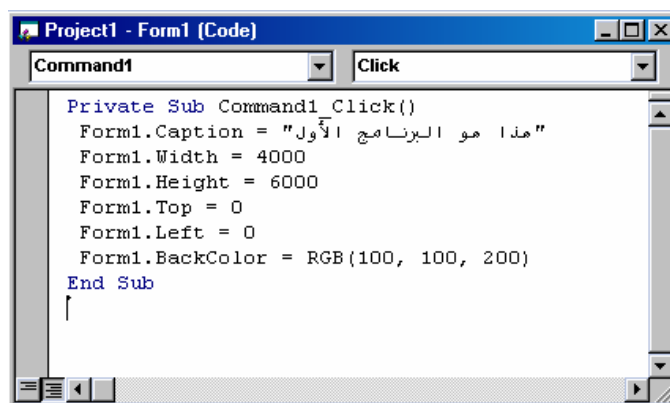
القيمة الجديدة: وهي القيمة الجديدة التي نريد منحها للخاصية، وهنا يجب التنبيه على أن القيمة الجديدة يجب أن تتوافق مع نوع معطيات الخاصية:

- في الخصائص الرقمية: يجب أن تكون القيمة الجديدة عبارة عن رقم.
- في الخصائص النصية: يجب أن تكون القيمة الجديدة عبارة عن نص محصور بين إشارتي تنصيص " " .
- في الخصائص المنطقية: يجب أن تكون القيمة الجديدة إما true أو False.
- في الخصائص اللونية يجب استخدام أحد توابع الألوان مثل Qbcolor(number) أو التابع RGB.

مثال :

- 1- ابدأ بمشروع جديد (من القائمة File اختر الأمر New Project ثم اختر مشروع قياسي).
- 2- ضع زر أوامر على النافذة، وغير خاصية Caption له إلى "تغيير الخصائص".
- 3- اضغط فوق الزر ضغطتين سريعتين، فتظهر نافذة الشفرة، اكتب فيها مايلي:

```
Form1.Caption = "هذا هو البرنامج الأول"
Form1.Width = 4000
Form1.Height = 6000
Form1.Top = 0
Form1.Left = 0
Form1.BackColor = RGB(100, 100, 200)
```



```
Project1 - Form1 (Code)
Command1 Click
Private Sub Command1_Click()
    Form1.Caption = "هذا هو البرنامج الأول"
    Form1.Width = 4000
    Form1.Height = 6000
    Form1.Top = 0
    Form1.Left = 0
    Form1.BackColor = RGB(100, 100, 200)
End Sub
```

- 4- نفذ البرنامج بالضغط على المفتاح F5 أو اختيار الأمر Start من القائمة Run، ومن ثم اضغط على الزر "تغيير الخصائص"، ولاحظ ما سيحدث.

الدالة اللونية **RGB(Red,Green,Blue)** تأخذ ثلاثة بارامترات:

Red: يأخذ القيم من 0 حتى 255 يحدد مقدار اللون الأحمر في المزيج اللوني.

Green: يأخذ القيم من 0 حتى 255 يحدد مقدار اللون الأخضر في المزيج اللوني.

Blue: يأخذ القيم من 0 حتى 255 يحدد مقدار اللون الأزرق في المزيج اللوني.

وتعود بقيمة لونية تمثل المزيج المكون من الألوان الثلاثة السابقة، أي تعطي 16000000 لون.

هناك دالة أخرى خاصة باللون وهي **QbColor(Number)** : وهي نفس الدالة المستخدمة في QBasic فالبارامتر **Number** يأخذ قيمه من 0 إلى 15، أي تستخدم لإعطاء خمسة عشر لوناً فقط.

الأحداث ونافذة الشيفرة:

تعتمد فكرة البرمجة في Visual Basic على الأحداث، وكما ذكرنا، الحدث هو فعل يقوم به المستخدم على البرنامج مثل ضغط زر أو تحريك الماوس. ويجب عليك كمبرمج Visual Basic أن ترد على أحداث المستخدم هذه، لذلك ستجد أن شفرة Visual Basic مقسمة إلى إجراءات (أحداث).

تملك Visual Basic نافذة خاصة لتحرير الشيفرة وكتابة التعليمات وهذه النافذة تعتبر بمثابة محرر نصوص بسيط، فهي توفر عمليات النسخ والقص واللصق، وتساعد في تحرير الشيفرة وإكمالها من خلال عرض قوائم الخيارات وما شابه.

وأهم ما تقوم به هذه النافذة هي تقسيم الشيفرة إلى أجزاء (برامج جزئية) ويتم ذلك اعتماداً على الأحداث، فمقابل كل حدث يوجد برنامج جزئي (إجراء) يُستدعى تلقائياً عند وقوع الحدث.

وللوصول إلى أحد أحداث أداة ما، نقوم بالنقر المزدوج فوق هذه الأداة فتظهر نافذة الشيفرة الخاصة بهذه الأداة ويجدتها الأكثر استخداماً، ويمكن التنقل بين الأدوات والأحداث الخاصة بها باستخدام القوائم الموجودة في أعلى نافذة الشيفرة.

فمثلاً: لو أضفت زر أوامر ثم ضغطت فوقه ضغطتين مزدوجتين ستظهر لك نافذة الشيفرة كما في الشكل (2-2):



الشكل (2-2) نافذة الشيفرة

كما تلاحظ تقوم Visual Basic بتوليد إجرائية خاصة بكل حدث، اسم هذه الإجرائية مكون من اسم الأداة واسم الحدث على الشكل:

اسم الحدث_ اسم الأداة

فما تلاحظ اسم الإجرائية السابقة هو: *Command1_Click*

تحتوي القائمة الموجودة في أعلى يسار نافذة الشفرة على أسماء جميع الأدوات الموجودة على النافذة، بالإضافة إلى اسم النافذة Form وقسم التصريح (General).

تحتوي القائمة الموجودة في أعلى يمين نافذة الشفرة على أسماء الأحداث التابعة للأداة المختارة من القائمة اليسرى. عند اختيار الأداة والحدث تتولد الإجرائية المناسبة كما يلي:

() اسم الحدث_ اسم الأداة *Private Sub*

End Sub

بعد ظهور سطري البداية والنهاية الخاصين بالإجرائية يمكنك كتابة الشفرة التي تريد، ولكن تذكر أنه يجب أن تبقى الشفرة ضمن السطرين السابقين، وأن هذه الشفرة ستنفذ عند وقوع الحدث.

الأحداث الشائعة:

يوجد مجموعة من الأحداث الشائعة التي تستخدم بكثرة وهي:

الحدث Click: يقع هذا الحدث عند الضغط على الأداة بزر الماوس الأيسر، يعتبر الحدث Click من أكثر الأحداث استخداماً فهو متوفر لمعظم الأدوات وأهمها زر الأوامر *CommandButton*.

مثال:

1- ابدأ بمشروع جديد.

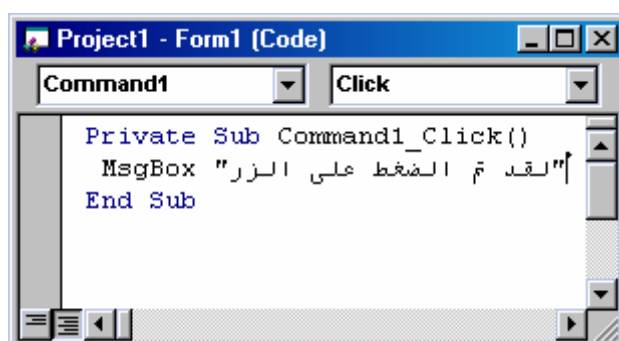
2- ضع زر أوامر *Command1* على النافذة.

3- اضغط على الزر ضغطتين سريعتين للحصول على نافذة الشفرة.

4- اكتب الشفرة التالية:

MsgBox "لقد تم الضغط على الزر"

وبالتالي يجب أن تبدو نافذة الشفرة كما يلي:



يستخدم الأمر MsgBox لعرض رسائل نصية على المستخدم.

5- من القائمة الموجودة في أعلى يسار النافذة اختر

Form، ومن القائمة اليمنى اختر Click، فيظهر السطرين:

```
Private Sub Form_Click ()
```

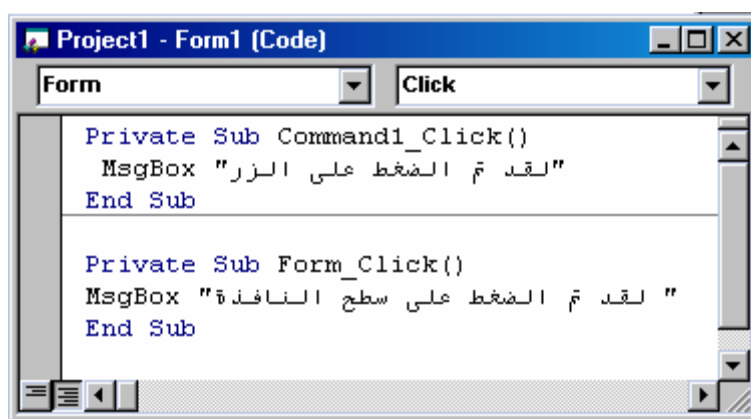
```
End Sub
```

وهذا دلالة على أن ما سنكتبه الآن سينفذ عند الضغط على سطح النافذة.

7- اكتب الشفرة التالية بين السطرين السابقين:

” لقد تم الضغط على سطر النافذة “ MsgBox

وبهذا ستبدو نافذة الشفرة مشابهة للشكل.



8- نفذ البرنامج السابق بالضغط على المفتاح F5 أو باختيار الأمر Run-Start، وبعد ذلك قم بتجربة البرنامج بالنقر

على الزر ثم على النافذة، وانتبه للرسائل التي ستظهر.

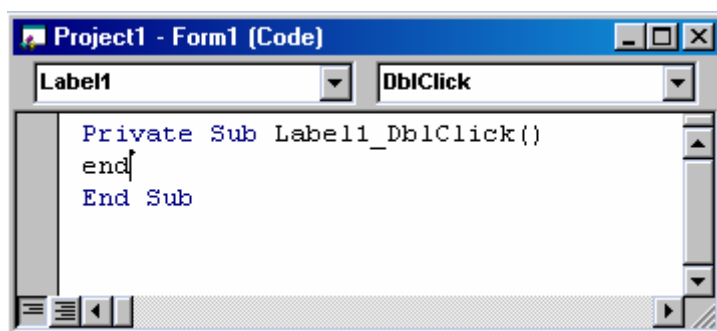
9- احفظ هذا المشروع، لأنك قد تعود إليه لاحقاً.

ملاحظة: تذكر أنك ستطالب بحفظ ملفين هما ملف النافذة ولاحقته FRM وملف المشروع ولاحقته VBP.

الحدث Dblclick:

يقع هذا الحدث عند الضغط المزدوج فوق الأداة.

مثال:



1- ابدأ بمشروع جديد.

2- ضع أداة عنوان Label على النافذة.

3- في الحدث Dblclick الخاص بالأداة Label، قم بكتابة الأمر:

End

يجب أن تبدو نافذة الشفرة مشابهة لما يلي:

4- نفذ البرنامج، ثم اضغط فوق أداة العنوان ضغطتين مزدوجتين فينفذ الأمر End ويغلق البرنامج.

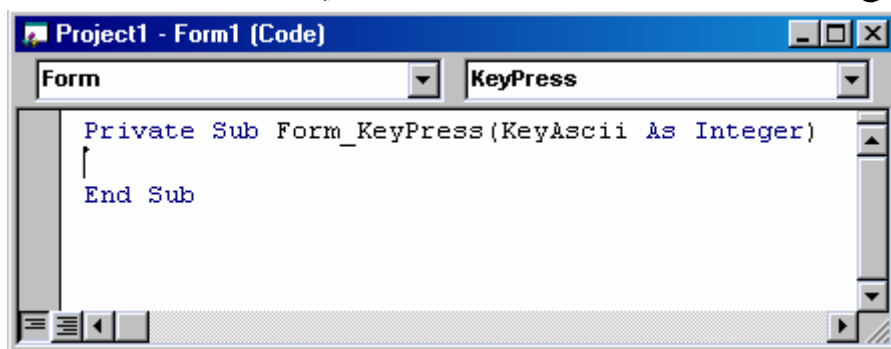
الحدث KeyPress:

يقع هذا الحدث عند الضغط على مفتاح ما من لوحة المفاتيح، وللمعرفة المفتاح المضغوط نستطيع اختبار قيمة الوسيط KeyAscii الخاص بهذا الحدث، حيث يمثل KeyAscii رقم المفتاح المضغوط في جدول الآسكي.

مثال:

1- ابدأ بمشروع جديد.

2- ادخل إلى الحدث KeyPress التابع للنافذة Form، فتظهر لك نافذة الشفرة كما يلي:



3- اكتب الأمر End.

```
Private Sub Form_KeyPress(KeyAscii As Integer)
```

```
End
```

```
End Sub
```

4- نفذ البرنامج، واضغط على أي مفتاح فيغلق البرنامج.

ملاحظة: كان من الممكن تحديد مفتاح معين ليقوم بإغلاق البرنامج، وذلك باختبار قيمة الوسيط KeyAscii على الشكل:

```
IF KeyAscii=27 THEN End
```

أي إذا ضغط على المفتاح 27 (Esc) قم بإنهاء البرنامج.

الحدث Keydown:

يقع هذا الحدث عندما نضغط فوق أحد أزرار لوحة المفاتيح وقبل الإفلات، وهو يقدم وسيطين هما:

KeyCode: وهو يمثل رقم المفتاح المضغوط، وهذا الرقم يشمل مفاتيح التحكم أيضاً.
Shift: ويستخدم لمعرفة إن كان أحد مفاتيح التحكم (Ctrl, Alt, Shift) مضغوطاً أثناء ضغط المفتاح، فهذا الوسيط يمكن أن يأخذ إحدى القيم التالية:

1 ويعني أن المفتاح Shift مضغوطاً أثناء ضغط المفتاح.

2 ويعني أن المفتاح Ctrl مضغوطاً أثناء ضغط المفتاح.

4 ويعني أن المفتاح Alt مضغوطاً أثناء ضغط المفتاح.

ملاحظة: لا يوجد 3 لأن $1+2=3$ ومعنى ذلك أن المفاتيح Shift و Ctrl مضغوطين، فانتبه لذلك!

مثال:

ادخل إلى الحدث KeyDown الخاص بالنافذة واكتب الشفرة:

```
Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
If Shift = 1 And KeyCode = 27 Then End
End Sub
```

أي إذا تم ضغط Shift+Esc قم بإنهاء البرنامج.

الحدث **KeyUp**: وهو نفس الحدث Keydown تماماً ولكنه يقع بعد إفلات المفتاح المضغوط .

الحدث **MouseMove**: يقع هذا الحدث عندما نقوم بتحريك مؤشر الماوس فوق النافذة، ويقدم أربع بارا مترات:

1. **Button**: ومنه تستطيع اختبار الزر المضغوط أثناء التحريك، ويأخذ هذا الوسيط ثلاث قيم هي:

1 للزر الأيسر، 2 للزر الأيمن، 4 للزر الأوسط.

ملاحظة: يمكن جمع الأرقام السابقة للحصول على ضغط زر، فمثلاً الرقم 3 يعني أن الزرين الأيسر والأيمن

مضغوطين أثناء تحريك الماوس.

2. **Shift**: ومنه تستطيع معرفة إن كان أحد المفاتيح (Shift, Ctrl, Alt) مضغوطة أثناء تحريك الماوس.

3. **X و Y**: يمثلان إحداثيات مؤشر الماوس على النافذة.

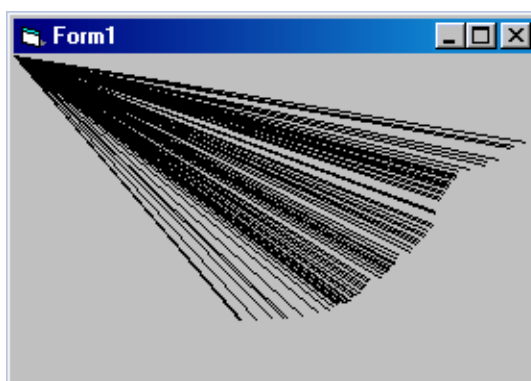
مثال:

1- أدخل الشفرة التالية في الحدث MouseMove التابع للنافذة:

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Line (0, 0)-(X, Y)
End Sub
```

أي ارسم خط من الموقع (0,0) إلى موقع المؤشر على النافذة (x,y).

2- نفذ البرنامج، ثم حرك مؤشر الماوس فوق النافذة وانظر كيف يتم رسم الخطوط.



3- أغلق البرنامج ثم استبدل الشفرة السابقة بالشفرة التالية:

```
IF Button = 2 Then Line (0, 0)-(X, Y)
```

الآن - وبعد التنفيذ- لن يتم رسم الخط إلا إذا ضغطنا الزر الأيمن أثناء تحريك الماوس على النافذة.

الحدث MouseDown: يقع هذا الحدث عندما يقوم المستخدم بالضغط على النافذة بأحد أزرار الفأرة وقبل الإفلات، وهذا الحدث يقدم أربع وسطاء هي نفسها وسطاء الحدث MouseMove.

الحدث MouseUp: يشبه هذا الحدث حدث MouseDown ولكنه يقع عند إفلات زر الماوس بعد أن يكون مضغوطاً.

الحدث Load: وهو خاص بالنافذة فقط، ويقع عند تحميل النافذة، أي يتم عند تنفيذ البرنامج. يستخدم هذا الحدث لعرض رسائل الترحيب، وتميئة المتحولات وفتح الملفات وغير ذلك.

مثال:

1- اكتب الأمر التالي في الحدث Load التابع للنافذة:

MsgBox "مرحباً بكم"

2- نفذ البرنامج، وستلاحظ ظهور الرسالة السابقة قبل ظهور النافذة.

المتحولات وأنواع المعطيات في Visual Basic:

يوجد عدة أنواع للمتحولات في Visual Basic، وذلك اعتماداً على مجال رؤية المتحول وعمره.

نوع المتحولات	مجال رؤيتها	عمرها
المتحولات الخاصة	ضمن الإجرائية حيث يوجد التصريح عن هذه المتحولات	تبقى قيمة هذه المتحولات معرفة طيلة فترة تنفيذ الإجرائية
المتحولات العامة على مستوى النافذة	ضمن النافذة ككل (أي تراها كل الإجرائيات، والدوال والأدوات)	تبقى قيمة هذه المتحولات معرفة طيلة فترة تحميل النافذة.
المتحولات العامة على مستوى المشروع	ضمن كامل المشروع	يساوي عمر البرنامج
المتحولات الساكنة	ضمن الإجرائية حيث يوجد التصريح عن هذه المتحولات	يساوي عمر البرنامج

1- المتحولات الخاصة: ويصرح عنها ضمن الإجرائية (الحدث)، وعمر هذه المتحولات يساوي فترة استدعاء

الإجرائية، أما مجال رؤيتها فهو داخل الإجرائية فقط، ويتم التصريح عن المتحولات بالشكل:

نوع معطيات As اسم متحول Dim

أنواع المعطيات المتوفرة في VB5 :

إليك الجدول التالي الذي يبين أنواع المعطيات المتوفرة في Visual Basic :

نوع المعطيات	الحجم في الذاكرة	المجال
Byte	1 Byte	0 To 255
Boolean	2 Bytes	True Or False
Integer	2 Bytes	- 32,768 To 32,767
Long	4 Bytes	+2,147,483,647 To -2,147,483,648
Single	4 Bytes	-3.402823E-38 To -1.401298E-45 +1.401298E-45 To 3.402823E38
Double	8 Bytes	4.94065645841247E-324 To 1.79769313486232E308
Currency	8 bytes	-922,337,203,685,477.5808 To 922,337,203,685,477.5807
Date	8 bytes	January 1, 100 To December 31, 9999
String	Length of string	1 to approximately 65,400
Variant with numbers	16 Bytes	Any numeric value up to the range of a Double
Variant With characters	22 Bytes + string length	Same range as for variable-length String
User-defined	Number required by elements	The range of each element is the same as the range of its data type.

مثال 1:

1- ابدأ بمشروع جديد.

2- ضع زر أوامر.

3- في الحدث Command1_Click اكتب الشفرة:

*Dim x AS Integer**Dim y AS Integer**Dim z AS Integer**x = 3**y = 5*

```
z = x + y
```

```
Print z
```

يقوم هذا البرنامج بالتصريح عن ثلاث متحولات خاصة في الحدث Click التابع لزر الأوامر، ومن ثم يقوم بعملية جمع عددين وإظهار النتيجة.

مثال 2:

1- استبدل الشفرة السابقة بما يلي:

```
Dim x AS Integer
```

```
x = x + 1
```

```
Print x
```

2- نفذ هذا البرنامج، واضغط على الزر عدة مرات ستجد أنه في كل مرة سيُطبع العدد واحد على الشاشة رغم أننا نزيد قيمة x، وذلك لأن x متحول خاص ينتهي عمره بمجرد انتهاء الضغط، ويتولد من جديد عند بداية الضغط.

3- ضع زر أوامر ثاني Command2 واكتب في الحدث Click :

```
Print x
```

ثم نفذ البرنامج واضغط على الزر ستجد أنه يتم طباعة صفر في كل مرة، وذلك لأن المتحول x غير مُعرف بالنسبة للزر الثاني.

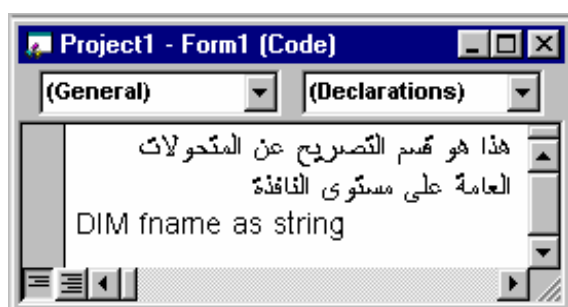
2- المتحولات العامة على مستوى النافذة:

يتم التصريح عن هذه المتحولات في القسم General-Declarations على الشكل:

1- نظهر نافذة الشيفرة بأي طريقة كانت .

2- من القائمة اليسرى نختار القسم General.

3- من القائمة اليمنى نختار القسم Declarations.



وفي هذا القسم يتم التصريح عن المتحولات العامة على مستوى النافذة أي المتحولات التي ستُرى من قبل جميع الأدوات والإجراءات والدوال الموجودة على هذه النافذة، وستحافظ هذه المتحولات على قيمتها ابتداء من لحظة تحميل النافذة إلى الذاكرة ولغاية إزالتها من الذاكرة .

مثال:

1- أدخل إلى قسم التصريحات العامة General-Declarations وصرح عن متحول من نوع عدد كسري:

```
Dim x AS Integer
```

المتحول x سيكون معروف لدى جميع الأدوات الموجودة على النافذة .

2- ضع زر أوامر Command1 واكتب في الحدث Command1_Click :


```
x = x + 1
Print x
```

3- ضع زر أوامر Command2 واكتب في الحدث Command2_Click :

```
Print x
```

4- نفذ البرنامج ثم اضغط على الزر الأول، ستجد أنه كلما ضغطت على الزر ستزداد قيمة x بمقدار واحد وستُطبع القيمة الجديدة، اضغط على الزر الثاني ستجد أنه سيطبع قيمة x، أي أنه يعرف المتحول x.

ملاحظة:

بالنسبة للمتحولات العامة على مستوى المشروع، والمتحولات الساكنة فسأتى على شرحهما لاحقاً.

بني التحكم في Visual Basic :

إن بني التحكم في Visual Basic هي نفسها بني التحكم في QBasic لذلك لن نشرح هذه البنى بالتفصيل، وإنما سنكتفي بذكرها.

عبارة IF الشرطية:

تملك عبارة IF الشرطية في Visual Basic أربعة أشكال هي:
الشكل الأول:

```
IF condition THEN statement
```

أي في حال تحقق الشرط نفذ التعليمة.

الشكل الثاني:

```
IF condition THEN
    statement(s)
END IF
```

أي في حال تحقق الشرط نفذ مجموعة التعليمات.

الشكل الثالث:

```
IF condition THEN
    statement(s)1
ELSE
    statement(s)2
END IF
```

أي في حال تحقق الشرط نفذ مجموعة التعليمات 1 وإلا نفذ مجموعة التعليمات 2

الشكل الرابع:

```
IF condition1 THEN
```

```

    statement(s)
ELSEIF condition2 THEN
    statement(s)
ELSEIF condition3 THEN
    statement(s)
...
END IF

```

ويسمى هذا الشكل بعبارـة If متعددة المداخل، لأنه كلما تحقق شرط يتم تنفيذ مجموعة محددة من التعليمات.

عبارـة :SELECT CASE

وتستخدم بدلاً من عبارـة If متعددة المداخل:

```

SELECT CASE expression
CASE value1:
    statement(s)
CASE value2:
    statement(s)
CASE value3:
    statement(s)
....
END SELECT

```

أي اختبر قيمة التعبير expression واعتماداً على قيمته حدد مجموعة التعليمات التي يجب تنفيذها.

الحلقة التكرارية :FOR

```

FOR variable=start TO end
    statement(s)
NEXT variable

```

الحلقات التكرارية الأخرى:

الشكل الأول:

```

DO
    statement(s)
LOOP UNTIL condition

```

الشكل الثاني:

```

DO
    statement(s)
LOOP WHILE condition

```

الشكل الثالث:

```
DO UNTIL condition  
    statement(s)  
LOOP
```

الشكل الرابع:

```
DO WHILE condition  
    statement(s)  
LOOP
```

لقد انتهينا من دراسة اساسيات Visual Basic الأساسية، وسنعرض في الفصل الثالث كيفية بناء تطبيقات متنوعة باستخدام أدوات وأوامر Visual Basic، وسيتم شرح كيفية استخدام معظم أدوات وأوامر Visual Basic أثناء بناء هذه التطبيقات.