المحاضرة الأولى

لغة البرمجة Visual Basic 6

تعتبر لغة V.Basic6 من لغات البرمجة عالية المستوى حيث تقسم لغات البرمجة إلى :

1- لغات عالية المستوى Height Level Language
2- لغات منخفضة المستوى Low Level Language

والمقصود بلغات البرمجة عالية المستوى أن تعليماتها البرمجية قريبة من لغة التخاطب البشري وهذه بدورها تقسم إلى قسمين :

- لغات نصية
- لغات مرئية

ومن أمثلة لغات البرمجة النصية :باسكال ,بيسك,C++ .....أما اللغات المرئية :الفيجوال بيسك,Delphi, C#....

بينما المقصود بلغات البرمجة منخفضة المستوى أنها تتعامل مع الحاسب مباشرة دون الحاجة إلى وسيط أو ما يسمى المترجم (-Compiler)مثل لغات التجميع Assembly لغات (0, 1)

نعود إلى الفيجوال بيسك فهي تستخدم لتصميم برامج تعمل تحت نظام التشغيل Windows وبالتالي يجب على من يريد تعلم هذه اللغة أن يكون ملماً بطريقة التعامل مع نظام التشغيل Windows.

إن الفيجوال بيسك من اللغات المقادة بالحدث شأنها شأن معظم لغات البرمجة المرئية (ديلفي , فيجوال C++....)واللغة المقادة بالحدث هي لغة تعتمد على فكرة تجزئة البرنامج إلى برامج جزئية صغيرة تنفذ عند وقوع حدث معين مثل الضغط على أحد الأزرار ,تحريك مؤشر الفأرة فوق نافذة معينة ,مرور فترة زمنية .....وبالتالي يجب عند البدء بالبرنامج تحديد الأحداث وكيفية الاستجابة لكل منها (إذا ضغط زر كذا افعل كذا ,عند تحريك مؤشر الفأرة فوق النافذة هذه أفعل كذا ....)

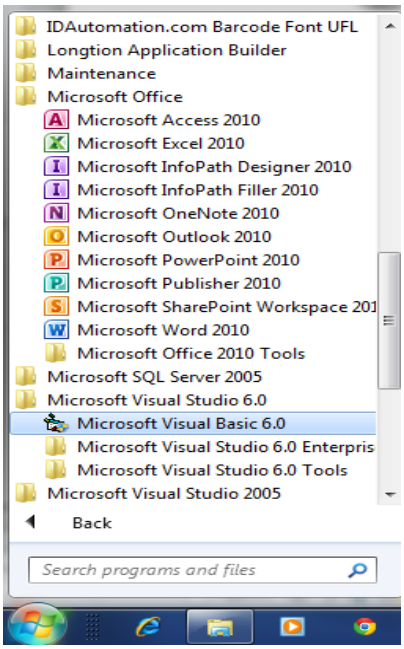
إن تنصيب هذه اللغة على الحاسب لا يختلف عن تنصيب أي برنامج من إنتاج شركة مايكروسوفت حيث يتم تنصيب مجموعة من المجلدات(مكتبات)مساعدة مثل مجلد إعداد التنصيب Setup مجلد الأدواتTools مجلد الرسومات Graphics وكلها ضمن المسار التالي :

C:\Program Files\Microsoft Visual Studio\common

ملاحظة :هذا المسار هو المسار الافتراضي للبرنامج.

تشغيل البرنامج :

بعد عملية التنصيب تضاف مجموعة من الاختصارات إلى قائمة بدء التشغيل وهي Microsoft Visual Basic6.0وداخل هذه المجموعة ستجد الاختصار Microsoft Visual Basic6.0 المسؤول عن تشغيل الفيجوال بيسك.



عند تشغيل البرنامج تظهر لدينا النافذة التالية :

من خلال هذه النافذة نلاحظ وجود ثلاثة علامات تبويب هي :

1- **New جديد** :تستخدم للبدء بمشروع جديد والذي يكون على عدة أشكال توضحها الأيقونات الموجودة ضمن هذه العلامة مثل :

- المشاريع القياسية Standard
- مشاريع تصميم مكتبات Dll
- مشاريع قواعد البيانات

المشروع	الشرح
Standard EXE	وهو يقوم بإنشاء برنامج تنفيذي بعد طلبك لذلك من (Make EXE) في قائمة. (File)
Activex EXE	وهو أيضاً يقوم بإنشاء ملفاً له الامتداد (DLL) وهو ملف ذو برامج دعيه مساعدة و هو لا يعمل بنفسه بل مع. exe
Activex EXE	وهو ينشئ ملفات لها امتداد EXE تعمل مع برامج إخرى لكي يعمل في شكل. (OLE)
Activex control	وهو يساعد على إنشاء ملف من نوع (ocx) و هي مهمة وتساعد على إحتواء واجهة مستخدم أو برامج فرعية.
Activex Document DLL	وهو يساعد على إنشاء ملف ذو امتداد (DLL) يقوم بتشغيل برامج على موقع في الانترنت
Activex Document exe	يقوم بإنشاء ملفات تظهر على الانترنت.
Add In	وهو خاص بالفيجوال بيسك حيث يمكنك إضافة واجهة مع ما ذكر سابقاً خاصة بك.
VB Application Wizard	وهو معالج تلقائي في هذه اللغة يساعدك على إنشاء نوافذ عدة دون كتابة سطر واحد من النوع
SS Aplication	نقصد بـ (ISS) إختصار لكلمة (Internet Information Server) و يقوم بالتحضير لتصميم (web class) وهو ما يسميه المستخدم صفحات. (HTML)
Data project	تضع مشروعاً لقواعد بيانات وتهيئة في بيئة متكاملة من أدوات الربط وصنع التقارير دون الحاجة لفتح برامج أخرى .
Vb wizard manager	و تبدو مشابهة لمعالجات (Microsoft) يمكنك إستخدامه
DHTML Application	تسهل عليك صنع صفحات (HTML) ديناميكية للإنترنت دون تعلم لغة. Java
Vb Enterprise	وهي تساعد على وضع جميع الأدوات المستخدمة بكثرة لقواعد البيانات و الاتصال ومن ثم

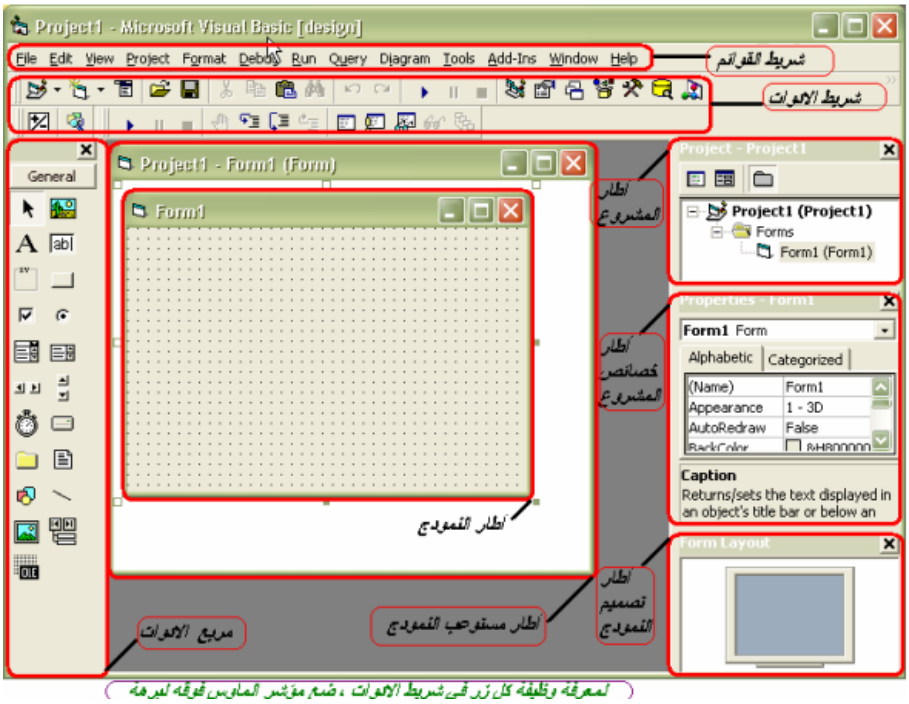
2- **علامة التبويب Existing** :وهي خاصة لفتح مشروع منشئ من سابق حيث يظهر مربع حوار يشبه إلى حد كبير مربع حوار فتح المعروف ضمن برامج Microsoft

3- **علامة التبويب Recent** :تستخدم لعرض قائمة المشاريع التي يتم العمل بها مؤخراً

بيئة التطوير

توفر لغة V.Basic أدوات قمة في الروعة مجتمعة تحت مسمى بيئة التطوير المتكاملة Integrated Development Environment وتختصر IDE حيث توفر لك كل ما تحتاجه

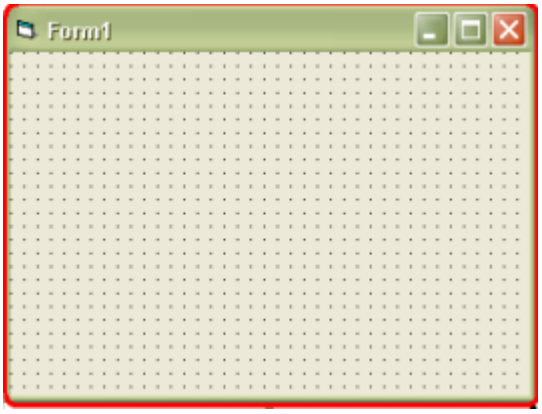
لتصميم نوافذ وكتابة شيفرات برمجية بل وتقدم لك خدمات خمس نجوم مثل خدمة التنقيح Debugging, إدارة ملفات المشروع, تحرير القوائم, إنشاء وتعديل قواعد البيانات



نوافذ بيئة التطوير :

بالنظر إلى النافذة الرئيسية لبية التطوير نلاحظ أنها من النوع متعدد المستندات Multiple Document Interface وتختصر MDI وستلاحظ احتوائها على نوافذ كثيرة , واول نافذة سنبدأ بها هي :

1- نافذة مصمم النماذج Form Designer:

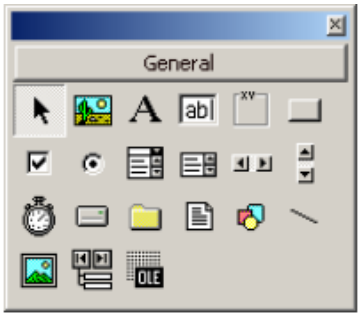


هذه النافذة تعتبر سر نجاح الفيجوال بيسك وهي أشهر نوافذ الفيجوال منذ الاصدارات الأولى ولها عنوان ابتدائي Form1 عن طريق هذه النافذة تقوم بعملية تصميم واجهة برنامجك إما بتعديل خصائصها أو وضع أدوات عليها باستخدام الفأرة .

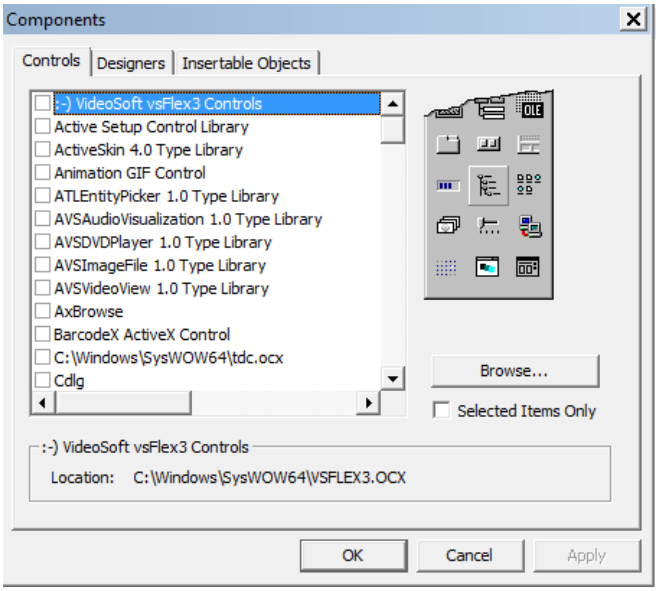
طريقة وضع الأدوات عليها أشبه ما تكون بعملية رسم مربعات كما في برنامج الرسام ,كما أن عمليات التحرير من نسخ ,قص, لصق مدعومة على جميع الأدوات التي تضعها على النافذة .

صندوق الأدوات :

يمكن القول ان صندوق الأدوات من الكائنات العائمة في بيئة التطوير , وهو شريط يحتوي على 20أداة قياسية والتي يمكن أن تضيفها إلى النماذج (واجهات المشروع).



ويمكن إضافة أدوات أخرى إلى هذا الصندوق بالنقر عليه بالزر الأيمن للفأرة واختيار الأمر Components ستظهر لدينا النافذة التالية :



نقوم باختيار الأداة التي تريدها ثم ننقر Apply او Ok وإذا أردت ادخال أدوات Active X أو Dll فقم باختيار الزر Browse ثم اختر الأداة .

عناصر شريط الأدوات :

1- Pointer وهذه الأداة تعيد شكل مؤشر الفأرة إلى شكل سهم بعد أن يكون على شكل إحدى الادوات(ولكن لا تشغل بالك بها فالمؤشر يعود إلى وضعه الطبيعي بعد الانتهاء من عملية وضع الأداة على النموذج)

2- Picture Box:وهي أداة تستخدم لوضع صور على النموذج ودمج هذه الصور إلى المشروع.

3- Label:وهي أداة تتيح لنا إضافة عناوين إلى الأدوات الموجودة على النماذج.

4- Text Box :تستخدم هذه الأداة لعرض أو إدخال البيانات إلى البرنامج مثل اسم الموظف ,تاريخ التعيين.....

5- Frame:أداة الاطار هذه تستخدم كحاضنة للأدوات التي تريد أن تكسبها نفس الخصائص(خاصية توريث)

6- Button(زر):يستخدم هذا الكائن لوضع أزرار على النموذج تستخدم لتنفيذ أوامر معينة مثل(حفظ , تعديل , إنهاء , إلغاء الامر.....)

7- List Box(قائمة)تستخدم هذه الأداة لوضع قائمة على النموذج فيها المجموعة من الخيارات والتي تتيح لنا اختيار أكثر من خيار بآن واحد.

8- CompoBox:نفس الأداة السابقة ولكن لا يظهر سوى خيار واحد يمكن استخدامه.

9- Timer(المؤقت):هي أداة تستخدم لتنفيذ أمر معين بعدة فترة زمنية محددة لها خاصية واحدة مهمة (Interval)سنتكلم عليها في حينها .

10- Drive List Box:وهي قائمة تظهر السواقات على شكل قائمة بما فيها السواقات المرنة والمضغوطة.

11- DirListBox:نفس عمل الأداة السابقة ولكن ينطبق الكلام على المجلدات وما تحتويه.

12- FileListBox:نفس الكلام السابق ولكن بالنسبة للملفات.

13- Image:صورة :تستخدم هذه الأداة لوضع صور على النماذج (تختلف بعض الشيء عن الأداة Picture Box.

14- Data :تستخدم هذه الأداة للتعامل مع قواعد البيانات .

15- OlE:تستخدم هذه الأداة للتعامل مع الكائنات مثل برنامج الوورد, برنامج الإكسيل.

16- ChekBox:صندوق الاختيار يستخدم لاختيار خيار واحد أو أكثر من مجموعة من الخيارات.

17- OptionButton:يستخدم لاختيار خيار واحد فقط من مجموعة من الخيارت

18- الأداتين HscrollBar وVscrollBar تستخدمان لوضع شريط تمرير أفقي أو عمودي على النافذة أو كليهما .

ملاحظة هامة:

يمكن تقسيم صندوق الأدوات هذا إلى مجموعة من علامات التبويب فقط بالنقر عليه بزر الفأرة الأيمن واختيار الخيار AddTab والذي يفتح لنا مربع حوار يطلب اسماً لهذه العلامة نعطيه الاسم ثم ننقر على OKنلاحظ أن الاسم قد توضع على اسفل صندوق الأدوات يمكن وضع الأدوات ضمن هذه العلامة بطريقة السحب والإلقاء .ولحذف هذه العلامة ننقر عليها باليمن ثم نختار الأمر DeleteTab.لا تخف الأدوات التي وضعتها فيها لا تحذف معها .

المحاضرة الثانية : واجهة البرنامج

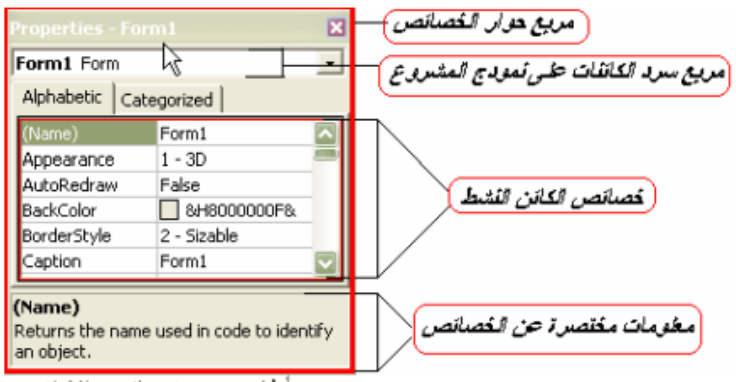
نافذة الخصائص:

بمجرد أن تنتهي من وضع الأداة على النموذج ستبحث في موضوع ضبط خصائصها والتي يمكن ضبطها بطريقتين:

1- من خلال نافذة الخصائص(مرحلة التصميم)
2- من خلال الشيفرة (مرحلة التنفيذ)

سنتكلم هنا عن نافذة الخصائص:

في اعلى النافذة يوجد قائمة تسمى في عالم الفيجوال بيسك CompoBox تمكننا من تحيد الكائن الذي نود عرض وضبط خصائصه مع العلم أنه يمكن تحديد الأداة من خلال النقر عليها وهي على سطح النموذج, ستلاحظ عندها ان محتويات نافذة الخصائص قد تغيرت حسب الأداة المحددة. كما يوجد علامتي تبويب يتم من خلالهما تحديد طريقة عرض الخصائص إما بترتيب أبجدي(Alphabetic)أو بترتيب حسب الفئة (Categorized).بالنسبة للجدول فإن العمود الأيسر يعرض الخصائص المتوفرة للأداة أما الأيمن فيعرض قيمة كل خاصية من هذه الخصائص .بعض هذه الخصائص يمكنك تعديلها مباشرة بكتابة قيمة عددية أو حرفية (Caption) والبعض الآخر عليك اختيار قيمة من عدة قيم كالخاصية (Visible)أو لون مجموعة لوح الألوان مثل الخاصية (BackColor)وهناك مربع حواري صغير في أقصى يمين العمود مكتوب عليه ثلاث نقاط "..."ويقصد به أن له صندوق حوار DialogBox له خيارات إضافية كالخاصية (Font).



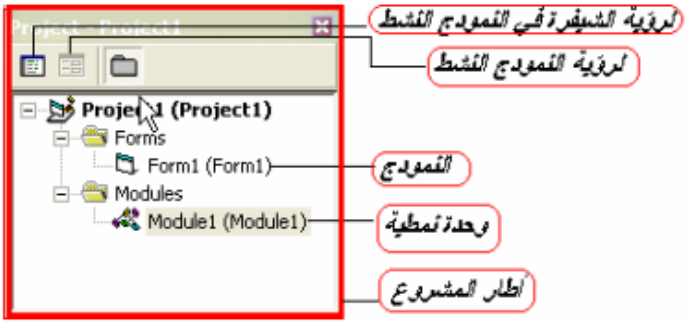
أطار مربع خصائص الاداة

نافذة مستكشف المشروع ProjectExplorer:

تزداد أهمية هذه النافذة بازدياد عدد الملفات التابعة لمشروعك, وهي الوسيلة الوحيدة التي تمكنك من عرض محتويات مشروعك مرتبة على شكل شجري برموز مختلفة. نلاحظ في هذه

النافذة كل الملفات المستعملة قي عملية البرمجة حيث يتم الوصول إليها من خلال زرين هما رؤية الكائنات ورؤية الشيفرة .

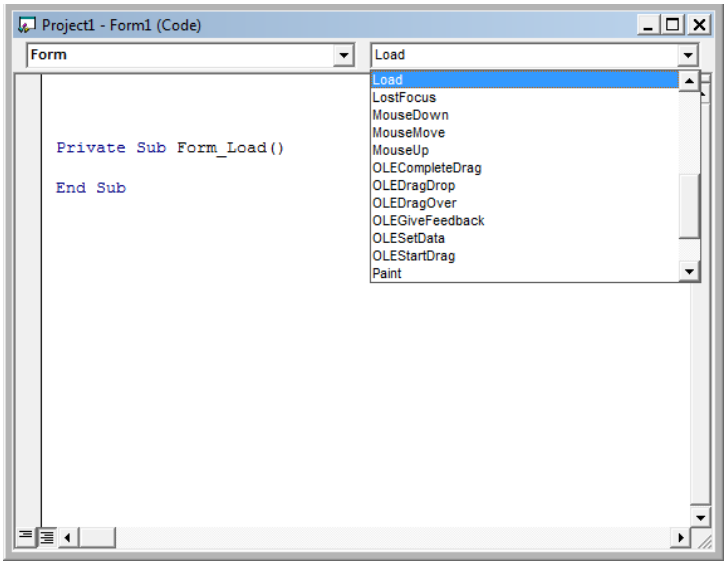
يسمى المشروع الذي يحتوي على لائحة بكل الملفات المساعدة في مشروع البرمجة ملف مشروع فيجوال بيسك وله الملحق (VBP).



نافذة محرر الشيفرة :

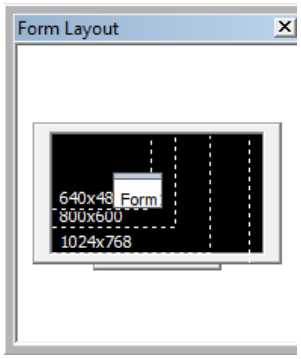
بما ان فيجوال بيسك لغة برمجة فبكل تأكيد عليك أن تكتب بعض العبارات البرمجية عن طريق نافذة محرر الشيفرة والتي تقدم محرر شيفرة ذكي جداً ومنسق كلمات يفتح نفس المبرمج لكتابة الشيفرة , ومن المناسب هنا أن ننوه ان بعض المبرمجين العرب الذين يستخدمون انظمة تشغيل تدعم مجموعة محارف الشيفرة الموحدة Unicode كنظام XP يواجهون مشاكل في كتابة الحروف العربية .

بإمكانك الوصول إلى نافذة محرر الشيفرة عن طريق النقر المزدوج على الأداة المطلوبة أو من خلال الضغط على مفتاح F7 من لوحة المفاتيح .في أعلى نافذة محرر الشيفرة يوجد قائمتين من نوع CompoBox اليسرى تعرض جميع الأدوات الموجودة في نافذة النموذج الحالية بالإضافة إلى النموذج نفسه وكذلك عبارة General المستخدمة في التصريح عن المتحولات البرمجية في النموذج وكذلك الاجراءات والدوال التي تقوم بإنشائها أما القائمة اليمنى فتعرض جميع الاجراءات والأحداث المرتبطة بما يتم اختياره في القائمة اليمنى



نافذة مخطط النموذج:

تتوضع هذه النافذة في الزاوية اليمنى السفلى من شاشة البرنامج , مهمة هذه النفاذة إعطاؤنا رؤية مبسطة عن موقع وحجم نافذة النموذج التي تصممها وقت التنفيذ من الشاشة,إلا أن الفائدة الأكبر هي مقارنة حجم نافذة النموذج مع الكثافات النقطية Resolutions المختلفة للشاشة ولمعرفة هذه الكثافات ننقر باليمين على النافذة ونختار الأمر Resolutions Guide من القائمة



كتابة البرنامج الأول

الخطوة الأولى: فكرة البرنامج

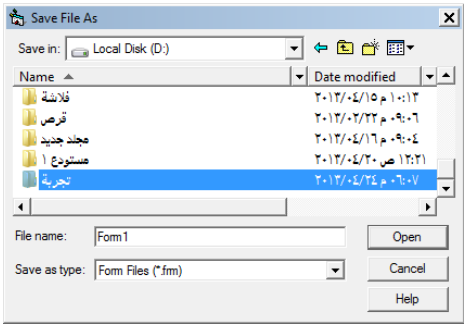
فكرة البرنامج لا تعتمد على نوع اللغة التي تكتب بها برنامجك ولا حتى بنظام التشغيل ,فمن البديهي أنك قبل أن تكتب برنامجك عليك معرفة ما الذي تريده من البرنامج.

ثم حدد المتطلبات والمشاكل التي قد تصادفك لآن حلها من المور الهامة في عملية كتابة البرنامج هذه العملية تسمى هندسة البرمجيات.

الخطو الثانية : إنشاء المشروع

يمكن الآن تشغيل بيئة التطوير واختيار المشروع نوع قياسي Standard Exe هذا النوع من المشاريع لبناء مشاريع قياسية تعمل تحت بيئة Windows وبعدها يمكن تكوين ملفاتها من النوع EXE.

احفظ المشروع (يمكن ذلك بعدة طريق)وذلك بفتح قائمة File واختيار الأمر SaveProjectومن المفضل حفظ المشروع بمجلد خاص به لأن المشروع قد يحتوي على عدة ملفات

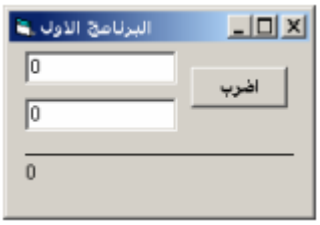


ملف المشروع يكون امتداه Vbp أما الملفات الأخرى فكل واحد منها له امتداه الخاص .

الخطوة الثالثة: تصميم الواجهة

نبدأ الآن بوضع الادوات على نافذة النموذج وذلك باستخدام الفأرة ومن الممكن تعديل خصائص هذه الأدوات في الوقت الحالي أو وقت التنفيذ .

انتقل إلى صندوق الأدوات وضع أداة عنوان Label وزر أمر Command Button وأداتي صندوق نص Text Boxواداة رسم خط Line ومن ثم رتب الدوات على سطح النموذج حتى تصبح على الشكل التالي :



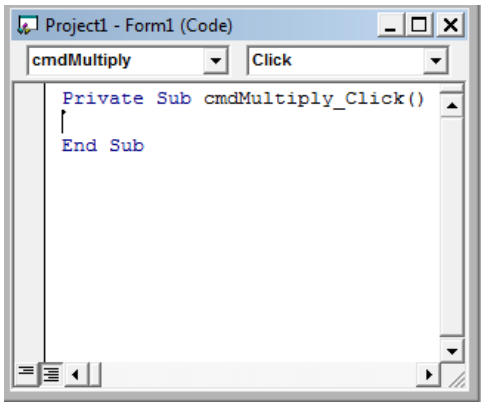
بعد ترتيب الأدوات ومحاذاتها سنبدأ بعملية تعيين خصائصها ,حدد الأداة بالنقر عليها وانتقل إلى نافذة الخصائص وقم بتعديل قيم خصائص الأدوات كما في الجدول التالي :

اسم الأداة	الخاصية	القيمة
نافذة النموذج	Name	frmMain

	Caption	البرنامج الأول
زر أمر	Name	Cmdmultiply
	Caption	اضرب
اداة العنوان	Name	LblProduct
	Caption	0
أداتي نص	Name	txtFirst
	Text	0
	Name	Txtsecond
	Text	0

الخطوة الرابعة كتابة التعليمات:

نقوم الآن بالنقر المزدوج على زر الأمر نلاحظ أن محرر الشيفرة قد ظهر وظهرت الشيفرة التالية



أي كود ستكتبه بين هذين السطرين سيتم تنفيذه إذا ما قام المستخدم بالنقر على رز الأمر بين هذين السطرين سنكتب الشيفرة التالية :

```
lblProduct.Caption = Val(txtFirst.Text) * Val(txtSecond.Text)
```

الخطوة الخامسة :التجربة والتعديل

اضغط على مفتاح F5 لتجريب البرنامج ثم أدخل اعدادا داخل صندوقي النص مثلا 5 ثم 10 ثم انقر على زر الأمر ستلاحظ أن اللافته أظهرت 50 نتيجة ضرب العددين ,قد تفرح في نتيجة ما توصلت إليه ولكن هل فكرت ماذا سيتصرف البرنامج إذا أدخل المستخدم نصاً في صندوق النص ولم يدخل أرقاماً حتماً البرنامج لن يعرف كيف سيتصرف في هذه المشكل هنا عليك تعديل الشيفرة السابقة إلى التالي :

```
If IsNumeric(txtFirst.Text) = True And IsNumeric(txtSecond.Text) = True
   Then

        lblProduct.Caption = Val(txtFirst.Text) * Val(txtSecond.Text )

 Else

      MsgBox " القيم المدخلة غير صحيحة"

 End If
```

الخطوة السادسة :الترجمة

المقصود بعملية الترجمة تحويل الملف إلى ملف تنفيذي EXE ويتم ذلك بفتح قائمة File واختيار الأمر Make MyFirstProgram.EXE

المحاضرة الثالثة

النماذج والأدوات

النموذج هو مصطلح خاص بالفيجوال بيسك مرادف للنافذة في نظام التشغيل ويندوز, وهو بالفعل نافذة تقوم أنت بتصميمها وتظهر في وقت التنفيذ كسائر نوافذ البرامج التطبيقية. أما الأدوات فهي كائنات توضع داخل النماذج وتحتضن فيها . ومن المهم أن نلفت النظر إلى وجود نوعين من الأدوات :

1- **الأدوات الداخلية** :وهي الأدوات العشرون التي توجد في صندوق الأدوات عند إنشائك مشروع قياسي Standard EXE.

2- **أدوات Active X Control**:وهي أدوات خارجية لها الامتداد OCX يمكن إضافتها عن طريق اختيار الأمر Components من القائمة Project أو بالنقر باليمين على صندوق الأدوات واختيار الأمر Components .

كل النماذج والأدوات تعتبر كائنات ولذلك هناك ثلاثة عناصر تتحكم بهذه الكائنات هي :

الخصائص(Properties),الطرق(Methods),الأحداث(Events)وكل كائن من هذه الكائنات له عناصره الخاصة به ولكن نلفت الانتباه إلى أن هناك عناصر مشتركة كثيرة بين النماذج والأدوات التي توضع عليها .

الخصائص المشتركة :

الخاصية هي قيمة تؤثر إما في شكل الكائن الخارجي مثل الخاصية BackColor أو الخاصية Font أو في سلوكه مثل الخاصية Enabled.

نافذة الخصائص هي المكان الذي يمكنك من تغيير قيمة الخاصية وقت التصميم أما وقت التنفيذ فتتغير الخصائص عن طريق كتابة الشيفرة اللازمة لذلك مثل التالي :

```
PictureBob1.BackColor=0

Label1.Caption="مرحباً بكم"
```

أو استخدام الكلمة المحجوزة With لتغيير مجموعة من الخصائص لكائن معين دفعة واحدة دون إعادة تكرار كتابة اسم الكائن :

```
With text1
        .text="نقابة المهندسين"
        .Font.Bold=True
        .BackColor=VBBlack
        .ForColor=VBWhite
End With
```

تتميز بعض الأدوات الداخلية بوجود خاصية افتراضية تغنيك عن كتابة هذه الخاصية بعد اسم الكائن فمثلاً خاصية Caption هي الخاصية الافتراضية للأداة Label عندها يمكن أن نكتب التالي :

```
Label1="نقابة المهندسين"
Text1="فرع حماة"
```

بالنسبة لنافذة النموذج يمكنك الوصول إلى خصائصها دون كتابة اسم النموذج باستخدام الكلمة المحجوزة Me أو حتى دون كتابتها نهائياً فكل العبارات التالية صحيحة :

```
Form1.caption="XXXX"
Me.Caption="XXXX"
.Caption="XXXX"
```

خاصية Name:

تعديل هذه الخاصية ممكن وقت التنفيذ فقط , وهي تمثل الاسم البرمجي للكائن أي هو الاسم الذي سنخاطب به الكائن أثناء كتابة الشيفرة وهنا ننصح بعدم الاعتماد على الأسماء الافتراضية التي يعطيها فيجوال بيسك مثل .....Form1,Form2,command1 وذلك لآنها تسبب تشويش المبرمج خاصة في البرامج الكبيرة وهنا لابد من ذكر الشروط الواجب أن نتبعها في تسمية الكائن :

- لا يبدأ برقم
- لا يزيد عن 40 حرف
- لا يحتوي على رموز خاصة ومسافات(@,$,&.....)

- أن لا يكون محجوزاً من قبل اللغة لاسم دالة أو اسم إجراء أو تابع .....

خصائص الموقع والحجم :

خصائص الموقع والحجم موجودة في جميع الأدوات القابلة للظهور(أي التي تحتوي على خاصية Visible)أما الأدوات الأخرى مثل أداة Timer فمن البديهي أن تكون هذه الخصائص غير موجودة .

خصائص الموقع للكائن(Left ,Top)تحددان موقع الزاوية العلوية اليسرى للأداة بالنسبة للأداة الحاضنة لها أو موقع الزاوية العليا اليسرى لنافذة النموذج بالمسبة للشاشة, ويمكن ضبط موقع الأداة عن طريق الشيفرة البرمجية مثل توسيط زر أمر وسط النموذج كالتالي :

```
Command1.Left=(Me.Scalewidth-Command1.width)/2

Command1.Top=(Me.ScaleHeight-Command1.Height)/2
```

خاصية Font:

جميع الأدوات التي تعرض نصوصاً على جبهتها تحتوي على هذه الخاصية والتي تحدد فيها نوع وحجم الخط المعروض على جبهة الأداة , ويمكن ضبط هذه الخاصية وقت التصميم من نافذة الخصائص أو وقت التنفيذ حسب الشيفرة التالية مثلاً :

```
With Text1
    .Font.name="Tahoma"
    .Font.Bold=True
    .Font.Size=20
End with
```

ومن المرونة التي يوفرها الكائن Font هي فكرة نسخ جميع خصائص الخط من اداة لأخرى كالتالي :

```
SetLabel1.Font=Label2.Font
```

وعند تعديل أي خاصية من خصائص الكائن Label2 يتم تعديل نفس الخاصية للكائن Label1 آلياً

خصائص اللون:

الخاصيتان BackColor,ForColor تمثلان لون الخلفية ولون الخط للأداة المحددة ,بعض الأدوات كـ ScrollBar لا تدعم هذه الخاصية فألوانها قياسية من ألوان النظام.

بعض الأدوات كـ Label لا تلحظ تغيير أي شيء إذا غيرت BackColor إلا إذا كانت قيمة الخاصية Backstyle هي 1 كذلك الحال مع الأداةCommandButton فلن تتمكن من مشاهدة التغيير اللوني لخلفية الأداة إلا إذا حولت قيمة الخاصية Styleإلى Graphical.

و الألوان تقسم إلى قسمين : ألوان قياسية Standard وألوان خاصةCustom ويفضل استخدام الألوان القياسية التي تمثل ألوان النظام ويندوز أما إذا رغبت في ألوان ثابتة لا تتغير مع ألوان النظام من لوحة التحكم فاستخدم الألوان الخاصة من نافذة الخصائص أو أحد الأكواد التالية :

```
Me.BackColor=VBGreen

Me.BackColor=VBBlue

Me.BackColor=RGB(255,0,0)
```

خصائص الجدولة :

معظم مستخدمي ويندوز يفضلون استخدام مفتاح الجدولة Tab للتنقل بين الأدوات , مع العلم أنك تستطيع معرفة الأداة المنتقاة من انتقال التركيز إليها مثلا شريط التمرير إن كان عمودياً او أفقياً يومض هذا الشريط أو يوضع مستطيل منقط حول الأداة المختارة .... إذا كل اداة قابلة للتركيز عليها لها خصائص الجدولة هي TabStop,TabIndex فالخاصية TabStop تحدد فيما إذا كانت الأداة يمكن استخدام المفتاح Tab معها أم لا بينما الخاصية الثانية TabIndex فتحدد ترتيب هذه الأداة مع مفتاح Tab مع العلم أن ترتيب الفهرسة يبدأ من الصفر .

ملاحظـــة: حتى لو كانت قيمة الخاصية TabStop هي False فإن المستخدم يستطيع أن ينقل التركيز على الأداة باستخدام الفأرة

مثـــــال :قد يرغب المبرمج أن يجبر المستخدم على كتابة شيء داخل صندوق النص وعدم مغادرته لصندوق النص هذا إلا بكتابته شيئاً فيه يمكن جعل ذلك ممكناً من خلال الكود التالي :

```
Private Sub Text1_LostFocus()
  If Trim(Text1.Text) = "" Then
    Text1.SetFocus
  End If
End Sub
```

الأداة checkbox:

تعطي هذه الأداة فرصة للمستخدم لتحديد اختيار معين إما بتفعيله أو لا , لذلك هذه الاداة سهلة التعلم ومهمة في نفس الوقت ,سنقوم بشرح كيفية استخدامها بواسطة مثال مع العلم أن قيمة التفعيل تحتجز في الخاصية Value والتي تأخذ إحدى القيم التالية :

Unchecked -0

Checked -1

Grayed-2 (رمادي)

مثال :

```
Private Sub Chcek1_Click
If Check1.Value=1 then
Image1.visible=True
End If
End Sub
```

الأداة OptionButton:

تسمى هذه الاداة في بعض الكتب Radio Button وعملها شبيه بالأداة checkbox إلا أن قيمة الخاصية Value هنا تكون trueأو False ولن تستطيع جعل قيمة Value تساوي True لأكثر من زر اختيار واحد فقط , لذلك يفضل وضع هذه الأزرار في اداة إطار Frame.

اداة القائمة ListBox:

تعرض هذه الأداة مجموعة من النصوص داخل صندوق يحوي على أشرطة تمرير scrollBars .الخاصية Sorted تقوم بفرز محتويات الأداة فرز تصاعدي بالاستناد إلى حروفها الأبجدية.تستطيع تعبئة محتويات الأداة في وقت التصميم عن طريق الخاصية List أو وقت التنفيذ باستخدام الطريقة AddItem

```
List1.AddItem "الأول"
List1.AddItem "الثاني"
```

ملاحظـة:

إذا كنت ستضيف مئات أو آلاف العناصر وقت التنفيذ فينصح بإخفاء الأداة مؤقتاً وبعد الاضافة تعيد إظهار الأداة من جديد , وذلك لأن الأداة تعيد رسم نفسها تلقائياً مع إضافة أي عنصر إليها مما يسبب بطئ الجهاز وارتعاش الأداة نفسها

```
List1.visible=False
For x=0 to 10000
  List1.AddItem x
Next
list1.Visible=True
```

العناصر الجديدة تضاف إلى نهاية سلسلة العناصر إذا كانت قيمة الخاصية Sorted هي False .كما يمكنك حذف عنصر من القائمة باستخدام الخاصيةRemoveItem أو حذف جميع العناصر بالطريقة Clear

```
List1.RemoveItem 0
List1.Clear
```

الخاصيةlistIndex :

تعود بقيمة العنصر المحدد في الأداة وتعود بالقيمة(-1)إن لم يكن هناك أي عنصر محدد أما الخاصية Text فتعود بنص العنصر المحدد

```
list1.ListIndex=0
Print List1.text
```

الخاصية listCount :

تعود هذه الخاصية بعدد جميع العناصر الموجودة والتي تستخدم بكثرة مع الخاصية List التي تمكنك من الوصول إلى العنصر

```
For x=0 to list1.listCout
  Print list1.list(0)
Next
```

بالنسبة للخاصية MultiSelect تمكن المستخدم من تحديد عدة عناصر متتالية في الاداة إذا كانت قيمتها 1-Simple أو عدة عناصر غير متتالية إذا كانت قيمة الخاصية 2-Extended.

اداة القائمة ComboBox:

معظم الخصائص والطرق للأداة listBox السابقة موجودة في الأداة ComboBox وذلك لأن الاداة ComboBoxهي الأداة listBox القياسية ولكن تحتوي على خانة نص اعلاها .تستطيع التحكم بعرض خانة النص بعدة طرق باستخدام الخاصية Style .

1- إذا كانت قيمة الخاصية Style تساوي 0-dropdown فإن أداة النص ستظهر مع سهم يؤدي النقي عليه إلى ظهور الجزء الثاني من الاداة .

2- إذا كانت قيمة الخاصية Style تساوي 1-Simple فكلا الجزأين سيظهر للمستخدم

3- أما إذا كانت قيمة الخاصية تساوي 2-dropDown list فهي نفس الخاصية أو القيمة الأولى سوى أن المستخدم لن يتمكن من الكتابة في صندوق النص .

مثال :

```
Private Sub Form1_Load()

Combo1.AddItem "مصطفى"

Combo1.AddItem "خالـــد"

End Sub
```

وبعد وضع العناصر في القائمة يمكن اختيار أحدها وتنفيذ مجموعة تعليمات تنفذ نتيجة لهذا الاختيار

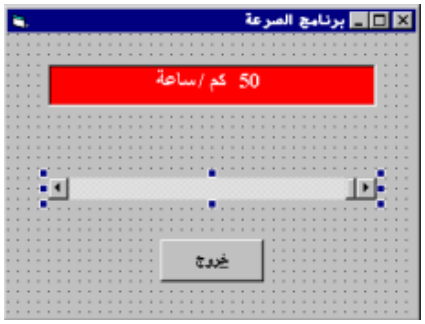
```
Private Sub Combo1_Click()

Select Case Combo1.ListIndex

Case 0
مجموعة تعليمات
Case 1
مجموعة تعليمات اخرى

End Select

End Sub
```

أشرطة التمرير ScroolBars:

تمكنك الأداتين HscrollBar و VscrollBar من محاكاة أشرطة التمرير المنتشرة في نوافذ وتطبيقات Windows .إبدا بالخاصيتين Min,Max لتحديد مجال القيم التي يمكنك قراءتها أو كتابتها عن طريق الخاصية Value والتي تمثل الموقع الحالي للمنزلقة المتحركة على شريط التمرير بعد ذلك حدد قيمة التغيير البسيط عن طريق الخاصية smallchange وهي مقدار التغيير في القيمة عندما يقوم المستخدم بالنقر على أحد أزرار أشرطة التمرير ,أما الخاصية LargeChange فهي مقدار التغيير في القيمة عندما يقوم المستخدم بالنقر على شريط التمرير نفسه.

برنامج السرعة :

يوضح المثال التالي كيفية استخدام شريط التمرير للحصول على قيمة معينة من المستخدم بحيث يظهر مؤشر شريط التمرير في مركز الشريط عند تشغيل البرنامج وتظهر عبارة كم/سا في مربع النص وعند تغيير موضع منزلقة شريط التمرير يظهر هذا التغيير في صندوق النص



كما جرت العادة سيكون هناك تمثيل مرئي للبرنامج والمرحلة الثانية هي كتابة الشيفرة

أولاً التمثيل المرئي :

القيمة	الخاصية	اسم الكائن
FrmPeed	Name	Form
برنامج السرعة	Caption	
TRUE	R To L	
CmdExit	Name	CommandButton
&خروج	Caption	
Hsbspeed	Name	HscrollBar

	Min	0
	Max	100
TextBox	Name	txtspeed
	Alignment	2-Center
	BackColor	Red
	ForColor	White
	Text	50كم/سا
	R To L	TRUE

نذهب الآن إلى نافذة الخصائص ونختار الخاصية Value لشريط التمرير ونعطيها القيمة 50 حتى توضع المنزلقة في منتصف شريط التمرير في بداة تشغيل البرنامج , ثم نختار الخاصية Min ونعطيها القيمة (0) وكذلك الخاصية Max نعطيها القيمة (100) الان لو ضغطنا على مفتاح F5 من لوحة المفاتيح سنرى شريط التمرير وقد وضع المنزلقة في منتصفه حرك المنزلقة إلى اليمين و اليسار لن ترى أي تغيير سيحصل (يمكنك تحريك المنزلقة باستخدام مفاتيح الأسهم لليمين واليسار)ولهذا ننقر على شريط التمرير الافقي مرتين للانتقال إلى مرحلة كتابة الكود.

ثانياً كتابة الكود:

في حدث change لشريط التمرير الأفقي نكتب العبارة التالية :

```
Txtspeed.text=Str(hsbspeed.Value)+"كم / سا"
```

عند التجريب تلاحظ أن مربع النص لا يتغير محتواه إلا بعد تحرير زر الفأرة ونحن نريد أن يتغير المحتوى اثناء تحريك المنزلقة إلى اليمين أو اليسار لهذا نضع العبارة التالية عند الحدث Scroll للكائن hsbspeed

```
Hsbspeed_change
```

أي نفذ الاجراء التالي عند انزلاق المنزلقة على الشريط .

التحكم في سير البرنامج :

المقصود بعملية التحكم في سير البرنامج هو تغيير سير تنفيذ التعليمات إذا تحقق شرط معين أو صادف حدوث حدث ما (ضغط المستخدم على زر Esc من لوحة المفاتيح مثلا) عندها يتم تنفيذ مجموعة تعليمات جديدة ,والجملة البرمجية المستخدمة للتحكم في سير البرنامج هي جملة IF.

هذه الجملة لا يستغني عنها أي مبرمج يعمل في أي لغة برمجية كانت ولهذا يمكن القول انها من اكثر العبارات البرمجية استخداماً هذه العبارة إما أن تنجز في سطر واحد أو عدة أسطر وهو الخيار المفضل .

امثلة :

```
1- IF X>0 THEN Y=0

2- IF X.0 THEN Y-0 ELSE Y=X
```

كما هو ملاحظ اننا لم نضع عبارة END IFفي نهاية الجملة البرمجية لأن عبارة IF وضعت على سطر واحد ولكن إذا كنا سنستخدم عبارة IF في عدة أسطر فيجب وضع عبارة END IF في نهاية التحقق من شرط معين .

أمثلة:

```
1- IF X=0 THEN
     Y=0
   END IF
2- IF M>0 THEN
     T=1
     ELSE
     T=-1
   END IF
3- IF M>0 THEN
     T=1
     ELSE IF M<0 THEN
```

```
T=-1
ELSE
T=0
END IF
```

الدالـــة IIF:

هي كاختصار لدالة IF.....ELSE كالتالي :

```
X=TEXT1.TEXT
MSGBOX IIF(X=7,"X=7","X<>7")
```

أي اجعل محتوى صندوق النص TEXT1.TEXT في المتحول X ثم أظهر رسالة بعد مناقشة حالة X إذا كانت =7فاكتب X=7 وإلا فاكتب X<>7.

التفرع باستخدام SELECT CASE:

تصلح عبارة الشرط IF إذا كان جواب الشرط احتمالين أو ثلاثة وأما إذا كنت تتوقع احتمالات كثيرة فمن المفضل أن تستخدم عبارة SELECT CASE يليها اسم المتغير الذي سيتم اختباره تأتي بعد ذلك احتمالات CASE بعد كل منها تأتي احدى قيم المتغير الذي ستتم مقارنته ثم تعقبها التعليمات التي ستنفذ إذا كان الشرط صحيحاً أو كان المتغير بهذه القيمة . واخيراً تأتي عبارة CASE ELSE ومعناها إذا كان المتغير لا يساوي أياً من القيم السابقة أو إذا لم يكن الشرط صحيحاً فإن التعليمات التي تلي ELSE هي التي تنفذ :

```
SELECT CASE TEXT1.TEXT
    CASE "محمد"
MSGBOX "مرحباً يا محمد"
   CASE "خالد"
MSGBOX "مرحباً يا خالد"
  CASE ELSE
MSGBOX "الاسم المدخل غير موجود"
END SELECT
```

ملاحظـــة :

إذا كنا سنستخدم عبارات مقارنة بعد Case فلا بد من استخدام IS معها

المحاضرة الرابعة:

البيانات والمتغيرات

مقدمة :

البيانات في أي لغة من لغات البرمجة بما فيها الفيجوال بيسك إما أن تكون متغيرات VARIABLES او ثوابتCONSTNTS , وفهم التعامل مع المتغيرات من المسائل الضرورية التي تمكنك من اختيار الأنواع المناسبة سواء لإرسالها إلى العمليات الحسابية أو الدوال أو الاجراءات .....

المتغير :هو مكان في الذاكرة يتم تخصيصه لك لتقوم بتخزين معلومات تريدها وذلك بعد أن تضع له عنوان (اسم)كما يمكن تغيير المعلومات الموجودة فيه بمعلومات أخرى في نفس المكان مع ثبات هذا العنوان .

مثال :إذا أردت أن تسأل عن اسم العميل الذي سيدخله المستخدم فإن اسم العميل قيمته متغيرة لأنك لا تعرف من هو العميل الذي سيقع عليه اختيار المستخدم في هذه الحالة تستخدم متغير اتضع فيه اسم العميل كمثال:

```
Nameclient=INPUTBOX("ادخل اسم العميل")
```

في هذا المثال سيعرض البرنامج رسالة تطالب فيها المستخدم ادخال اسم العميل الذي تريد البحث عنه عن طريق صندوق ادخال ويبقى المتغير محتفظا بنفس الاسم إلى أن يقوم المستخدم بإدخال اسم عميل آخر يطلب البحث عنه

اما **الثابت** فمثله مثل المتغير تماماً إلا أنك لن تحتاج لتغيير محتوى العنوان أي يبقى محتفظاً بقيمته أثناء تنفيذ البرنامج ,بمعنى إذا كان عملك يرتبط بمجموعة من العمليات الحسابية تتطلب التعامل مع قيمة ثابتة طوال فترة عمل البرنامج نصرح عنها بالشكل التالي ك

```
CONST PI=3044 AS DOUBLE
```

فما الفائدة من ذلك ؟ أثناء كتابة تعليمات البرنامج لن تحتاج إلى كتابة الرقم 3.14 كلما احتجت إليه بل يكتفى بكتابة اسم الثابت وهو هنا في مثالنا(PI) كما توجد فائدة أخرى عظيمة هي أنك إذا استخدمت ثابتا ما في مجموعة من العمليات الحسابية باسمه مثلاً(PI) واردت ان تغير قيمة هذا الثابت فإن جميع العمليات الحسابية التي يدخل فيه هذا الثابت سوف تتغير قيمها .

أنواع المتغيرات :

توجد في الفيجوال بيسك انواع كثيرة من المتغيرات نذكر منها :

الاسم	الحجم في الذاكرة	المدي
Byte	1 byte	0 to 255
Boolean	2 bytes	True or False
Integer	2 bytes	-32,768 to 32,767
Long(long integer)	4 bytes	-2,147,483,648 to 2,147,483,647
Single	4 bytes	-3.402823E38 to -1.401298E-45
Double	8 bytes	-1.79769313486232E308 to -4.94065645841247E-324 للقِيَمِ السلبيةِ ; 4.94065645841247E-324 to 1.79769313486232E308 للقِيَمِ السلبيةِ
Currency	8 bytes	-922,337,203,685,477.5808 to 922,337,203,685,477.5807
Decimal	14 bytes	+/- 79,228,162,514,264,337,593,543,950,335 بدون فاصلة عشريةِ:-/+ 7.9228162514264337593543950335ب28 مكانِ إلى يمين العشرين؛ الأصغر عدد غير الصفر +/-0.0000000000000000000000000001
Date	8 bytes	January 1, 100 to December 31, 9999
Object	4 bytes	أيّ جسمِ
String (fixed-length)	Length of string	1 الي 65,400تقريباً

Variant(with numbers)	16 bytes	أيّ قيمة عددية مضاعفة (2 4 8 16)
Variant(with characters)	22 bytes + string length	المدي نفسة

الشروط الواجب توفرها في اختيار اسم للمتغير :

1- يجب أن لا يزيد اسم المتغير عن 40محرف

2- يجب أن يبدأ بحرف

3- لا يحتوي على مسافات أو رموز أو نقاط ....

4- ألا يكون كلمة محجوزة في لغة الفيجوال بيسك .

ملاحظة :يوجد نوعان مت المتغيرات نوع نصي STRING متغير ثابت الطول ومتغير غير ثابت الطول مثال ذلك : Dim tab1 as string*10

كيفية الاعلان عن المتغيرات :

عند استخدام المتغير في الفيجوال بيسك فإن الفيجوال تتعرف على المتغير بمجرد استخدامه في الكود وهذه الطريقة مريحة ولكنها خطرة ويكمن خطورتها أنك إذا أخطأت في كتابة اسم المتغير فإن الفيجوال بيسك سيعتبره متغيراً جديداً لذلك هناك خيار يجب تفعيله لعدم حدوث هكذا اخطاء هذا الخيار هو OPTION EXPLICIT,حيث أن هذا الخيار يفرض علينا الاعلان عن المتغير حتى لا يحدث أي خطأ في كتابة اسم المتغير أو اسناد قيمة إلى متغيرات لم يعلن عنها مسبقاً مما يؤدي إلى توقف البرنامج عن العمل, وهذا التعبير يظهر في أعلى نافذة كتابة الكود وإذا لم يكن ظاهراً نفعله بالطريقة التالية :

TOOLS ➔ OPTIONS ➔EDITOR ➔REQUIRE VARIABLE DECLARATIO

قابلية الرؤية و عمر المتغير :

قابلية الرؤية وعمر المتغير أحد المبادئ الضرورية في جميع اللغات البرمجية ,فقابلية الرؤية والمدى للمتغير تمثل قدرة البرنامج على الوصول إلى المتغير واستخدامه ,فالمتغير (X) الموجود في الشيفرة التالية لا يمكن الوصول إليه خارج نطاق الاجراءMysub1

Sub Mysub1()

```
    Dim X As Integer
    X=20
End Sub
Sub Mysub2()
    Print X          هنا قيمة المتغير X لا تمثل قيمة X السابقة
End Sub
```

اما عمر الحياة للمتغير فتمثل الفترة التي يظل فيها المتغير محتفظاً بقيمته فالمتغير X الموجود في الشيفرة السابقة سينتهي ويزال تلقائياً من الذاكرة بمجرد الخروج من الاجراء SUB.

المتغيرات المحلية الديناميكية :

المتغيرات المحلية الديناميكية هي متغيرات تولد مع السطر الذي تعلن عنها فيه داخل الاجراء وتنتهي بعد نهاية الاجراء مباشرة ,ويتم تحرير المساحة التي حجزها المتغير في الذاكرة .

بالنسبة لقابلية الرؤية لن تستطيع الوصول إلى هذه المتغيرات إلا في نفس الاجراء الذي صرح فيه المتغير .تستخدم الكلمة المحجوزة Dim للتصريح عن المتغير ثم يأتي بعدا اسم المتغير ثم يتم تحديد نوع البيانات التي سيحتويها المتغير على الشكل التالي:

```
Dim VarName As DataType
```

أمثلة:

```
Dim Fname As String
Dim Age As Integer
```

المتغيرات المحلية الاستاتيكية :

قابلية رؤية المتغيرات المحلية الاستاتيكية هي مثل قابلية رؤية المتغيرات المحلية الديناميكية إي لن تتمكن من الوصول إليها إلا من داخل الاجراء المصرح عنها فيه ,بينما حياة المتغير الاستاتيكي فهو يبقى محتفظاً حتى نهاية البرنامج ,وللتصريح عن متغير ستاتيكي نستخدم الكلمة Static على النمط التالي :

```
Static اسم المتغير As نوع البيانات
```

مثال :

```
Static pi As Inreger
```

الكلمة المحجوزة Static لا تطبق إلا على المتغيرات المحلية فلا تحاول تطبيقها على المتغيرات العامة أو على مستوى الوحدة فهي بطبيعتها ستاتيكية

المتغيرات على مستوى الوحدة :

القصد من كلمة الوحدة هي الوحدة البرمجية Module المتمثلة في ملف برمجي Bas أو نافذة النموذجForm أو فئة Class ..إلخ ومكان التصريح عن متغير على مستوى الوحدة في منطقة الاعلانات العامة للوحدة أي خارج الاجراءات ,قابلية الرؤية لهذا النوع من المتغيرات يكون لجميع شيفرات الوحدة في حالة استخدام الكلمة المحجوزة Dim أو private:

```
Dim sname As String

Dim Age as Integer

Sub Date()

   Sname="مصطفى الخالد"

   Age=99

End Sub

Sub PrintData()

   Print sname

  Print Age

End Sub
```

أما إذا كنت تريد تعريف متغيرات عامة قابلة للوصول إليها من جميع أنحاء المشروع فالكلمة المحجوزة Public تفي بالغرض:

```
Public Current as string      يكتب هذا الكود في نافذة النموذج

Public  NumberOfUsers As integer

Private Sub Form_Load()

   If NumberOfUsers <= 0 Then

      Exit Sub

   Else
```

```
        Me.Caption = Form1.sCurrentUser
    End If
End Sub
```

هنا هذا النوع من التغيرات يظل محتفظاً بقيمته حتى انتهاء تنفيذ البرنامج

أمر الإعلان عن المتغيرات :

الأمر Dim : يستخدم للتعريف عن متغير ديناميكي Dynamic Variable ضمن الاجراء ويكون مجال رؤية هذا المتغير داخل الاجراء فقط عمر الاجراء أي عندما ينتهي الاجراء ينتهي معه المتغير المحلي ويصبح لا قيمة له .

المحاضرة الخامسة :

صناديق الحوار

عندما تفتح أي برنامج فإنك سوف تحفظ عملك في النهاية ,فإذا طلبت ذلك يظهر لك مربع حوار حفظ ,وإذا فتحت أي برنامج وتريد فتح ملف سوف يظهر لك أيضاً صندوق حوار خاص بعملية الفتح وهو شبيه بصندوق الحفظ ,هذه الصناديق تسمى Common Dialog Box والتي هي عبارة عن أدوات تضاف إلى صندوق الأدوات(ملفاتOCX)بنفس طريقة إضافة أي أداة إلى صندوق الأدوات .

هناك طريقتان لاستخدام هذه الأداة :

1- باستخدام أداة جاهزة MicroSoft Common Dialog Control6.0

2- باستخدام إجراءات API وهذه العملية صعبة لا نتطرق إليها حالياً.

لإضافة هذه الأداة إلى صندوق الأدوات ننقر باليمين فوق صندوق الأدوات ثم نختار الأمر Components ثم نبحث عن الأداة MicroSoft Common Dialog Control6.0 ثم ننقر OK فتظهر أيقونة الأداة ضمن صندوق الأدوات وستأخذ الاسم Common Dialog1 كاسم افتراضي .

مثال :

افتح مشروع جديد وضع على النموذج الأداة Image وزر Command وكذلك صندوق حوار Common Dialog وافتح نافذة الكود واكتب التالي في حدث Click لزر الأمرCommand

```
CommonDialog1.DialogTitle="فتح صورة"

CommonDialog1.Filter="صورGIF|*.gif|صورJPG|*.JPG|صورBMP|*.Bmp|
كافة الملفات|*.*"

CommonDialog1.ShoeOpen

IF CommonDialo1FileName="" Then Exit Sub

Image1.Picture=loadPicture(Commondialog1.FileName)
```

شرح الكود :

في السطر الأول أعطينا صندوق الحوار عنوان فتح صورة

في السطر الثاني حددنا حسب الخاصية Filter ما هي الملفات المطلوب عرضها في صندوق النص الخاص باسم الملف ضمن صندوق الحوار .

في السطر الثالث تم فتح مربع الحوار

في السطر الرابع إذا كان صندوق النص الخاص بصندوق الحوار فارغ فاخرج من البرنامج

في السطر الخامس تم إظهار الصور المحددة من صندوق الحوار ضمن الأداة Imageالموجودة على سطح النافذة .

ملاحظـــة:يفضل أن نضع الخاصية Stretch الخاصة بالأداة Image على القيمة True.

مثال :

تلوين نص ضمن صندوق نص باختيار لون من صندوق الألوان ك

نضع على نموذج مربع نص ونكتب فيه أي عبارة نريدها ثم نضيف زر أمر Commonad ونضع له عنوان (تلوين)ونفتح نافذة كتابة الكود ونكتب فيها :

```
CommonDialog1.ShowColor

Text1.forColor=CommonDialog1.Color

IF commonDialog1.Color=0 then Exit Sub
```

صندوق الرسائل MsgBox:

الشكل العام لهذا الصندوق :

```
MsgBox"نص الرسالة","العنوان",Symbol"
```

نص الرسالة عبارة عن سلسلة نصية (حرفية)تمثل الرسالة التي نريدها أن تظهر في صندوق الرسائل .

العنوان :عبارة عن سلسلة حرفية (String)تمثل عنوان صندوق الرسالة في شريط العنوان.

Symbol:رموز تمثل قيماً صحيحة أو ثابتاً حرفيـــاً .

امثلة:

```
MsgBox"حدث خطأ","خطأ",VBcritical"

MsgBox"أكتب رسالتك هنا","تعليمات",VBMsgBoxHelpButton"

MsgBox"مرحبا بكم","العنوان",VBYesNo+VBQuestion"
```

```
Dim x,a,y as string
X="محمد"
A="هل تريد الخروج يا"&Vbcrlf
a=a & " ( " & x & ") " & Vbcrlf
a=a & "نهائياً"
 Y=MsgBox(a,Vbcritical+VbyesNo+VbmsgBoxRight,"خروج")
If Y= yes Then End
```

برنامج القائمة المتغيرة :

سنكتب الآن برنامجاً يدعى برنامج القائمة المتغيرة ,حيث يوضح كيفية إضافة بنود أو إزالتها من القائمة خلال مرحلة التنفيذ ,دعنا في البداية نحدد ما يفترض أن يقوم به برنامج القائمة المتغيرة ثم نتابع بعد ذلك كتابة البرنامج .

يظهر عند بدء تشغيل البرنامج شريط قائمة يتضمن العنوان تغيير

تحتوي القائمة تغيير في بداية التشغيل على ثلاثة بنود هي على التوالي :

- إضافة بند
- إزالة بند
- خروج
- خط فاصل

كما في الشكل التالي :

والذي تلاحظ فيه أن بند إزالة يظهر بلون باهت .يتم إضافة بند جديد إلى القائمة تغيير كلما اخترت البند إضافة مما يؤدي إلى تغيير القائمة لتبدو كما في الشكل التالي بعد اختيار إضافة بند ثلاث مرات ويتم حذف البند الأخير من القائمة تغيير كلما اخترت البند إزالة بند.

يظهر عند اختيار أي من العناصر التي أضفتها رسالة تحمل اسم البند الذي اخترته فمثلاً عند اختيار البند (2)تظهر الرسالة المبينة في الشكل التالي :

التصميم المرئي لبرنامج القائمة المتغيرة :

- ابدأ مشروعاً من النوع التنفيذي Standard Exe
- احفظ نموذج المشروع باسم Grow.Frm وملف المشروع باسمGrow.VBP
- أنشئ النموذج وفق الجدول التالي :

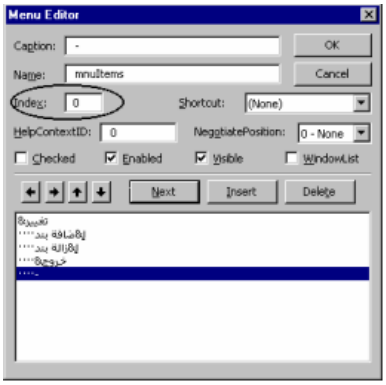
القيمة	الخاصية	اسم الأداة
FrmGrow	Name	Form
برنامج القائمة المتغيرة	Caption	
True	R TO L	
White	BackColor	

سنبدأ الآن بربط القائمة مع النموذج frmGrow:

- اختر النموذج Frmgrow
- اختر البند MenuEditor من قائمة Tools
- صمم القائمة وفق الجدول التالي :

العنو ان	الاسم
&تغيير	MnuGrow
....إ&ضافة بند	MnuAdd
....إ&زالة بند	Mnuremove
....&خروج	MnuExit
.... -	mnuitems

ضع الإضاءة فوق آخر بند من القائمة (الخط الفاصل)ثم اكتب القيمة صفر في مربع النص Index ثم انقر OK في الإطار MenuEditor.انتهت الآن مرحلة التصميم المرئي لبرنامج القائمة المتغيرة .



إنشاء مصفوفة عناصر تحكم القائمة :

لقد أدى إسناد القيمة صفر للخاصية Index التابعة للخط الفاصل إلى ما يدعى بمصفوفة عناصر تحكم القائمة سوف تدعى هذه المصفوفة بالاسم MnuItems لأنك كتبت الاسم MnuItems في الخاصية Name .

تتألف المصفوفة MnuItems حتى هذه اللحظة من بند واحد هو الخط الفاصل Separator Bar أي بمعنى أن البند(0) MnuItems هو البند الفاصل .

سيضيف برنامجنا مع كل اختيار للبند إضافة بند من القائمة تغيير أثناء تنفيذ البرنامج بنداً جديداً إلى المصفوفة MnuItems ونتيجة لذلك سيظهر المزيد من بنود القائمة تحت بند الخط الفاصل .

إدخال نص برنامج القائمة المتغيرة :

- اكتب النص التالي في قسم التصاريح العامة :

```
Dim glastelement as integer
```

- اكتب النص التالي في الإجراء Form_Load():

```
Glastelement =0
mnuRemove.enabled=False
```

- اكتب النص التالي في الإجراء mnuAdd_Click()

```
Glastelement= glastelement+1
Load mnuItems(glastelement)
'(mnuItems) إضافة عنصر جديد للمصفوفة
mnuItems(glastelement).Caption="البند"+  Str(glastelement)
```

بما أنه قد تم إضافة بند جديد للقائمة المتغيرة لذلك يجب تفعيل البند إزالة يند:

```
mnuRemove.Enabled=True
```

اكتب النص التالي في الإجراء mnuRemove_Click() :

```
Unload mnuItems(glastelement)
Glastelement= glastelement-1
IF glastelement=0 then
  mnuRemove.Enabled=False
End IF
```

اكتب النص التالي في الإجراء mnuItems_Click():

التصريح عن المتحول الذي سيحوي الرسالة

```
Dim myMsg As String
```

التصريح عن المتحول الذي سيحوي عنوان الرسالة

```
Dim mytitle As String
```

عرض الرسالة التي تبين البند الذي اختاره المستخدم

```
Mytitle="برنامج القائمة المتغيرة"
myMsg="لقد اخترت البند"+Str(Index)
MsgBox mymsg,VBokonly +VBMsgBoxRtlReading+VBMsgBoxRight,_
Mytitle
```

اكتب النص التالي في الإجراء mnuExit_Click()

```
End
```

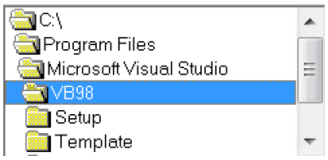
إحفظ المشروع الآن باختيار البند SaveProject من القائمةFile.

يركز هذا الفصل على استخدام عناصر تحكم نظام الملفات لكتابة برنامج يمكّن المستخدم من اختيار ملف ما من أي محرك أقراص .

توجد ثلاثة أنواع من عناصر التحكم هذه وهي:

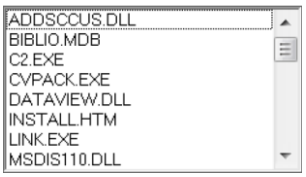
- **أداة عرض السواقات Drive List Box**: 

- وهي أداة لها شكل مربع سرد وتحرير ComboBox يظهر فيها أقسام القرص الصلب والأقراص المحمولة (المرن – CD)
- **أداة عرض المجلدات Directory List Box**:



وهي عبارة عن أداة صندوق قائمة ListBox تقوم بعرض المجلـــدات Folders في مسار معين تحدده أنت

- **أداة عرض الملفات File List Box**:



وهي عبارة عن أداة صندوق قائمة ListBox تقوم بعرض الملفاتFiles في مسار معين

الربط بين هذه الأدوات:

المطلوب عند تغيير محرك الأقراص تظهر المجلدات المحفوظة عليه وعند دخولنا إلى مجلد تظهر كافة الملفات التي بداخله .فنبدأ بمحرك الأقراص Drive1 في الحدث Change ونكتب الكود التالي :

```
Dir1.Path=Drive1.Drive
```

بمعنى إذا تغير محرك الأقراص Drive1 فإن قائمة المجلدات Dir1 ستتغير تبعاً للقرص الظاهر وبنفس الطريقة بالنسبة لتغيير المجلداتDir1 فعند النقر المزدوج على مجلد معين فإن الملفات التي ستعرض في قائمة الملفات هي الملفات المحفوظة بداخل ذلك المجلد ,لذلك سنكتب الكود التالي في الحدث Change الخاص بالأداة :Dir1

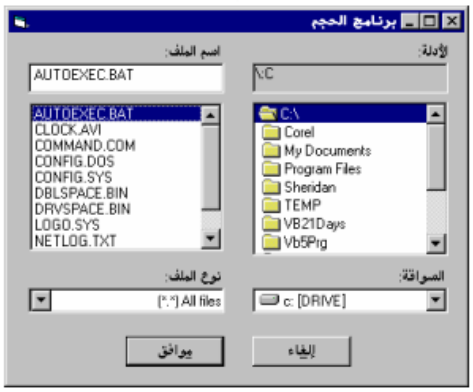
```
File1.Path=Dir1.Path
```

تستخدم هذه الأنواع جنباً إلى جنب في برنامج نموذجي يسمح للمستخدم اختيار الملفات من محركات الأقراص المختلفة,عندما يرغب المستخدم اختيار ملف ما حيث يمكن للمستخدم اختيار الملف المطلوب بانتقاء محرك الأقراص المناسب من أداة عرض محركات الأقراص ثم انتقاء دليل معين من أداة عرض الأدلة وأخيراً اختيار الملف من أداة عرض الملفات

برنامج الحجم :

سنكتب برنامجاً ندعوه برنامج الحجم يتضمن ثلاثة عناصر تحكم نظام الملفات وتستطيع استخدامه لاختيار ملف ما من محرك الأقراص وإظهار حجم الملف المنتقى.يفترض في هذا البرنامج إنجاز ما يلي :

- إظهار نموذج على الشاشة لاختيار الملفات عند تشغيل البرنامج حسب ما يبينه الشكل التالي :



يحتوي إطار برنامج الحجم على مربع سرد وتحرير يدعى نوع الملفات ويقع تحت مربع سرد الملفات ويمكّنك من اختيار نوع ملف محدد من لائحة أنواع الملفات الجاهزة .

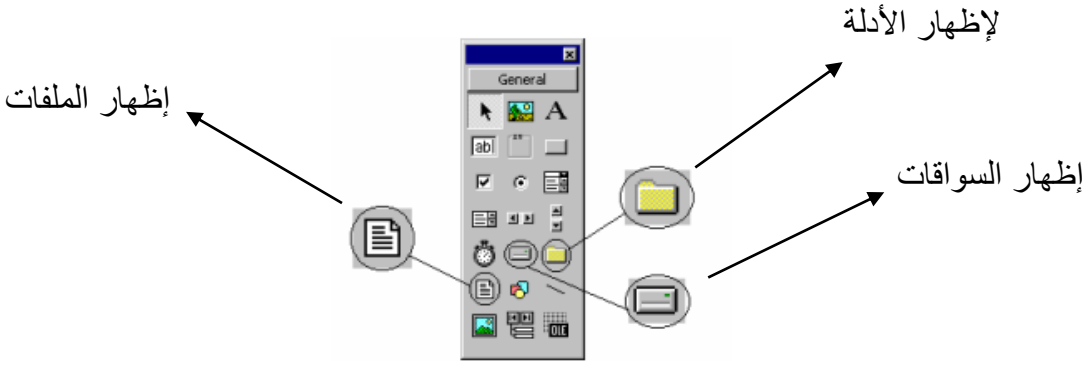
- إظهار لائحة محركات الأقراص المتوافرة وتمكين المستخدم من اختيار السواقة المناسبة .
- إظهار لائحة الملفات الموجودة حالياً في الدليل الذي اختاره المستخدم من مربع سرد الأدلة .

- إظهار اسم الملف في مربع النص لدى اختياره من مربع لائحة الملفات .
- إظهار حجم الملف المنتقى عند نقر الزر موافق

- إنهاء تنفيذ البرنامج لدى الضغط على زر Esc
- إظهار الملفات المنتمية إلى النوع الذي تختاره من مربع سرد وتحرير (نوع الملفات)فقط فمثلاً عند اختيار الملفات النصية (*.TXT)تظهر فقط الملفات النصية TXT في مربع سرد الملفات

التمثيل المرئي لبرنامج التحكم :

سنستخدم الأدوات التالية المبينة في الشكل الآتي والتي تستخدم في إنشاء مربعات خاصة بسرد أنواع الأقراص الموجودة وكذلك الأدلة ثم الملفات



- أنشئ مشروعاً جديداً من النوع Standard EXE
- أنشئ النموذج وفقاً للجدول التالي :

القيمة	الخاصية	الكائن
Frmsize	Name	Form
برنامج الحجم	Caption	
True	R TO L	
CmdOK	Name	CommandButton

&موافق	Cption	
True	R TO L	
CmdCancel	Name	CommandButton
إل&غاء	Caption	
True	R TO L	
dirDirectory	Name	Directory List Box
drvDrive	Name	Drive List Box
filFile	Name	File list Box
txtFileName	Name	TextBox
فارغ	Text	
cbofileType	Name	CommboBox
2-Dropdownlist	Style	
lblFileName	Name	Label
اسم الملف	Caption	
True	R To L	
lblFileType	Name	Label
نوع الملف	Caption	
True	R To L	
lblDirctory	Name	Label
الأدلة :	Caption	
True	R To L	

lblDirName	Name	Label
فارغ	Caption	
True	R To L	
1-Fixedsingle	BorderStyle	
lblDrive	Name	Label
السواقة :	Caption	
True	R To L	

إدخال نص برنامج الحجم :

- يجب أن تكون العبارة Option Explicit ظاهرة في قسم التصاريح العامة لماذا؟؟؟
- اكتب النص التالي في الإجراء Load Form()

```
cboFileType.AddItem "All files (*.*)"
cboFileType.AddItem "Text files(*.TXT)"
cboFileType.AddItem "Doc files (*.DOC)"
cboFileType.ListIndex = 0
lblDirName.Caption = dirDirectory.Path
```

- اكتب الإجراء التالي في drvDrive_Change( ):

```
On Error GoTo DriveError
dirDirectory.Path = drvDrive.Drive
Exit Sub
DriveError:
 MsgBox "خطأ في السواقة !", vbExclamation, "خطأ"
 drvDrive.Drive = dirDirectory.Path
```

- أكتب النص التالي في الإجراء dirDirectory_Change( ):

```
filFiles.Path = dirDirectory.Path
lblDirName.Caption = dirDirectory.Path
```

- اكتب الإجراء التالي في الحدث filFiles_Click( ):

```
txtFileName.Text = filFiles.filename
```

- اكتب الإجراء التالي في الحدث cboFileType_Click( ) :

```
Select Case cboFileType.ListIndex
Case 0
```

```
   filFiles.Pattern = "*.*"
Case 1
   filFiles.Pattern = "*.TXT"
Case 2
   filFiles.Pattern = "*.DOC"
End Select
```

- اكتب الإجراء التالي في الحدث ( ) cmdOk_Click:

```
Dim PathAndName As String
Dim FileSize As String
Dim Path As String
If txtFileName.Text = "" Then
   MsgBox "يجب أن تختار ملفاً ",  vbMsgBoxRight, "برنامج الحجم "
   Exit Sub
End If
If Right(filFiles.Path, 1) <> "\" Then
  Path = filFiles.Path + "\"
Else
  Path = filFiles.Path
End If
If txtFileName.Text = filFiles.filename Then
  PathAndName = Path + filFiles.filename
Else
PathAndName = txtFileName.Text
End If
On Error GoTo FileLenError
'
FileSize = Str(FileLen(PathAndName))
MsgBox "حجم الملف  " + PathAndName + ": هو" _
 + FileSize + "  = بايت ",vbMsgBoxRight Or vbMsgBoxRtlReading,_
 "برنامج الحجم "
Exit Sub
FileLenError:
MsgBox "لم استطع إيجاد حجم الملف  " + PathAndName, _
  vbMsgBoxRight Or vbMsgBoxRtlReading, _
```

```
" برنامج الحجم "
Exit Sub
```

- أدخل الإجراء التالي في الحدث filFiles_DblClick ( ):

```
txtFileName.Text = filFiles.filename
cmdOk_Click
```

- أدخل النص التالي في الإجراء cmdCancel ( ):

```
End
```

- أحفظ المشروع من الأمر Save Project من القائمة File.

تنفيذ المشروع :

نفذ البرنامج وتمرن على شتى عناصر التحكم التي تظهر على الشاشة ,لاحظ المظاهر التالية عند تشغيل برنامج الحجم :

- تظهر أدلة محرك الأقراص الذي تنتقيه في مربع سرد الأدلة ,حاول اختيار محرك أقراص من مربع سرد محركات الأقراص.
- عند اختيار محرك أقراص غير جاهز تظهر رسالة خطأ ويسترجع مربع السرد قيمته الأصلية .
- يؤدي النقر المزدوج على الدليل المطلوب في مربع سرد الأدلة إلى اختياره .
- تظهر حال اختيار أحد الأدلة ملفات ذلك الدليل في مربع سرد الملفات ويظهر اسم الدليل الحالي فوق مربع سرد الأدلة .
- يظهر اسم الملف في مربع النص المسمى(اسم الملف)حال توضع الإضاءة فوقه في مربع سرد الملفات
- لاحظ أنك لا تستطيع الكتابة في منطقة النص لمربع السرد والتحرير وذلك بسبب إسناد القيمة DropDown إلى الخاصية Style لمربع التحرير والسرد أثناء طور التصميم
- تظهر عند نقر زر موافق رسالة تعطي حجم الملف المنتقى
- يمكنك كتابة اسم الملف في مربع النص اسم الملف بدلاً من انتقائه من مربع سرد الملف

كيف يعمل البرنامج :

نص الإجراء( ) Load_Form

ينفذ الإجراء ( ) Load_Form آلياً عند تشغيل البرنامج ويتم في هذا الإجراء تجهيز مربع التحرير والسرد CboFileType واللافته lbldirname.

وكما هو ملاحظ تستخدم الطريقة AddItem ثلاث مرات لملء مربع CboFileType بثلاث عناصر Allfiles(*.*),textfiles(*.txt),Docfiles(*.Doc)

إسناد القيمة صفر إلى الخاصية ListIndex لمربع نوع الملفات CboFileType يجعلها تشير إلى أول بند من بنوده وهو AllFile(*.*)

وأخيراً إسناد القيمة الابتدائية للخاصية Path لمربع سرد الأدلة إلى الخاصية Caption للافته lblDirName وهي الدليل الحالي وهكذا منذ بدء تشغيل البرنامج تُظهر اللافته lblDirName اسم الدليل الحالي .

نص الإجراء drvDrive_change():

يُنفذ الإجراء drvDrive_change() آلياً عند تغيير محرك الأقراص في مربع سرد محركات الأقراص بحيث يجدد هذا الإجراء الخاصية Path لمربع سرد الأدلة بإسناد محرك الأقراص الجديد الذي تم اختياره إليها.

قبل ان يغير الإجراء الخاصية Path لمربع سرد الأدلة كتبنا سطر مصيدة الأخطاء ,تعتبر مصيدة الأخطاء هذه لازمة لأن تغيير مسار مربع سرد الأدلة قد ينجم عنه خطأ ما,فمثلاً قد يغير المستخدم مربع سرد السواقات إلى محرك الأقراص المضغوطة ولا يوجد قرص فيها في هذه اللحظة يتسبب تغيير المسار بظهور خطأ لتلافي حصول خطأ من هذا النوع أثناء التنفيذ وضعت المصيدة الممثلة بالعبارة **On Error GoTo Drive Error**

فسيعمل فيجوال بيسك على نقل تنفيذ البرنامج إلى نص البرنامج تحت اللافتة Drive Error ليظهر نص البرنامج الوارد تحت هذه اللافته رسالة خطأ ويسترجع القيمة الأصلية لمحرك الأقراص وذلك باستخدام العبارة التالية :

```
drvDrive.Drive=dirDirectory.Path
```

لاحظ أن القيمة dirDirectory.Path لم تتغير (مازالت محتفظة بقيمتها الأصلية)لأن العبارة التي تسببت بالخطأ لم تنفذ بعد.أما في حال لم يحدث خطأ(اختيار السواقة صحيح)يتغير مسارPath مربع سرد الأدلة إلى محرك الأقراص المنتقى ويتم عرض الأول الموجود في محرك الأقراص الذي انتقيته في مربع سرد الأدلة.

نص الإجراء dirDirectory_change:

ينفذ الإجراء dirDirectory_change عند تبديل الدليل الحالي في مربع سرد الأدلة ,يُحدث نص الإجراء الخاصية Path لمربع سرد الملفات ويُسند الدليل الجديد إلى الخاصية Cption للافته lblDirName

```
FilFiles.Path=DirDirectory.Path
lblDirName.Caption=dirDirectory.Path
```

يُظهر مربع سرد الملفات والملفات المحتواة فب الدليل الذي اختاره المستخدم وذلك نتيجة لإسناد الخاصية Path لمربع سرد الأدلة إلى الخاصية Path لمربع سرد الملفات .

نص الإجراء CboFileType_Click():

ينفذ هذا الإجراء نتيجة اختيار نوع ملفات آخر من مربع نوع الملفات **CboFileType** وهذا الإجراء يُحدث الخاصية Path لمربع سرد الملفات تبعاً لنوع الملف المنتقى .

```
Select Case cboFileType.ListIndex
Case 0
  filFiles.Pattern = "*.*"
Case 1
  filFiles.Pattern = "*.TXT"
Case 2
```

```
   filFiles.Pattern = "*.DOC"
End Select
```

تستخدم العبارة الشرطية select case لتحديد نوع الملف الذي اخترته من مربع سرد أنواع الملفات CboFileType.

نص الإجراء FilFiles_Click():

ينفذ الإجراء FilFiles_Click() عند اختيار ملف ما من الملفات الموجودة في مربع سرد الملفات ويُحدث نص هذا الإجراء مربع النص TXTFileName باسم الملف المنتقى

```
txtFileName.Text = filFiles.filename
```

نص الإجراء CmdOk_Click():

ينفذ الإجراء CmdOk_Click عند النقر على الزر موافق ويظهر نص هذا الإجراء حجم الملف المنتقى من مربع سرد الملفات

```
Dim PathAndName As String
Dim FileSize As String
Dim Path As String
عند عدم اختيار ملف أخبر المستخدم بذلك وأن الإجراء'
If txtFileName.Text = "" Then
   MsgBox "يجب أن تختار ملفاً" _
   vbMsgBoxRight Or vbMsgBoxRtlReading, _
  " برنامج الحجم "
   Exit Sub
End If
```

(تحقق من أن المسار ينتهي بـ \ )

```
If Right(filFiles.Path, 1) <> "\" Then
  Path = filFiles.Path + "\"
Else
  Path = filFiles.Path
End If
If txtFileName.Text = filFiles.filename Then
  PathAndName = Path + filFiles.filename
Else
  PathAndName = txtFileName.Text
End If
On Error GoTo FileLenError
FileSize = Str(FileLen(PathAndName))
```

```
MsgBox " حجم الملف  " + PathAndName + ": هو " _
 + FileSize + "  بايت ",vbMsgBoxRight Or vbMsgBoxRtlReading,_
 " برنامج الحجم "
Exit Sub
FileLenError:
MsgBox " لم استطع ايجاد حجم الملف" + PathAndName, _
  vbMsgBoxRight Or vbMsgBoxRtlReading, " برنامج الحجم "
Exit Sub
```

أول شيء يفعله الإجراء هو التأكد من أنك اخترت ملفاً ما,وذلك بمقارنة الخاصية Text لمربع النص txtFileName مع رمز الفراغ " " فإذا تحقق الشرط تظهر رسالة مفادها أنك لم تختر ملفاً ويتم إنهاء الإجراء .

أما إذا تأكد الإجراء من أن المستخدم انتقى ملفاً فيتم عند ذلك تحديث المتحول Path باسناد مسار الملف المنتقى إليه.

يستخدم التابع الوظيفي Right() للتأكد بأن الرمز الواقع أقصى يمين المسار المنتقى هو الرمز"\" فإذا لم يكن كذلك فإنه يضيفه إلى المتحول Path

وبعد أن يصبح المتحول Path جاهزاً يمكن تحديث قيمة المتحول PathAndName بوتسطة عبارة If شرطية :

```
If txtFileName.Text = filFiles.filename Then
  PathAndName = Path + filFiles.filename
Else
  PathAndName = txtFileName.Text
End If
```

تتحقق هذه العبارة من تطابق الاسم المضاء في مربع سرد الملفات مع الاسم الموجود في مربع النص txtFileName فإذا كان هناك اختلاف فهذا يعني أنك كتبت يدوياً مسار واسم الملف لهذا يتم إسناد الشيء الذي كتبته إلى التحول PathAndName

```
Path+FilFile.FileName
```

وبعد تجهيز المتحول PathAndName يصبح بالإمكان استخدام التابع الوظيفي()FileLen لإيجاد حجم الملف,وباعتبار أن استخدام التابع الوظيفي ()FileLen قد يؤدي إلى حدوث خطأ أثناء زمن التنفيذ (كأن يدخل المستخدم اسم ملف غير موجود) فقد وضعت مصيدة الخطأ FileLenError فإذا وقع الخطأ تظهر رسالة بذلك وإلا يتم إظهار حجم الملف على الشاشة على شكل رسالة.

مثال ألبوم صور :

المطلوب عند تغيير محرك الأقراص تظهر المجلدات المحفوظة علية وعند دخول المجلد تظهر كافة الملفات التي بداخله وعند الضغط على أي ملف صورة يقوم البرنامج بعرض الصورة في أداة عرض الصور Image1 .

لعمل هذا البرنامج نحن بحاجة إلى الأدوات التالية :

1- Drive list Box

2- Dir list Box

3- File List Box

4- Image

في الحدث Clickللأداة Fileنكتب الكود التالي :

```
Dim k as string
K=file1.path & "\" & File1.FileName
Image1.picture=loadpicture(K)
```

في الحدث Change للأداة Drive نكتب الكود التالي :

```
Dir1.path=Drive1.Drive
```

وفي الحدث Change للأداة Dir1نكتب الكود التالي :

```
File1.path=Dir1.Path
```

تحرير الملفات

يوفر الفيجوال بيسك مجموعة من الأوامر التي تمكننا من تحرير الملفات لحفظ بيانات برامجك فيها بالتنسيق والهيئة التي تريدها وقبل إجراء أي عملية تحرير على الملف لابد من فتحه باستخدام العبارة Open والتي صيغتها العامة على الشكل

Open اسم الملف For نوع الوصول As # رقم الملف

بالنسبة لرقم الملف هو رقم يمثل الملف بحيث يمكنك الوصول إليه من كافة أنحاء البرنامج ولا يمكن لهذا الرقم أن يتكرر لأكثر من ملف لذلك حتى تتفادى أخطاء التعارض يفضل استخدام الدالة FreeFile التي تعود برقم غير محجوز لفتح الملف .

أما أنواع الوصول فهي الطريقة التي تود أن تتعامل مع الملف بها وهي ثلاثة أنواع :

1- للقراءة من الملف نستخدم الطريقة Input

2- للكتابة إلى الملف نستخدم الطريقة Output

3- للإضافة إلى الملف نستخدم الطريقة Append

بإمكانك قراءة سطور من الملفات المفتوحة بالكلمة المحجوزة Input باستخدام العبارة LineInput حتى نهاية الملف والذي تختبره عن طريق الدالة EOF

مثال :

```
X=FreeFile
Dim A As string
Open APP.Path & "\myfile.txt " For Input As # X
While No EOF (X)
Input # X,A
Text1.text=text1.text+VBCRLF+a
```

```
Loop
Close # X
```

مثال :

```
Do While not EOF (x)
Input # X , A
Text1.text=text1.text & vbNewline & A
Loop
Close # X
```

الإضافة إلى الملف :

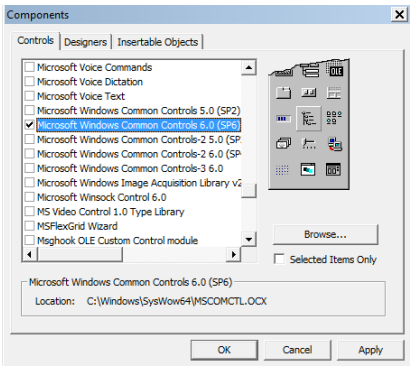
```
Open "D:\Myfile.txt" For Append As # x
Print # x ,text1.text
Close # x
```

الكتابة إلى الملف :

```
X=freeFile
Open App.Path & "\myFile.txt: For output As # x
Print #x,text1.text
Close # X
```

المحاضرة السابعة **إنشاء شريط أدوات ToolBar**

شريط الأدوات من الأدوات الخارجية لذلك عليك إحضاره بالنقر على صندوق الأدوات ToolBox باليمين واختيار الأمر Components من القائمة التي تظهر مما يؤدي إلى ظهور مربع حوار Components



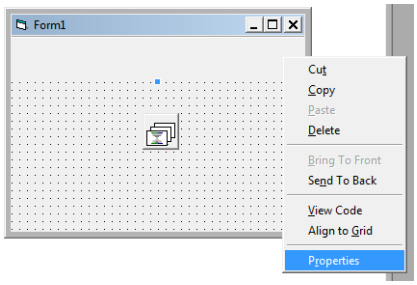
من مربع الحوار هذا نختار البند Microsoft Windows Common Control6.0المسئول عن إظهار مجموعة من الأدوات منها Image list بالإضافة إلى Tool Bar

نختار الأداة Image List ونضعها على النموذج ,هذه الأداة مسئولة عن الصور التي ستظهر على شريط الأدوات الذي نريد أن نصنعه.

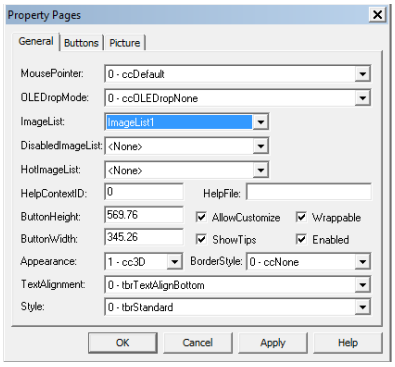
بعد وضعها على النموذج ننقر عليها باليمين ونختار الأمر خصائصProperties فيظهر لدينا مربع حوار خصائص Image list .

من علامة التبويب General عام نحدد مقاس الصورة أو الأيقونات التي ستوضع على Tool Bar وثم نضغط علة علامة التبويب image ونختار منها Insert picture فيظهر لدينا مربع حوار يشبه مربع حوار فتح يمكننا من وضع الأيقونات على الشريط.

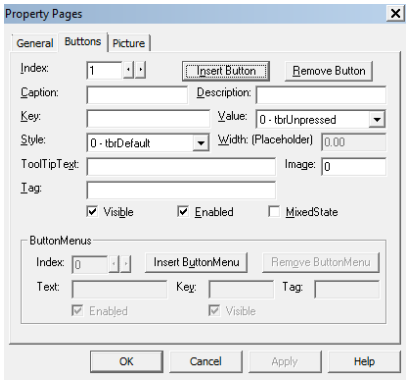
الآن سنحضر الأداة Tool Bar لذلك ننقر عليه مرتين فيظهر في القسم العلوي من النموذج ,ننقر عليه باليمين ونختار خصائص properties .



يظهر مربع حوار جديد خاص بشريط الأدوات من علامة التبويب General نحدد Image list التي تحتوي الأيقونات والصور وهي تكون بالاسمimagelist1 في العادة .



نضغط الآن على علامة التبويب Buttons ونختار منها الزر Insert Button



الآن ليكون لكل زر أيقونته الخاصة نختار رقم الزر من خانة Index وذلك بالضغط على الأسهم الصغيرة ثم في خانة image اكتب رقم الأيقونة التي وضعتها في Image list .

لوضع قائمة لأحد الأزرار على شكل قائمة منسدلة نحدد البند style من مربع الحوار السابق ونختار من قائمته الخيار 5-tbrDropdown ثم من نفس مربع الحوار ننقر على الزر Insert Button Menu سنحدد عدد الاختيارات في القائمة المنسدلة فمثلاً نريد عند الضغط على السهم الصغير تنسدل لنا قائمة فيها فتح – ضغط – حفظ – حفظ باسم – خروج ولعمل ذلك نضغط على زر insert Button Menu سنجد أن رقم index أصبح 1 في الخانة Text نقوم بكتابة أول اختيار وهو فتح ونكرر العملية مع كل اختيار .

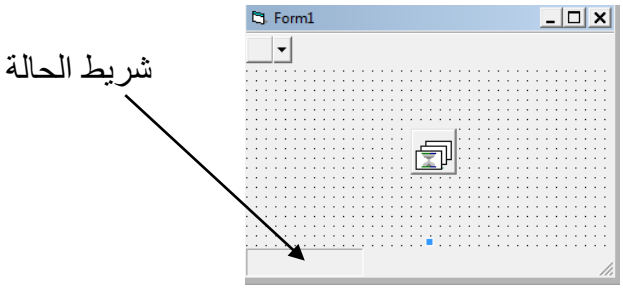
الآن مرحلة كتابة الكود لذلك ننقر مرتين على شريط الأدوات لفتح نافذة الكود ونستخدم جملة Select Case على الشكل التالي :

```
Select case Button.index
    Case 1:نضع هنا الكود الواجب تنفيذه
    Case 2:نضع هنا الكود الواجب تنفيذه
    Case 3:نضع هنا الكود الواجب تنفيذه
    ……
End select
```

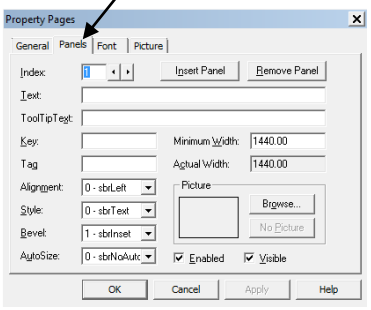
أما الكود الخاص بأوامر القائمة المنسدلة الخاصة بأحد الأزرار تأكد من الحدث Buttonmenu_Click() وبعد ذلك استخدم الجملة Select Case

شريط الحالة StatusBar

من صندوق الأدوات نختار الأداة StatusBar بالنقر عليه مرتين فيتوضع على القسم الأسفل للنموذج ثم ننقر عليه باليمين ونختار البند خصائص .

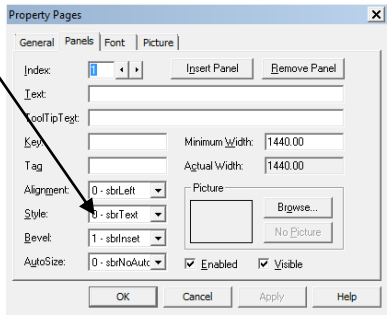


يظهر مربع حوار نختار منه علامة التبويب Panels والتي تحدد عدد خانات هذا الشريط وذلك من خلال الزر Insert panel .



من خانة Index نختار الخانة الأولى (1)وذلك لوضع شيء في هذه الخانة وفي صندوق النص Text نكتب أي شيء يعبر عن محتوى هذه الخانة مثلاًWWW.teckno.Com

في الخانة 2 سنضع التوقيت الحالي للجهاز فنقوم باختيار الصندوق Style ونختار SbrTime وهكذا .....



يمكن وضع picture لوضع صورة على شريط الحالة بحيث نستطيع اختيارها بالضغط على زر Browse

المحاضرة الثامنة اكتشاف الأخطاء

إن كتابة برنامج دون أخطاء شيء يتحقق في الخيال فقط ولكن كلما زادت احتياطاتك لتفادي الأخطاء قلت نسبة ظهورها .تصنف الأخطاء في أي لغة برمجية إلى صنفين على أساس وقت حدوثها :

- إما في وقت التصميم
- أو في وقت التنفيذ

هذه الأخطاء تسبب انهيار برنامجك وإنهاء تنفيذه بالإضافة إلى ذلك يوجد نوع من الأخطاء التي لا تظهر بشكل مباشر تعرف بالشوائب Bugs .

أخطاء وقت التصميم Design Time Error:

وتعرف أيضاً بالأخطاء النحوية Syntax Error وهي أسهل أنواع الأخطاء اكتشافاً وإصلاحاً.وقت حدوث هذه الأخطاء يكون في مرحلة التصميم أو الترجمة للبرنامج, سببها الرئيسي في طريقة كتابة العبارات البرمجية الخاطئة فمثلاً قد تكتب اسم دالة غير موجودة أو تنشئ حلقة For دون إقفالها بعبارة Next .

توفر بيئة التطوير المتكاملة فيجوال بيسك تقنية في قمة الروعة هدفها قنص الأخطاء تلقائياً بمجرد الوقوع فيها وذلك بعد الضغط على المفتاح Enter فمثلاً قم بكتابة X==4 واضغط Enter مباشرة ستظهر رسالة خطأ توضح الخطأ وقد قلب لون السطر إلى اللون الأحمر تعرف هذه التقنية بالتدقيق النحوي التلقائي.

أخطاء وقت التنفيذ Run Time Error :

وقت ظهور هذه الأخطاء مختلف فلن تظهر الرسالة المزعجة السابقة وقت كتابة الكود وإنما وقت التنفيذ عندما يصل المفسر عند سطر صحيح نحوياً لكنه خاطئ منطقياً ستظهر رسالة خطأ بعنوان Run Time Error ويظهر تحديد لمكان السطر الذي وقع الخطأ فيه مثلاً :

```
Dim A As Byte
A=256
```

من الواضح ان الصيغة النحوية لهذا الكود صحيحة ومن الناحية المنطقية خطأ جرب تنفيذ البرنامج وستلاحظ ظهور رسالة خطأ over Flow وذلك لأن القيمة القصوى التي يمكن أن يحملها أي متغير من نوع Byte هي 255 طبعاً أخطاء وقت التنفيذ كثيرة جداً فأنت عندما تصمم برنامجاً تتوقع أن كل الاحتمالات الخارجية كما هي في حالة تصميم البرنامج مثلاً لو وجد في أحد السطور أمر يقوم بمسح ملف معين وكتبت هذا الكود Kill "FileName.txt" افترض أن الملف لم يكن موجوداً فستظهر رسالة الخطأ فوراً لذلك يجب أن تتأكد من وجود الملف باستخدام دالة Dir ومن ثم حذفه

```
If Dir$("FileName.txt") then kill "FileName.txt"
```

ولكن ماذا لو كان الملف موجود ولكن خاصية readonly مدعومة به أي أنه غير قابل للحذف عندها ستظهر رسالة الخطأ من جديد:

```
IF Dir$("FileName.txt") Then
  IF Not (GetAttr("FileName.txt") And VbReadOnly) Then
```

```
Kill "FileName.txt"
End IF
End IF
```

الشوائب Bugs:

قد يكون الكود سليم من الناحية النحوية ولا توجد فيه أي أخطاء في وقت التنفيذ ولكن فيه شوائب لا يوجد برنامج إلا و فيه شوائب فما هي الشوائب ؟؟؟

هي أخطاء في سلوك تنفيذ البرنامج ولكنها لا تسبب في إيقافه ووهي صعبة الإيجاد والاكتشاف لذلك تجد أغلب البرامج التجارية الكبيرة تصدر نسخة تجريبية Beta توزع على أشخاص وشركات معينة الهدف منها تجربة البرنامج والتحقق من اكتشاف الشوائب الموجودة فيه .

من اكبر الأخطاء التي يقع فيها المبرمج هي محاولة اكتشاف الشوائب بنفسه لأنك لن تستطيع اكتشاف الشوائب إلا عن طريق غيرك(صاحب البرنامج) وفيجوال بيسك نفسها فيها الكثير من الشوائب التي تكتشف دورياً وتصدر شركة مايكروسوفت تقارير عنها تجدها في مكتبة MSDN بعضها تم إصلاحه والبعض الآخر لا .

الكائن Err:

بدلاً من كتابة عشرات الأسطر للتأكد من قابلية حذف الملف استخدم كائن الخطأ Err وقبل تطبيق هذا الكائن عليك معرفة أن كل خطأ من أخطاء وقت التنفيذ له رقم خاص يميزه عن غيره من الأخطاء وكذلك وصف نصي مختصر للخطأ وعند حدوث الخطأ يتم وضع هذه البيانات الخاصة بالخطأ في الكائن Err .عند رغبتك في الاستمرار في عملية تنفيذ البرنامج ,حتى عند حدوث الخطأ لا بد من كتابة التعليمة On Error Resume Next عند بداية كل إجراء تتوقع حدوث خطا فيه حتى يستمر في تنفيذ سطور البرنامج كمثال ذلك :

```
On Error Resume Next
Kill "FileName.txt"
IF Err Then
MsgBox Err . Description
Err . clear
End IF
```

هنا سنقوم بمحاولة حذف الملف ,إن لم يستطع البرنامج فعل ذلك فإن الكائن Err سيحتوي على خصائص تتعلق بذلك الخطأ وستظهر رسالة توضح وصف الخطأ ومن المهم التأكد من تنظيف الكائن Err حتى نخبر البرنامج أننا انتهينا من البحث عن اخطأ وأنه لا توجد أخطاء أخرى .

أما إذا كانت أكواد البرنامج طويلة ولا تود كتابة الجملة الشرطية IF Err Then مرات متعددة فيفضل استخدام الجملةON Error GoTo X والتي تؤدي إلى الانتقال إلى سطر معين في حال حدوث الخطأ مثال ذلك :

```
On Error GoTO XX
.....
.....
```

```
XX:
Msgbox Err.Description
Err.Clear
```

المحاضرة التاسعة الإجراءات

الإجراءات :هي مجموعة من التعليمات يتم تنفيذها دفعة واحدة عند استدعاء الإجراء ثم يعود البرنامج إلى تنفيذ بقية التعليمات .

الفكرة الرئيسية وراء استخدام الإجراءات بكفاءة تكمن في تقسيم البرنامج إلى مهام صغيرة يمكن احتوائها على انفراد في إجراءات أو دوال أو كائنات ويؤدي ذلك إلى سهولة اختبار كل إجراء على حده وعدم تكرار الكود بلا داع .

إنشاء الإجراءات الفرعية واستخدامها :

لإنشاء إجراء مباشرة

- ضع مؤشر الإدخال في نافذة الكود في قسم الإعلان العام .
- اكتب Sub واتبعها بمسافة
- اكتب اسم الإجراء مثلا Test
- اضغط Enter لإنشاء الإجراء

بمجرد الضغط على مفتاحEnter يقوم فيجوال بيسك بالآتي :

- وضع أقواس المعاملات (Arguments) بعد اسم الإجراء مباشرة
- إضافة عبارة End Sub في السطر التالي
- كتابة اسم الإجراء الجديد يظهر في مربع الأحداث

ولاستدعاء هذا الإجراء نكتب اسمه مباشرة في الكود .

تمرير البيانات من وإلى الإجراء :

هناك طريقتان لتبادل البيانات مع الإجراء :

- استخدام المتغيرات العامة Public والتي يمكن استدعاؤها من أي مكان في الكود ومن ثم يمكن قراءتها وتغييرها من خلال الإجراء .

```
Public Name1 As String
Sub Test()
Name1="Mustafa"
End Sub
```

- استخدام المعاملات parameters:هذه المعاملات يمكن تمريرها من وإلى الإجراء دون الحاجة لمتغيرات عامه.

```
Sub Test(Name1 As String)
Name1="Mustafa"
End Sub
```

إنهاء الإجراء :

لسبب من الأسباب قد تحتاج إلى إنهاء الإجراء دون إكمال تنفيذ بقية أوامره يتم ذلك باستخدام العبارة Exit Sub

```
Sub Test(Name1 As String)
If Name1="" Then Exit Sub
```

End IF

استخدام الدوال (التوابع الوظيفية)Function:

ما هي الدوال ؟:

الدوال على نوعين :

1- دوال جاهزة من الفيجوال بيسك

2- دوال معرفة من قبل المبرمج (UDF)

الدوال الجاهزة هي أسماء محجوزة ومعرفة من قبل الفيجوال بيسك لتقوم بعمل معين مثل المصفوفات والقيم المطلقة وغيرها ...وكأمثلة :

- الدوال الرياضية
- دوال السلاسل
- دوال الوقت والتاريخ
- دوال التحقق من أنواع البيانات
- دوال المدخلات والمخرجات
- دوال مالية

إنشاء الدوال :

تشبه عملية إنشاء الدوال لإنشاء الإجراءات تماماً ولكن الدالة ترجع قيمة عند ندائها حيث يمكن تخزينها في متغير أو استخدامها في تعبير مباشرة.

```
Public Function Sum(X As integer,Y As integer)
IF X=0 Or Y=0 then
Exit Function
End IF
Sum=X+Y
End Function
```

تحديد مجال الإجراءات والدوال :

الإجراءات والدوال كالمتغيرات تماماً لها مجال رؤية Scope وهي الأماكن التي يمكن نداء الإجراء أو الدالة منه وتنقسم إلى نوعين :

1- إجراءات عامة Public

2- إجراءات خاصةPrivate

الإجراءات العامة هي التي يراها البرنامج في أي مكان منه وتستخدم لها الكلمة Public بينما الخاصة ينحصر مجال الرؤية فقط ضمن الإجراء المصرح عنها ضمنه

أمثلة عن الدوال الجاهزة :

الدالة Rnd:هي دالة لإيجاد رقم عشوائي عشري ولكي لا يتكرر الرقم استخدم قبلها الدالة Randomize.

الدالة Int:تقوم بجعل العدد العشري عدد صحيح مثال ذلك :

```
Label1.Caption=Int(1.958)
```

النتيجة هي ظهور العدد (1)

الدالةSTR:تقوم بتحويل الرقم إلى نص

الدالة Round:تقوم بتقريب العدد للمنزلة المطلوبة مثال ذلك

```
Label1.Caption=Round(2.567,1)
```

النتيجة 2.6

الدالةLen:تستخدم لمعرفة عدد أحرف جملة معينة أو حتى أرقام مثل :

```
Label2.Caption=Len("حسن")
```

النتيجة هي (3)

الدالة Trim:إذا كانت لديك جملة نصية أو رقمية واحتمال كتب المستخدم في بدايتها مسافة أو حتى نهايتها فيكون باستخدام هذه الجملة حذف المسافات

```
Label1.Caption=Trim(Text1.Text)
```

حالات خاصة :

- الحذف من اليسار

```
Label1.Caption=LTrim(Text1.Text)
```

- الحذف من اليمين

```
Label1.Caption=RTrim(Text1.Text)
```

الدالة Mod: تقوم بعرض باقي القسمة لعددين

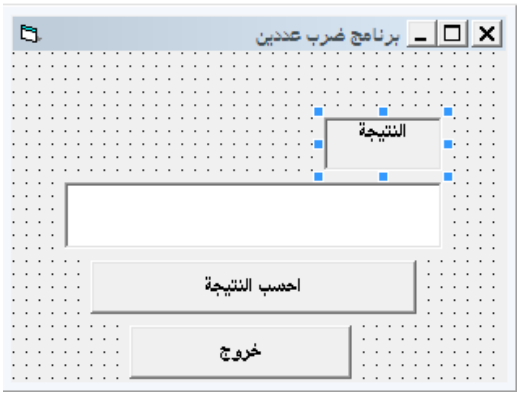
```
Label1.Caption=4 Mod 3
```

النتيجة (1)

مثال :برنامج ضرب عددين :

القيمة	الخاصية	الأداة
frmMultiply	Name	Form
برنامج ضرب عددين	Caption	
True	R TO L	
CmdExit	Name	CommandButton
&خروج	Caption	
CmdCalculate	Name	CommandButton
ا&حسب النتيجة	Caption	
txtResult	Name	Textbox

فارغ	Caption	
lblResult	Name	Label
النتيجة	Caption	



في حدث click() للزر احسب النتيجة اكتب Multiply 2,3

هنا تم استدعاء الإجراء المسمى Multiply وتطبيقه على العددين 2 و 3

السؤال الآن كيف يمكن إضافة الإجراء :

- افتح القائمة Tools واختر الأمرAddprocedure
- بعد ظهور مربع الحوار الخاص بالأمر أدخل الاسم Multiply في مربع النص Name
- ضع علامة بجانب Sub ضمن الحقل Type
- ضع علامة بجانب Public ضمن الحقل Scopeحتى يصبح الإجراء مرئياً لجميع الإجراءات في المشروع.
- يستجيب فيجوال بيسك بإضافة الإجراء Multiply إلى المنطقة العامة من النموذج فيظهر نص الإجراء multiply بالشكل التالي :

```
Public Sub Multiply()
End sub
```

سنحتاج إلى تعديل السطر الأول من الإجراء ليصبح كالتالي :

```
Public Sub Multiply(X as Integer,Y As Integer)
End sub
```

وذلك لأننا سنمرر للإجراء وسيطين يمثلان العددين المطلوب معرفة ناتج ضربهما

```
Dim Z As Integer
Z=X*Y
txtResult.text=Str(Z)
```

يحتاج الإجراء لمعرفة العددين اللذين سيحسب حاصل ضربهما ولهذا قدمنا له العددين 3 و 2 كوسيطين له .

التابع (Function)

اختر AddProcedure من القائمة Tools

اختر الاسم Multiply ضمن مربع النص Name

اختر Function ضمن الحقل Scope

انقر OK

يستجيب الفيجوال ويضيف تابع على الشكل التالي :

```
Public Function multiply()
End Function
```

غير السطر الأول لهذا التابع

```
Public Function multiply(X As Integer,Y As Integer)
End Function
```

الآن التابع الوظيفي Multiply له وسيطين.أضف ضمن التابع العبارات التالية :

```
Dim Z As Integer
Z=X*Y
Multiply  =Z
```

غير النص للإجراء CmdCalculate_Click():

```
txtResult.Text=Str(Multiply(2,3))
End  Sub
```

يستدعي نص الإجراء القيمة المعادة من التابع الوظيفي Multiply إلى الخاصية Text لمربع النص txtResult.

ملاحظـــة:

الطرق هي إما تابع وظيفي function أو إجراء Procedure.

مجموعة محاضرات قواعد البيانات ؟

- ما هي قواعد البيانات ..
- المصطلحات جدول ، سجل و حقل .. ما هو معناها ؟؟
- ما هي نظم ادارة البيانات DBMS و ما هي مكوناتها ...
- ما هو الـ Microsoft Database Jet ، و ما علاقته بالـ VB !!!
- انشاء قواعد البيانات بـ Visual Data Manager
- تقنيات الوصول الى قواعد البيانات
  - Data Control
  - ActiveX Data Objects (ADO)
- إنشاء التقارير باستخدام Data Report

من أهم الأشياء في قواعد البيانات هو تصميم قاعدة البيانات بعد معرفة نظام قواعد البيانات الذي تريد استخدامه في إنشاء قاعدة البيانات ,هذه العملية تحددها حجم المؤسسة التي طلبت منك البرنامج فالمؤسسات الكبيرة تتطلب قاعدة بيانات من نوع أوراكلOracle بينما المؤسسات المتوسطة والصغيرة فهي بحاجة إلى قاعدة بيانات من نوعSQL Server بينما إذا كتن البرنامج لمحل تجاري فيكفي برنامج أكسس ACCESS لتصميم قاعدة بيانات له .بعد تحديد نوع قاعدة البيانات نحدد طريقة الربط مع هذه القاعدة حيث يوجد العديد من الطرق للربط مع قاعدة البيانات منها :ADO – DAO(Active X Data Object) – RDO ....ولكن الأفضل والأشهر والأقوى في تقنية الربط هو ADO

قواعد البيانات هي ملف ينشأ بأحد برامج إدارة قواعد البيانات (أوراكل- أكسس – SQL SERVER)والهدف من ذلك تجميع البيانات والمعلومات بصورة منظمة وبترتيب معين ضمن جداول مترابطة حتى يتم استخدام محرك البحث لنظام إدارة قواعد البيانات للبحث عن المطلوب بسرعه وسهولة مع إمكانية الإضافة والتعديل على قاعدة البيانات .

الجدول :Table

هو عبارة عن مجموعة من الأعمدة والأسطر متقاطعة فيما بينها بحيث نسمي العمود حقل Field بينما نسمي السطر سجل Record.

الحقل Field:

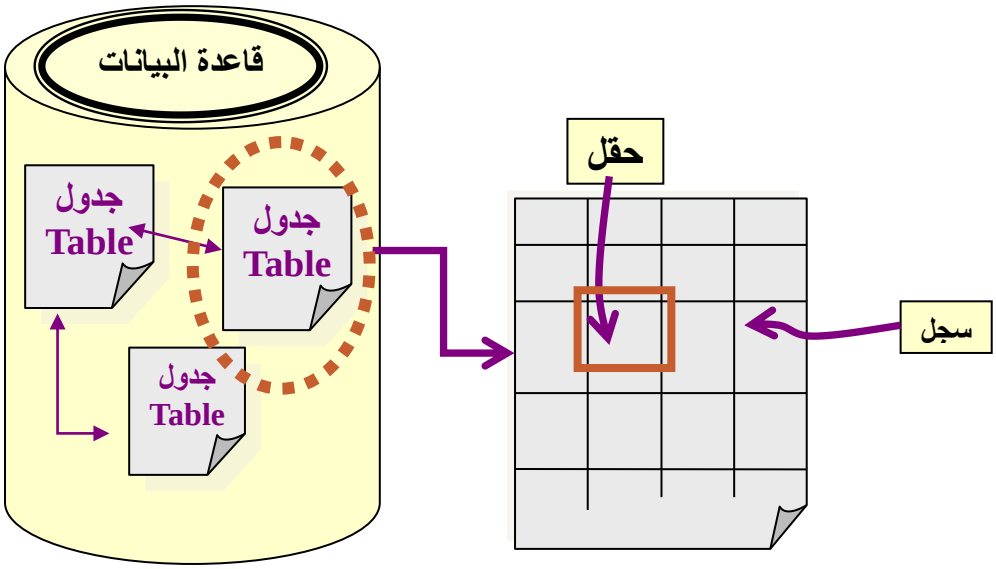
هو عمود في جدول يستخدم لتخزين نوع واحد من البيانات (رقمية,نصية,منطقية....)

السجل Record:

هو سطر في جدول يحتوي على بيانات من مجموعة من الحقول

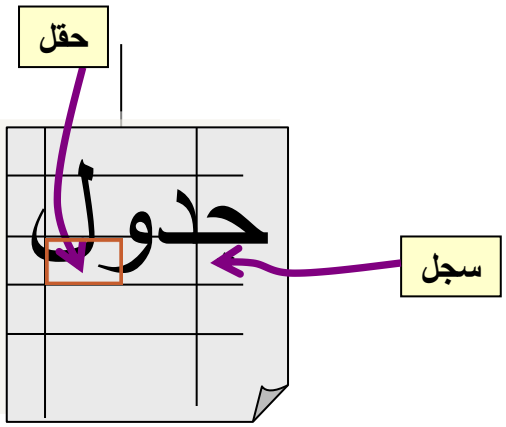
مثال على ذلك : اسماء الطلاب و درجاتهم الدراسية و بيانات المواد و المدرسين ، الخ

الشكل التالي يوضح شكل قاعدة البيانات و مكوناتها :



- **نظام إدارة قاعدة البيانات DBMS :**

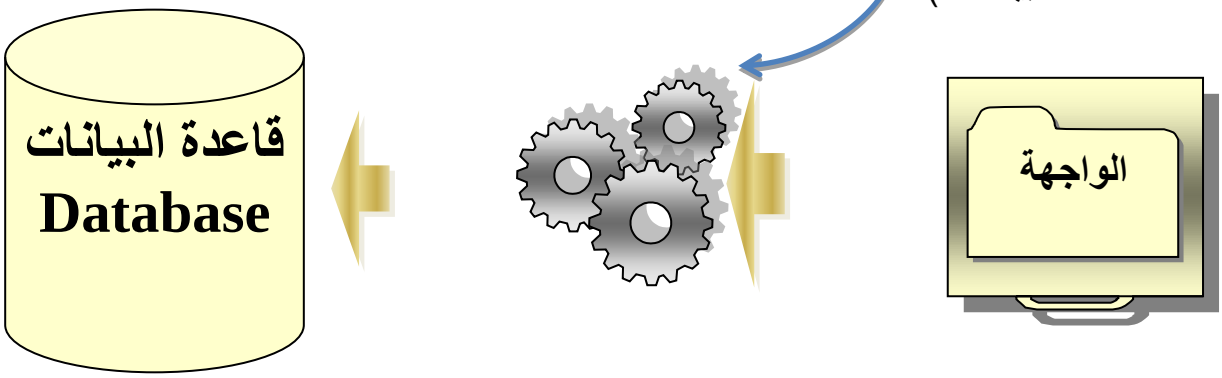
هو نظام يدير كافة العمليات التي تجرى على قاعدة البيانات مثل عملية إنشاء الجداول أو التعديل عليها أو حتى حذفها من قاعدة البيانات ، و أيضا عملية بناء العلاقات بين جداول قاعدة البيانات .مثال على أنظمة إدارة قواعد البيانات DBMS : Access و Oracle و MS-SQL Server و FoxPro و غيرهم الكثير ..



مكونات نظم ادارة قواعد البيانات:

يتكون نظام ادارة قاعدة البيانات من جزئيين هما :

1. ***الواجهة User Interface*** التي يتعامل من خلالها مع المستخدم ..
2. ***محرك قاعدة البيانات Database Engine*** و هو عبارة عن الآلية التي من شأنها تنظيم التعامل مع قاعدة البيانات (من إضافة و حذف و تعديل ... الخ) و ذلك في سبيل ضمان تكامل بيانات قاعدة البيانات و سريتها (ما معنى تكامل Integrity البيانات ؟) .



- Microsoft Database Jet هو أحد الامثلة على محركات قاعدة البيانات و الذي يوفر الوسائل الأولية للفيجوال بيسك لكي يتعامل مع قواعد البيانات المختلفة (Access , Oracle .. الخ)
- لم يكن Microsoft Database Jet جزءاً من VB كما هو الآن (و ذلك حتى الاصدارة الثالثة من VB) مما استدعى استخدام برنامج الاتصال المفتوح لقواعد البيانات ODBC داخل بيئة الفيجوال بيسك و الذي يعتبر احد البرامج التي يقدمها ويندوز ، و يحتوى هذا البرنامج على مشغلات Drivers ، حيث ان كل مشغل يخص نوع معين من قواعد البيانات ، مثال على ذلك Microsoft Access Driver الذي يتح التعامل مع قواعد البيانات المبنية بواسطة برنامج Access المعروف .

خطوات تصميم قاعدة بيانات سليمة :

1- تصميم خريطة للنظام

2- تحديد أنواع البيانات المطلوبة للنظام

3- تنظيم البيانات داخل الجداول

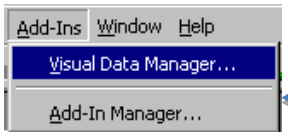
4- إنشاء العلاقات بين الجداول

إنشاء قواعد البيانات بـ Visual Data Manager:

كما هو ممكن إنشاء قاعدة البيانات داخل برنامج الـ Access نفسه

فإنه من الممكن أيضا إنشاء أي نوع من انواع قواعد البيانات داخل بيئة

خاصة داخل VB تسمى Visual Data Manager، و ذلك كما يلي:

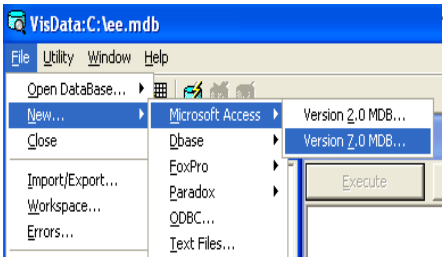


الخطوة الأولى فتح الفيجوال بيسك كالمعتاد و من ثم

الذهاب إلى قائمة Add-ins >> visual data manager...

ستفتح لك نافذة البرنامج اذهب

إلى File>>new>>Microsoft access>>version 7.0 mdb

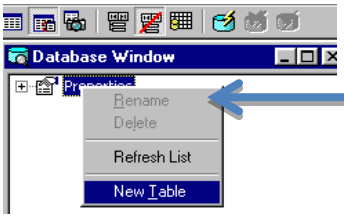


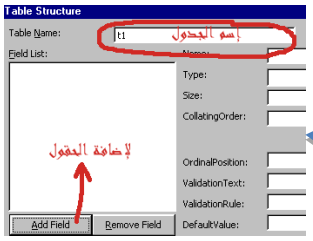
سيظهر لك مربع حفظ لتحدد الموقع الذي

تريد أن تحفظ فيه قاعدة بياناتك, بعدها اختر حفظ .

بعد ذلك ستظهر لك قاعدة البيانات ، حدد

الخصائصProperties بالزر الأيمن



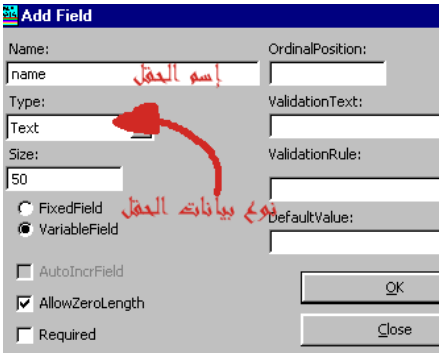


ثم أختر New Table من اجل انشاء جدول .

بعد ذلك ستفتح لك نافذة تكتب فيها أسم الجدول

و ثم أضف حقول إلى الجدول بالضغط على زر

.AddField



يتفتح لك نافذة إضافة الحقول ، اكتب إسم الحقل

في المكان المخصص و حدد نوع بيانات الحقل

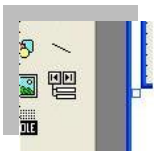
أي هل هي رقمية أو حرفية ... الخ ، ثم كـــرر

العملية حتى تنهي جميع الحقول التي تريدها .

تقنيات الوصول الى قواعد البيانات:

- **التقنية *Data Access Objects*) DAO) :** توفر هذه التقنية مجموعة من الكائنات (Objects) للتخاطب مع قاعدة البيانات، وإرسال الأوامر لها. وهي مصممة بشكل رئيس للتعامل مع قواعد بيانات أكسس (الى الاصدارة 97) ، و يمكنك استخدامها في التعامل مع قواعد بيانات أخرى من خلال المحرك Jet الذي توفره.

- **أداة التحكم في البيانات Data Control :** و هي تقنية سهلة للتعامل مع قاعدة البيانات و هي احدي أدوات التحكم الموجودة في صندوق الأدوات القياسي



و هي تمكننا من الاتصال بمجموعة من السجلات Recordset

في قاعدة بيانات Jet للتعامل معها و كذلك ربطها بعدة ادوات في

فيجوال بيسك مثل أداة Textbox..

- **تقنية Remote Data Objects (RDO) :** و طوّرت هذه التقنية بشكل رئيس لتوفر طبقة مُبَسِطة لتقنية ODBC لمبرمجي لغة فيجوال بيسك.

- ***تقنية ActiveX Data Objects (ADO)*** : واستكمالاً لمسيرة التقدم والتطوير طرحت شركة مايكروسوفت بعدها نظام ADO، والتي قامت بتسهيل عملية التعامل مع قواعد البيانات بشكل كبير، وأصبحت الشكل النهائي للوصول لقاعدة البيانات ليس في لغة VB و حسب ، بل و في العديد من لغات البرمجة .

الربط بالأداة Data Control :

- **قبل عملية الربط يجب علينا أولا تصميم قاعدة البيانات ، و يجب أن تكون قاعدة البيانات من نوع Access 97 على أقصى تقدير .**

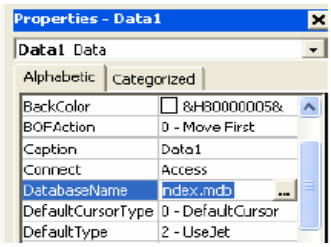
- **خطوات الربط ستكون على النحو الآتي :**

1. يتم إنزال الأداة Data Control من صندوق الأدوات إلى الواجهة Form ، و سيعطى لها الاسم Data1.

2. من خلال خصائص الأداة Data Control

يتم تحديد قاعدة البيانات المراد الاتصال بها

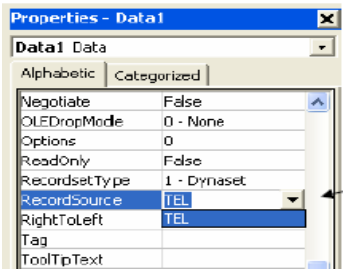
عن طريق الخاصية DatabaseName .



3. بعد ذلك يتم تحديد الجدول المراد الاتصال

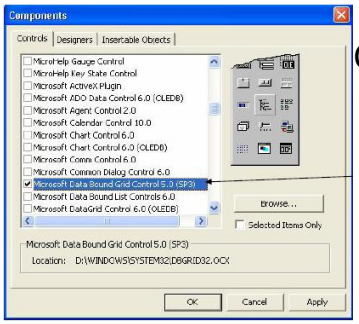
به عن طريق الخاصية *RecordSourse*

و التي معناها السجلات التي أود استرجاعها



عرض البيانات بـ Data Grid:

- من الممكن كذلك عرض البيانات داخل أداة خاصة تعرف باسم Data Bound Grid و التي تتميز بعرض بيانات جدول ما في قاعدة البيانات عل شكل جدول و ذلك بحسب الخطوات التالية :



1- من القائمة Project نختار الأمر Component

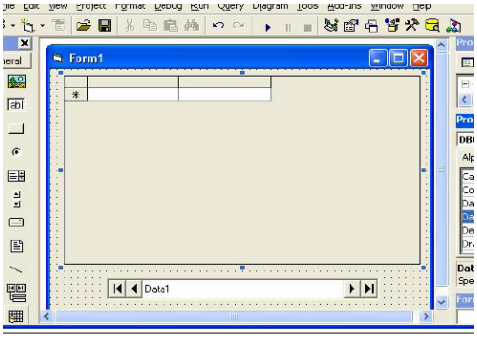
و الذي سيظهر مجموعة الأدوات الإضافية التي

يوفرها VB ، و نختر منها الأداة **Data Bound Grid**

و بعدها نضغط موافق . الجديدة

2- من مربع الأدوات نختر الأداة الجديدة

3- تظهر لنا الأداة على الواجهة Form كما في الشكل التالي .



4- الآن من خصائص الأداة Data Grid نحدد في الخاصية DataSource اسم الـ Data Control التي تصلنا بقاعدة البيانات ، والتي سيكون اسمها Data1 ،و ذلك لأن Data1 سوف تحوي مسار قاعدة البيانات المطلوبة و الجدول المطلوب .

العمليات الرئيسية على سجلات قاعدة البيانات:

- **إنشاء سجل جديد في قاعدة البيانات :**

Data1.RecordSet.Addnew

حيث أن :

Data1 : هو اسم الـ Data Control التي تصلنا بقاعدة البيانات المطلوبة ..

RecordSet : يقصد به مجموعة السجلات (التابعة للجدول المراد إضافة سجل اليه) و التي ترتبط بها الأداة Data1 .

- **تعديل بيانات السجل الحالي :**

```
Data1.Recordset.Edit
```

حيث يقوم هذا الأمر بحفظ التعديلات التي تجرى على السجل الحالي .

- **حفظ البيانات (أو التعديلات) في قاعدة البيانات :**

```
Data1.RecordSet.Update
```

و هي تجرى بعد عملية الإضافة أو التعديل (فقط) من اجل حفظ السجل الجديد داخل قاعدة البيانات .

- **حذف سجل من قاعدة البيانات :**

```
Data1.Recordset.Delete

Data1.Refresh
```

حيث أن هذا الآمر يقوم بحذف السجل الحالي ، و الخاصية Refresh تقوم تحديث بيانات قاعدة البيانات بعد عملية الحذف .

العمليات الرئيسية على سجلات قاعدة البيانات :

الانتقال إلى السجل الأول Data1.Recordset.MoveFirst

الانتقال إلى السجل الأخير Data1.Recordset.MoveLast

الانتقال إلى السجل التالي Data1.Recordset.MoveNext

الانتقال إلى السجل السابق Data1.Recordset.MovePrevious

العمليات الرئيسية على سجلات قاعدة البيانات:

- **أوامر البحث بين السجلات :**

```
Data1.Recordset.(طريقة البحث) " col_name = 'المطلوب' "
```

حيث ان col_name هو اسم العمود (في الجدول) المطلوب البحث بين حقوله عن القيمة المطلوبة ... و طرق البحث كما يلي :

Data1.Recordset.FindFirst للبحث عن أول سجل

Data1.Recordset.FindLast للبحث عن آخر سجل

Data1.Recordset.FindNext للبحث عن السجل التالي

Data1.Recordset.FindPrevious للبحث عن السجل السابق

آمر الوصول إلى عمود في الجدول :

Data1.RecordSet![اسم العمود]

مثال :

Data1.RecordSet![***name***]

- الدالة EOF (Data1. Recordset. EOF) تعيد القيمة True إذا كان المؤشر عند آخر سجل في الجدول .
- الدالة BOF (Data1. Recordset. BOF) تعيد القيمة True إذا كان المؤشر عند أول سجل في الجدول .
- الدالة Exclusive تحدد ما إذا كان الآخرون يستطيعون التعامل مع قاعدة البيانات أثناء استخدام التطبيق لها (Data1.Exclusive)

تطبيقات عملية

التطبيق الأول : لإضافة سجل إلى جدول (البيانات موجودة في مربعات نص غير مرتبطة بقاعدة البيانات **!!**) .

```
Data1. Recordset. AddNew

Data1. Recordset![name] = Txt1.Text

Data1. Recordset![address] = Txt2.Text

Data1. Recordset![phone] = Txt3.Text

Data1. Recordset. Update
```

التطبيق الثاني : الانتقال إلى السجل التالي و عرض البيانات في مربعات نص غير مرتبطة بقاعدة البيانات .

```
If Data1. Recordset. EOF Then
```

MsgBox " هذا آخر سجل "

Else

Data1. Recordset. MoveNext

Txt1. Text = Data1. Recordset![name]

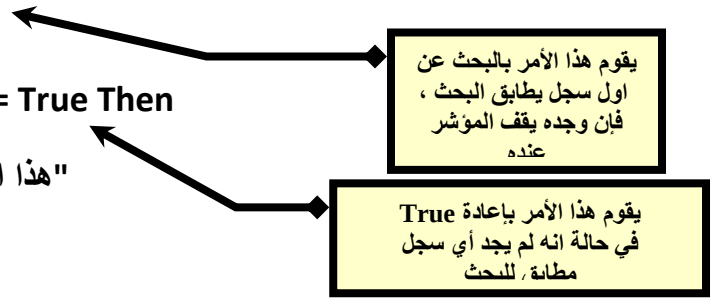
Txt2. Text = Data1. Recordset![address]

Txt3. Text = Data1. Recordset![phone]

End If

التطبيق الثالث : البحث عن سجل معين بحسب اسم الشخص

```
Dim name,str As String
name = InputBox(" أدخل الاسم المطلوب ", " بحث " )
If Len( name) = 0 Then
        MsgBox " لم تقم بإدخال أي بيانات"
        Exit Sub
End If
str= "name= ' " + name + " ' "    ' str = (name = ' الاسم المطلوب ')
Data1.Recordset.FindFirst str
If Data1.Recordset.NoMatch = True Then
        MsgBox " هذا الاسم غير موجود"
Else
        Txt1. Text = Data1. Recordset![name]
        Txt2. Text = Data1. Recordset![ address]
        Txt3. Text = Data1. Recordset![ phone]
End If
```

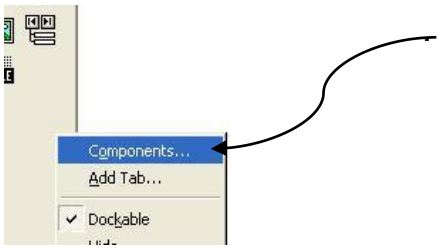


الربط بالأداة ADODC :

الأداة ADODC هي اختصار لـ ADO Data Control ، حيث تعتبر ADO من التقنيات الحديثة المهمة للارتباط بقواعد البيانات و التي يتم استخدامها ليس فحسب في لغة VB و لكن في العديد من لغة البرمجة . طريقة التعامل مع الأداة ADODC لا تختلف عن طريقة التعامل مع الأداة Data Control غير أنها تتيح التعامل مع قواعد البيانات الحديثة من قواعد بيانات Access على عكس الأداة Data Control التي لا تتعامل مع قواعد بيانات احدث من Access97.

- ***<u>طريقة اضافة الأداة ADODC إلى المشروع كما يلي :</u>***

نختار الأمر Component بالضغط على الزر الأيمن للفأرة في مربع الادوات من اجل اختيار الأداة من مربع الحوار Component

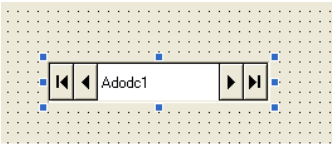


من مربع الحوار Component

نقوم باختيار الأداة :**MICROSOFT ADO DATA CONTROL 6. 0 (OLEDB)**

بعد الضغط على زر موافق ، تتم إضافة الاداة إلى ملف المشروع .

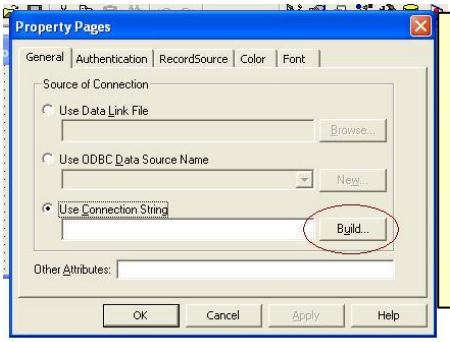
و أخيرا ، نقوم بإضافة الأداة إلى الـ Form.و يبقى معنا خطوات ربط الأداة ADODC بقاعدة البيانات المراد الارتباط بها ..



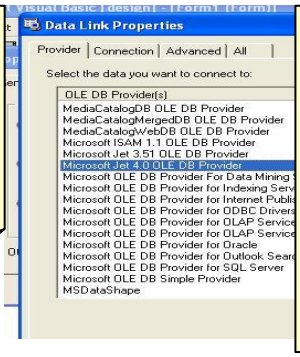
خطوات ربط الأداة ADOBC بقاعدة البيانات :

بعد أن تمت عملية إضافة الأداة ADODC إلى المشروع نقوم بربطها بقاعدة البيانات المطلوب الارتباط بها .

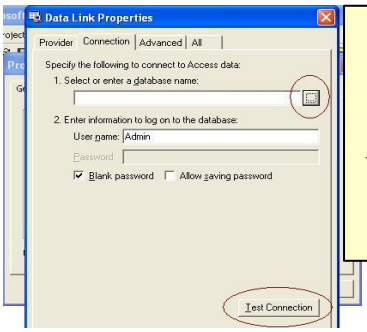
أولا : يتم ضبط خصائص الأداة بالدخول إلى خصائص الأداة من القائمة المنسدلة.



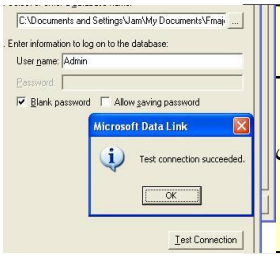
ثانياً : يتم الضغط على زر Build و الذي يبدأ سلسلة خطوات الارتباط بقاعدة البيانات.



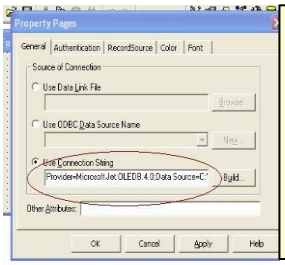
ثالثاً : في بداية الاتصال بقاعدة البيانات نحدد *المزود (المحرك)* المراد الاتصال به .. و هنا نحدد الاتصال بـ Access.



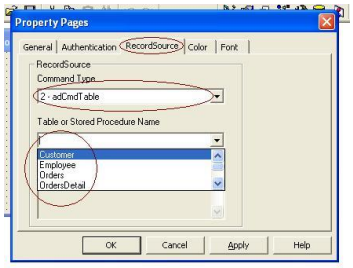
رابعاً : نحدد مسار قاعدة البيانات Access التي سيتم الارتباط بها ، و بعد ذلك نضغط على زر اختبار الاتصال للتأكد ان الاتصال قد تم



خامساً : في حالة ان الاتصال تم بنجاح ، سوف تظهر لنا النافذة الظاهرة في الشكل ، فنضغط على زر موافق.



سادساً : بعد ذلك سنعود إلى نافذة الخصائص من جديد ، و بها تفاصيل الارتباط ، بعدها نختار التبويب *RecordSource*



أخيرا ، حدد ان نوع الأوامر Command Type سيكون على الجداول من القائمة المنسدلة الأولى ، ثم حدد *الجدول الذي سيتم الارتباط به* ، و من الممكن استخدام *جمل SQL من اجل استرجاع السجلات التي سيتم الارتباط بها باستخدام جملة Select* مما يتح لنا التعامل مع أكثر من جدول في نفس الوقت (كيف؟)

عرض البيانات بـ Text BOXES و DataGrid:

يتم عرض بيانات حقول الجدول الذي تم الارتباط به بالأداة ADODC باستخدام مربعات النص بنفس الطريقة التي تم ذكرها عند ربط مربعات النص بالأداة Data Control (الخاصية Data Source لتحديد أداة الربط ADODC و الخاصية Data Filed لتحديد الحقل) .

من الممكن كذلك عرض البيانات داخل اداة خاصة تعرف باسم DataGrid (ليست الاداة Data Bound Grid التي تم استخدامها مع الاداة Data Control و لكنها تربط بنفس الطريقة و ذلك بتغير الخاصية Data Source إلى اداة الربط ADODC1) و هي كذلك تتميز بعرض بيانات جدول ما في قاعدة البيانات عل شكل جدول ، و يتم اختيارها من Component بإضافة الأداة Microsoft DataGrid

العمليات الرئيسية على الأداة ADODC :

إنشاء سجل جديد في قاعدة البيانات :

ADODC1.RecordSet.Addnew

حيث ان :

ADODC1 : هو اسم الـ ADODC التي تصلنا بقاعدة البيانات المطلوبة ..

RecordSet : يقصد به مجموعة السجلات (التابعة للجدول المراد إضافة سجل اليه) و التي ترتبط بها الأداة ADODC1 .

حفظ البيانات (أو التعديلات) في قاعدة البيانات :

ADODC1.RecordSet.Update

و هي تجرى بعد عملية الإضافة أو التعديل (فقط) من اجل حفظ السجل الجديد داخل قاعدة البيانات .

حذف سجل من قاعدة البيانات :

ADODC1.Recordset.Delete

ADODC1.Refresh

حيث أن هذا الأمر يقوم بحذف السجل الحالي ، و الخاصية Refresh تقوم تحديث بيانات قاعدة البيانات بعد عملية الحذف .

أوامر التنقل بين السجلات :

ADODC1.Recordset.MoveFirst

ADODC1.Recordset.MoveLast

ADODC1.Recordset.MoveNext

ADODC1.Recordset.MovePrevious

أمر البحث بين السجلات هو الأمر Find و الذي يحتوي على اربع وسائط ، يعتبر اولها الزامي و البقية اختياري :

ADODC1.Recordset.Find 1 , 2 , 3 , 4

1/ اسم العمود و الحقل المراد البحث عنه : "Name= 'Ali' "	2/ نقطة بداية البحث و توضع 0 افتراضياً او يتم البدء في البحث من النقطة المتوقف عندها المؤشر حالياً	3/ اتجاه البحث و هو احد خيارين : adSearchBackward أي وصولاً إلى اول سجل adSearchForward أي وصولاً إلى اخر سجل	4/ نقطة بداية البحث و لكننا نحدد هنا من أي صف سوف يبدأ البحث

امر الوصول الى الحقل المشار اليه حالياً من عمود في الجدول :

ADODC1.RecordSet![اسم العمود]

مثال :

ADODC1.RecordSet![***name***]

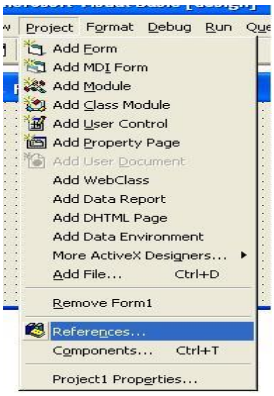
- الدالة EOF (ADODC1. Recordset. EOF) تعيد القيمة True اذا كان المؤشر عند اخر سجل في الجدول .
- الدالة BOF (ADODC1. Recordset. BOF) تعيد القيمة True اذا كان المؤشر عند اول سجل في الجدول .

الاتصال بقاعدة البيانات عن طريق المكتبة ADO

تعلمنا في الموضوع السابق كيفية الاتصال و التعامل مع قاعدة البيانات باستخدام الأداة ADODC ، و في هذا الدرس سوف نتعلم – إن شاء الله – كيفية الاتصال و التعامل مع قاعدة البيانات مع خلال المكتبة ADO التي يوفرها لنا Visual Basic .

المكتبة ADO تعتبر الأسلوب الجديد - قبل الأداة الجديدةADO.NET - للتعامل مع قاعد البيانات ، فمثلا يمكننا الاتصال وقراءة محتويات قاعدة بيانات اكسس Access بدون الحاجة لوجود Accessفهذه المكتبة توفر خدمة الاتصال والإجراءات المختلفة للتعامل مع قاعدة البيانات .

<u>خطوات اضافة المكتبة إلى ملف المشروع كما يلي :</u>



1. من خلال القائمة Project نختار References

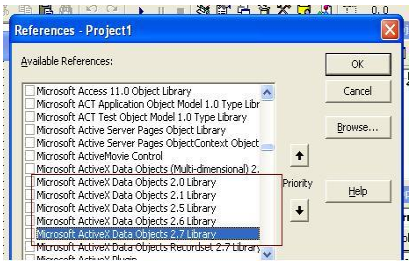
فتظهر لنا النافذة References – Project1 و

التي تحتوي على مكتبات عديدة ، ما يهمنا منها

هو مكتبة ADO .

2. نقوم باختيار المكتبة :

Microsoft ActiveX Data Objects 2.X Library



حيث أن الـ X يعبر عن الإصدار المثبت على

جهازك ، فأختر احد هذه الإصدارات و

اضغط موافق ليتم إضافة المكتبة ADO

إلى ملف المشروع .

العمليات الرئيسية التي تجرى باستخدام المكتبة ADO:

<u>اولاً : عملية الاتصال بقاعدة البيانات :</u>

يتم في البداية تكوين متغير يعبر عن قاعدة البيانات ، الهدف منه هو توصيف الاتصال بقاعدة البيانات ، بمعنى من هو مزود (محرك) قاعدة البيانات و ما هو مسار قاعدة البيانات ، كما يلي

Dim db As New ADODB.Connection *<u>(متغير عام)</u>*

حيث db يعتبر متغير (كائن بالمعنى الأصح) ، و هذا الكائن يستخدم لتوصيف الاتصال بقاعدة البيانات كما يلي :

db.Provider = "Microsoft.JET.OLEDB.4.0;"

db.Open "c:\db1.mdb "

الخاصية Provider تعني مزود (محرك) قاعدة البيانات و التي هي قاعدة بيانات Access

الخاصية Open تستخدم لتحديد مسار قاعدة البيانات

ملاحظات عند الاتصال بقاعدة البيانات :

- **الجملة :** **db.Provider = "Microsoft.JET.OLEDB.4.0; "**

تستخدم للاتصال بقاعدة بيانات من نوع Access ، و عند الاتصال بقاعدة بيانات SQL نستخدم

db.Provider = " SQLOLEDB. 1; "

- في حالة ان كانت قاعدة البيانات محمية بكلمة مرور (مثلاً jam هو اسم المستخدم و كلمة المرور هي 123) نقوم بفتح الجداول كما يلي :

db.Open "c:\SALE.mdb", "userID='jam'", "password=123"

- من الممكن اختصار جملتي تخصيص المزود و فتح جدول من قاعدة البيانات بإستخدام الخاصية ConnectionString و التي ستكون :

db. ConnectionString = "*provider =microsoft. jet. OLEDB. 4. 0;data source= c:\SALE.mdb*"

ثانياً : عملية فتح جدول من قاعدة البيانات :

يتم تكوين متغير يعبر عن الجدول (مجموعة السجلات) التي سيتم التعامل معه في قاعدة البيانات كما يلي :

Dim rs As New ADODB.Recordset *(متغير عام)*

حيث rs يعتبر متغير (كائن بالمعنى الأصح) ، و هذا الكائن يستخدم لتحديد مجموعة السجلات (الجدول) التي يتم الارتباط معها من اجل التعامل معها :

rs.Open "اسم الجدول", متغير قاعدة البيانات, نوع السجلات, نوع التزامن المطلوب

مثال :

rs.Open "[Table1]", db, adOpenStatic, adLockReadOnly

ثانياً : تابع عملية فتح جدول من قاعدة البيانات :

rs.Open 1, 2 , 3 , 4

اسم الجدول المراد الاتصال به	اسم المتغير الذي تم تعريفة من نوع ADODB.Connection	نوع التزامن : adLockBatchOptimistic غلق السجل من الاخرين اثناء التعامل معه ، ثم اطلاع السيرفر على التحديث adLockOptimistic غلق السجل من الاخرين اثناء التعامل معه adLockPessimistic غلق السجل اثناء التعديل فقط adLockReadOnly السجل للقراءة فقط	نوع السجلات المرتبطة : adOpenDynamic تعامل مباشرة من السجلات الحقيقة adOpenForwardOnly بالتحرك داخل السجلات في الاتجاه للأمام فقط adOpenKeyset للقراءة فقط مع التعديل adOpenStatic للقراءة فقط

عملية فتح جدول (جداول) من قاعدة البيانات باستخدام جمل SQL:

استخدام جمل SQL يتيح مرونة كبيرة من اجل استرجاع السجلات المراد التعامل معها ، حيث أننا نستطيع هنا إن نحدد الأعمدة المراد استرجاعها ، أو استخدام الشرط Where لتحديد السجلات المراد استرجعها و كذلك استرجاع السجلات من أكثر من جدول (بشرط وجود علاقة بين الجدول) و ذلك باستخدام JOIN .

مثال (1) : استرجاع أعمدة محددة (الأمثلة على قاعدة البيانات SALE) :

```
rs.Open "select CustomeName,Country from Customer",
db,adOpenKeyset, adLockReadOnly
```

مثال (2) : استرجاع سجلات من جداول بينها علاقة :

```
rs.Open “select Customer.CustomeName,Orders.OrderDate FROM
Customer INNER JOIN Orders ON Customer.CustomerID =
Orders.CustomerID",      db,adOpenKeyset, adLockReadOnly
```

استعراض بيانات حقول الجداول :

أمر الوصول إلى حقل في السجل المشار إليه حالياً هو :

rs![اسم العمود]

حيث rs هو اسم المتغير (الكائن) الذي تم تعريفه من نوع : ADODB.Recordset

- لاحظ انه عند التعامل مع المكتبة ADO سوف تختصر أوامر الأداة ADODC التي تحوي ADODC1.Recordset إلى اسم المتغير (الكائن) rs الذي تم تعريفه من نوع ADODB.Recordset ثم بقية الأوامر المعتادة .
- مثال لعرض بيانات الحقل ID :

```
If Not IsNull(rs![ID]) Then Text1 = rs![ID]
```

حيث ان الغرض من *Not IsNull* هو التأكد من الحقل الذي تم ارجاعه ليس فارغاً

إنشاء سجل جديد في قاعدة البيانات :

```
rs.Addnew
```

حيث ان :

rs : هو الكائن المعرف من نوع ADODB.Recordset و الذي سيجرى من خلاله التعامل مع بيانات قاعدة البيانات التي تم الاتصال معها .

<u>تعديل بيانات السجل الحالي :</u>

```
rs.Update
```

حيث يقوم هذا الأمر بحفظ التعديلات التي تجرى على السجل الحالي .

<u>حذف سجل من قاعدة البيانات :</u>

```
rs.Delete

rs.MoveNext
```

حيث أن هذا الآمر يقوم بحذف السجل الحالي ، و الخاصية MoveNext تقوم بالانتقال إلى السجل التالي بعد الحذف.

الدالة EOF (rs. EOF) تعيد القيمة True إذا كان المؤشر عند آخر سجل في الجدول .

الدالة BOF (rs. BOF) تعيد القيمة True إذا كان المؤشر عند أول سجل في الجدول .

تطبيقات عملية:

التطبيق (1) : عملية الإضافة ، على اعتبار أن البيانات المراد إضافتها موجودة في مربعات النصوص Text Boxes :

```
rs. AddNew

rs![name] = Txt1

rs![ address] = Txt2

rs![ phone] = Txt3

rs. Update
```

التطبيق (2) :عملية التعديل (حفظ التعديلات)،على اعتبار ان البيانات المراد تعديلها و حفظ التعديل موجودة في مربعات النصوص :

```
rs![name] = Txt1

rs![address] = Txt2

rs![phone] = Txt3
```

rs. Update

العمليات الرئيسية التي تجرى باستخدام المكتبة ADO:

ذكرنا سابقاً أن الأمر [ ]!rs هو للوصول إلى حقل المشار إليه من العمود ، و كذلك يمكن الوصول إلى الحقل المشار إليه حالياً من عمود في الجدول بالطريقة التالية :

rs.Fields(#no)

حيث ان no# تمثل رقم العمود بحسب ترتيب استرجاعه من قاعدة البيانات ، مثال :

rs.Fields(0)

و كذلك يمكن استخدام الطريقة التالية :

rs.Fields![اسم العمود]

و للاسترجاع القيمة الحقل الذي يشار اليه :

Text1.text= rs.Fields![اسم العمود] . Value

أوامر التنقل بين السجلات :

rs.MoveFirst

rs.MoveLast

rs.MoveNext

rs.MovePrevious

العمليات الرئيسية التي تجرى باستخدام المكتبة ADO :

أمر البحث بين السجلات هو الأمر Find و الذي يحتوي على أربع وسائط ، يعتبر أولها إلزامي و البقية اختياري :

rs.Find 1 , 2 , 3 , 4

1/ اسم العمود و الحقل المراد البحث عنه : “Name= ‘Ali’ ”	2/ نقطة بداية البحث و توضع 0 افتراضياً او يتم البدء في البحث من النقطة المتوقف عندها المؤشر حالياً	3/ اتجاه البحث و هو احد خيارين : adSearchBackward أي وصولاً إلى اول سجل adSearchForward أي وصولاً إلى اخر سجل	4/ نقطة بداية البحث و لكننا نحدد هنا من أي صف سوف يبدأ البحث

مثـــال :

rs.Find " Name = 'Ali' ", 0, adSearchForward, 1

الامر Find يسترجع سجل وحيد من السجلات التي تم الارتباط بها باستخدام الكائن rs ، و لكن ماذا إذا أردنا أن نسترجع أكثر من سجل أثناء عملية البحث ؟ الأمر الذي يستخدم لذلك هو استخدام الخاصية Filter ، كما يلي :

rs.Filter = "Country = 'سوريا'"

حيث ان السجلات التي تم استرجعها من قاعدة البيانات باستخدام الأمر rs.Open سوف يتم تصفيتها (فلتراتها) إلى السجلات التي قيمة الحقل فيها 'سوريا' من العمود Country .

لاحظ ان عملية التصفية باستخدام الأمر Filter تقوم بالتأثير على بيانات التي تم استرجعها باستخدام rs.Open ، و من اجل إلغاء عملية التصفية هذه ، و العودة إلى السجلات الأصلية التي تم استرجاعها باستخدام الأمر rs.Open نقوم بالتالي :

rs.Filter = ""

يستخدم الأمر rs.RecordCount لمعرفة عدد السجلات التي تم استرجاعها باستخدام الأمر rs.Open أو الأمر rs.Filter .

الأمر db.Close يستخدم لغلق قاعدة البيانات ، حيث ان db هو المتغير (الكائن) الذي تم تعريفه ADODB.Connection من اجل توصيف الاتصال بقاعدة البيانات .

عرض البيانات باستخدام المكتبة ADO:

عندما تحدثنا عن الأداة ADODC عرفنا انه من الممكن عرض البيانات داخل أداة خاصة تعرف باسم DataGrid وهي تتميز بعرض البيانات على شكل جدول ، و من الممكن أيضا استخدامها مع المكتبة ADODB ، و يتم اختيارها من Component بإضافة الأداة Microsoft DataGrid :

و من اجل عرض البيانات في الـ DataGrid باستخدام المكتبة ADO فإن الكود البرمجي الذي يلزمنا هو :

```
db.CursorLocation = adUseClient
Set DataGrid1.DataSource = rs
```

للإشارة إلى ان جهة التعامل مع قاعدة البيانات ليست Server بل Client

حيث إن rs يعتبر كائن و لذلك نستخدم الأمر Set عندما نريد أن نسند كائن إلى آخر . (لاحظ انه أولا يتم إضافة الـ DataGrid إلى الفورم) .

عرض البيانات في DataList و ComboList :

من الممكن كذلك عرض الحقول داخل قوائم منسدلة ، و يستفاد من ذلك في تقيد عملية إدخال البيانات من المستخدم و ذلك بالاعتماد على جداول الترميز .

- تستخدم الأداتان DataList و ComboList من اجل عرض البيانات داخل قوائم منسدلة ، و يتم إضافة الأداتان من Component بإضافة Microsoft DataList Control6.0 و سوف يظهران بالشكل التالي : 
- من اجل عرض بيانات عمود ما فيهما (حقل) نقوم بالبحث في خصائصهما عن الخاصية RowSource و إسناد أداة الربط بقاعدة البيانات ADODC1 اليها و الخاصية ListFeild و إسناد اسم العمود المراد عرضه اليها .
- إما عرض البيانات داخل DataList و ComboList باستخدام المكتبة ADODB فيلزم منا ذلك إسناد القيم إلى الخصائص RowSource و ListFeild برمجياً كما يلي (المثال التالي للأداة ComboList و هو لا يختلف أبدا عن عرض البيانات برمجياً في الأداة DataList) :

```
Set DataCombo1.RowSource = rs

DataCombo1.ListField = "CustomeName"
```

المرحلة الأخيرة ،، إنشاء التقارير :

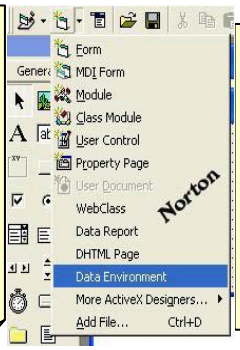
تعتبر التقارير هي الخلاصة الأخيرة للبيانات ، و التي يستفيد منها المستخدم (و التي يقاس من خلالها جودة أداء النظام) ، و يتم ربط التقارير بقاعدة البيانات ، و عرض سجلات محددة منها إما بشكل مباشر أو عن طريق جمل SQL ، و من الممكن جعل المستخدم تحديد التقرير الذي يريده و من ثم ترجمة ذلك إلى جملة SQL نعرض من خلالها التقرير المطلوب ، مع إمكانية إخراج ذلك التقرير بالطابعة للمستخدم .

يوفر Visual Basic بيئة جاهزة للاتصال بقاعدة البيانات تسمى هذه البيئة بـ DataEnvironment و بيئة أخرى جاهزة لإنشاء التقارير تسمى DataReport .

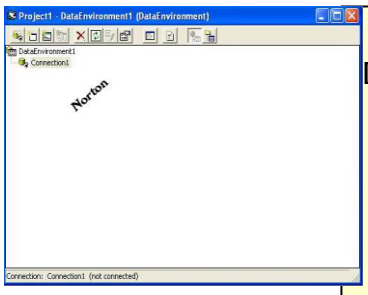
خطوات إنشاء التقارير:

يتم إنشاء التقرير على مرحلتين ، الأولى هي الاتصال بقاعدة البيانات عن طريق البيئة DataEnvironment ثم إنشاء التقرير عن طريق البيئة DataReport

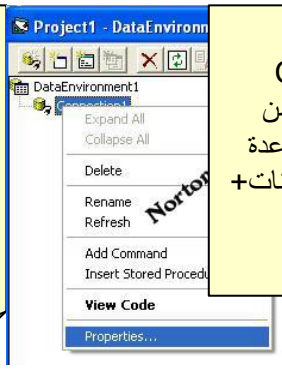
1/ يتم الاتصال بقاعدة البيانات عن طريق البيئة DataEnvironment ، و التي يتم انشاءها من القائمة Project ثم add DataEnvironment



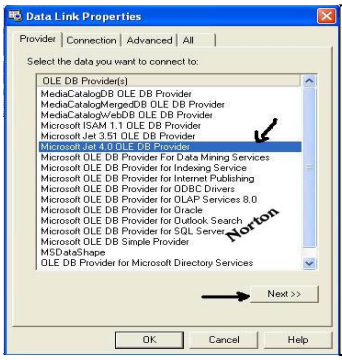
2/ يتم انشاء البيئة DataEnvironment1 و التي سوف تحتوي على عنصر Connection1 افتراضي نتصل من خلاله بقاعدة البيانات



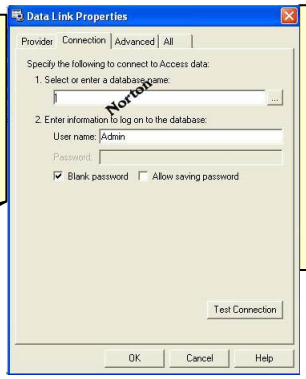
3/ من القائمة المنسدلة للعنصر Connection1 نختار الامر خصائص ، من اجل توصيف الاتصال بقاعدة البيانات (مزود قاعدة البيانات+ مسار قاعدة البيانات)



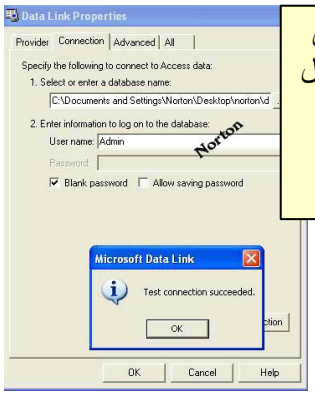
4/ في اول خطوة من خطوات توصيف الاتصال بقاعدة البيانات نحدد كما هو معتاد المزود (المحرك) الخاص بقاعدة البيانات المراد الاتصال بها



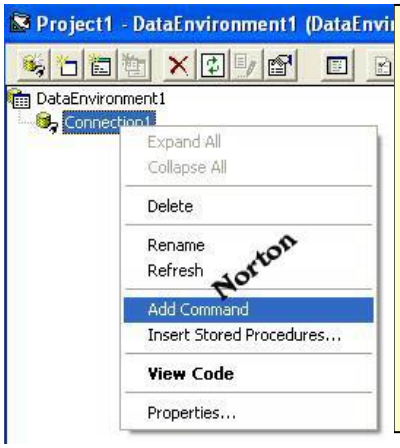
5/ ثانية خطوة من خطوات توصيف الاتصال بقاعدة البيانات هو ان نحدد مسار قاعدة البيانات من خلال الزر الذي يحتوي ثلاث نقاط



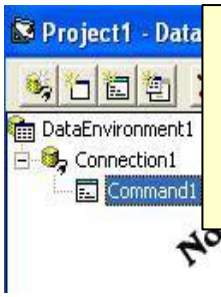
6/ أخيرا يتم التأكد من ان الاتصال تم بنجاح عن طريق تجربة الاتصال بالزر Test Connection



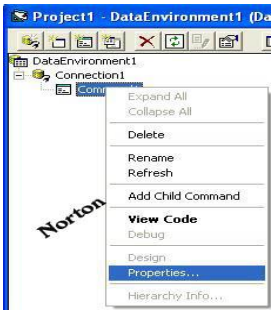
7/ بعد أن تم الاتصال بقاعدة البيانات ، نحدد من هي الحقول المراد عرضها في التقرير ، و قبل ذلك لابد من إنشاء أمر Command نجري من خلاله الاتصال بالحقول المراد عرضها و ذلك من القائمة Connection1 ثم نختار الأمر Add Command ، فيتم إنشاء Command جديد ، و من الممكن إنشاء أكثر من Commandحيث أن كل Command ممكن أن يشكل تقريراً ما ..



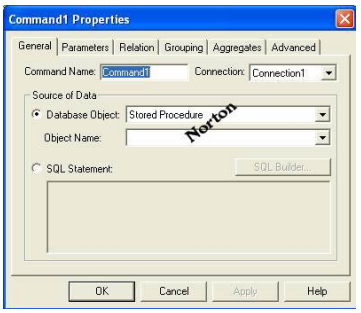
8/ نلاحظ الآن انه تم إضافة أمر Command1 في Connection1 (التي توصلنا بقاعدة البيانات) ، و من الممكن ان يكون لدينا أكثر من Command على قاعدة البيانات الواحدة .

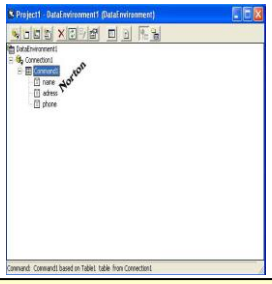


9/ الآن يتم الدخول إلى خصائص من القائمة المنسدلة الخاصة بالأمر Command من اجل تحديد الحقول المراد عرضها في التقرير .

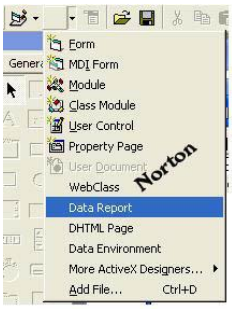


10/ من نافذة الخصائص ، العنصر Command Name يتيح لنا تغير اسم الـ Command إلى أي اسم مناسب لنا ، و في الأمر DataBase Object نحدد من القائمة المنسدلة اننا نريد الاتصال بجدول ، و في القائمة المنسدلة للأمر Object Name نحدد اسم الجدول ، و الخانة SQL Statement من الممكن نكتب جملة SQL نحدد من خلالها الحقول المطلوبة ، و بعد الموافقة نلاحظ أن الأمر Command كون شجرة بالحقول المطلوبة



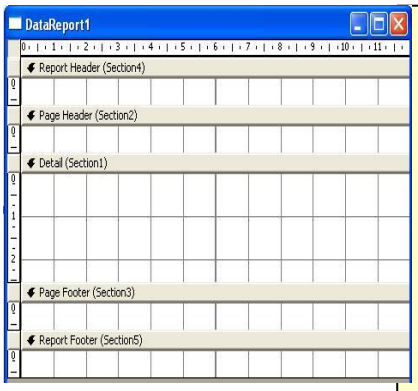


11/ بعد ان تمت عملية الارتباط بقاعدة البيانات ، و استخلاص الحقول المطلوبة ، نقوم بإنشاء التقرير ، و ذلك من القائمة Project ثم Add Data Report ، فنلاحظ انه تم إنشاء التقرير.



12/ الآن نقوم بربط التقرير بالـ DataEnvironment1 التي تحتوي توصيف قاعدة البيانات ، و ذلك من خلال الخاصية Data Source في خصائص التقرير

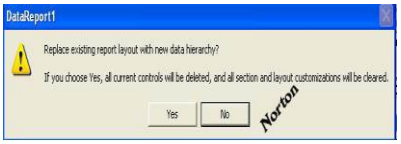
13/ بعد ذلك نقوم بتحديد الـ Commandالذي يحتوي على الحقول المراد عرضها في التقرير عن طريق الخاصية Data Member من خصائص التقرير .



14/ لاحظ شاشة التقرير ، سوف تجد انها مقسمة إلى خمسة اجزاء كما يلي :

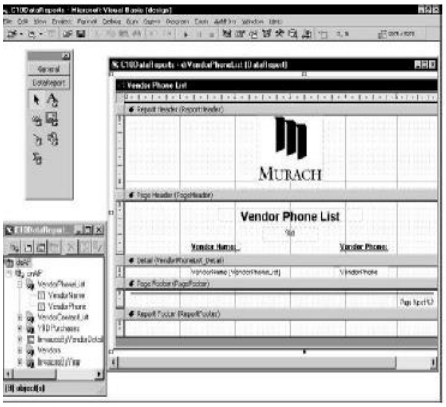
- الجزء الأول و الأخير Report Header – Report Footer يقصد به
الرأس و التذيل الذي سوف يظهر في الصفحة الاولى من التقرير .
- الجزء الثاني و الرابع Page Header – Page Footer يقصد به الرأس
و التذيل الذي سوف يظهر في كل صفحات التقرير .
- الجزء الأخير (الثالث) جز التفاصيل و ه مخصص لعرض تفاصيل التقرير فيه ، أي عرض حقول
قاعدة البيانات

15/ في أي مكان من شاشة التقرير اضغط بالزر الايمن و من القائمة المنسدلة اختر الامر Retrieve Structure و اضغط موافق على شاشة رسالة التنبيه التي تظهر .

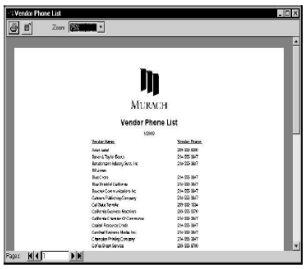


16/ الآن تبدأ مرحلة تصميم و تنسيق، بالنسبة لتنسيق التقرير من خلال صندوق الادوات نستطيع اضافة العديد من الادوات (Label, Image ...) و بالضغط على الزر الايمن و اختيار الامر Insert Control نستطيع اضافة ترقيم للصفحات او التاريخ ... الخ

17/ الآن تأتي المرحلة المهمة و هي عرض الحقول التي تم ادراجها ضمن الامر Command1 داخل التقرير ، و يتم ذلك بفتح شاشة التقرير و شاشة الـ DataEnvironment في آن معاً ، ثم نقوم بسحب اسم الحقل من الشجرة الخاصة بالـ Command إلى جزء التفاصيل في التقرير ... فنلاحظ انه تم انشاء خانة خاصة بالحقل داخل التقرير و خانة اخرى تمثل Label يوضح اسم الحقل .... و هكذا حتى ننقل كل الحقول امطلوبة إلى التقرير ... نستطيع تنسيق الحقول و البيانات التي يظهرها التقرير من خلال خصائص كل خانه و من المستحب نقل Labels الخاص بكل جزء إلى قسم Page Header من التقرير حتى يكون ظاهر في كل صفحة



18/ تأتي الآن اخر مرحلة .. و هي عرض التقرير داخل المشروع من خلال احدى النوافذ Forms و ذلك باستخدام الامر DataRepot1.Show



خطوات إنشاء التقارير الحي ☺:

- إلى الآن تم تصميم تقارير ناجحة باستخدام الـ Data Report ، غير أن هذه التقارير تقوم بعرض كل السجلات التي تحتويها قاعدة البيانات ... و حتى في حالة استخدام جملة SQL فإننا نلزم المستخدم بعرض محدد .. و من المرونة أن نجعل المستخدم يحدد نوع التقرير الذي يريده ، و ذلك (على سبيل المثال) بأن نجعل المستخدم يحدد ان التقرير الذي يريده هو ناتج عن البحث بحسب حقل معين (و ليكن تاريخ الطلبية ...) ، و يكون عرض التقرير ناتج ذلك البحث ، و هكذا فإننا سوف نكون قد كونا تقريراً يعرض الخانات المطلوبة حيث ان جملة الاستعلام التي يعتمد عليها التقرير سوف تعتمد على الشرط الذي يحدده المستخدم و هذا ما يسمى بالتقرير الحي ..

- بعد ان ننتهي من تجهيز DataEnvironment تحتوي على Commandو كذلك من ربط التقرير بـ DataEnvironment نقوم بما يلي :

1/ من خلال تصميم التقرير نغير الخاصية Data Field لمربعات النصوص إلى اسماء الحقول التي سوف تعرضها ..

2/ في داخل الفورم .. و عند نقطة البحث التي سيتم من خلالها استدعاء التقرير نضيف الشفرة التالية :

```
Private Sub Command1_Click()

Dim name, SQLstr As String
name = InputBox("ادخل اسم الدولة")

SQLstr = "select * from Customer where Country = '" & name & "'"

DataEnvironment1.Commands(1).CommandType = adCmdText
DataEnvironment1.Commands(1).CommandText = SQLstr
DataEnvironment1.Commands(1).Execute

DataReport1.Show

End Sub
```

الشفرة السابقة تقوم بالطلب من المستخدم ان يدخل اسم الدولة التي سيكون على أساسها التقرير ، و من ثم ينشئ استعلاماً مناسباً يسنده إلى الخاصية التالية من اجل جعل Command1 ينفذ الاستعلام :

DataEnvironment1.Commands(1).CommandText

و قبل ذلك يجعل Command1 مستعداً لاستقبال استعلام عن طريق الخاصية :

DataEnvironment1.Commands(1).CommandType = adCmdText

و من ثم ينفذ الاستعلام بالأمر :

DataEnvironment1.Commands(1).Execute

و أخيرا يعرض التقرير عن طريق الامر :

DataReport1.Show

انتهى والحمد لله رب العالمين