أخوكم في الله / أبوبكر شرف الدين (الهاكر) / ليبيا
صاحب الكتاب

بسم الله الرحمان الرحيم
هدا الكتاب أعيد دمجه في ملف أكروبات بواسطة :
هاني
إلى كل من يقرأهدا الكتاب أتمنى أن يدعو للوالدة الكريمة بالرحمة
كتب في :
٢٠٠٩فيفرييه
على الساعة العاشرة وأربعون دقيقة

شكرا جزيلا
من أراد العلا سهر الليالي
؟

الفصل الأول: فجول بيسك وبيئتها التطويرية

ما هي لغة فجول بيسك ؟

لغة فجول بيسك هي أداة عظيمة وبيئة تصميم وتطوير قوية تستخدم في هندسة وتصميم وتطوير البرامج المختلفة بحيث تكون البرامج المصممة على شكل يسمى (واجهة المستخدم الرسومية Graphical user interface).
هذه اللغة تعتمد في تنفيذ أوامرها ووظائفها على الأحداث ، فتسمى بالبرمجة المسيرة بالأحداث ، بمعنى أن جزءاً من البرمجة يبقى خاملاً حتى يتم إطلاق حدث معين ، وإذا تم إطلاق الحدث يتم تنفيذ البرمجة المرتبطة به فقط(١).
ومن أمثلة الأحداث: النقر المفرد Click أو تحريك مؤشر الفأرة MouseMove على عنصر معين أو غير ذلك.

بعض مزايا فجول بيسك :

تتضمن مجموعة كبيرة جداً من الكائنات objects.
إمكانية استخدام عدد كبير من الأيقونات والصور Icons and pictures في البرامج.
تدعم جميع أحداث ووظائف الفأرة Mouse ولوحة المفاتيح Keyboard.
تدعم استعمال الحافظة Clipboard والطابعة Printer.
تتضمن مجموعة كبيرة من الدوال الجاهزة والأوامر والوظائف الرياضية والمنطقية والرسومية وكذلك التعامل مع السلاسل الرمزية.
تتعامل مع الأنواع المختلفة من البيانات.
تتعامل مع الملفات بأنواعها المختلفة وبشتى طرق الوصول إلى البيانات (تسلسلية – عشوائية).
لديها إمكانيات هائلة في ميدان اكتشاف وتصحيح الأخطاء.
لديها مجموعة قوية من أدوات التعامل مع قواعد البيانات.
تدعم التعامل مع العناصر البرمجية المستوردة Activex controls.
تدعم طريقة جيدة في حزم البرامج وإنشاء برامج التنصيب setup programs بكفاءة عالية وأقل قدر ممكن من الأخطاء.
تدعم أدوات كثيرة تتعامل مع الشبكة الدولية Internet.
جاءت شعبية لغة فجول بيسك من خلال بيئتها التطويرية المتكاملة التي وفرت للمطور جميع السبل والأدوات اللازمة لتطوير برامج تطبيقية تعمل على منصة النظام Windows.
هذه البيئة وفرت متطلبات بناء البرامج التطبيقية المختلفة من محرر الكود Code Editor وعناصر التحكم (الأدوات) Controls لبناء واجهة المستخدم User Interface وتجربتها وتعديلها واكتشاف وتصحيح أخطائها ومن ثم تحويلها إلى برنامج تنفيذي (ذي امتداد EXE) قائم بذاته يعمل على جهاز المستخدم حتى في عدم وجود لغة فجول بيسك (بشرط وجود ملف محرك اللغة).

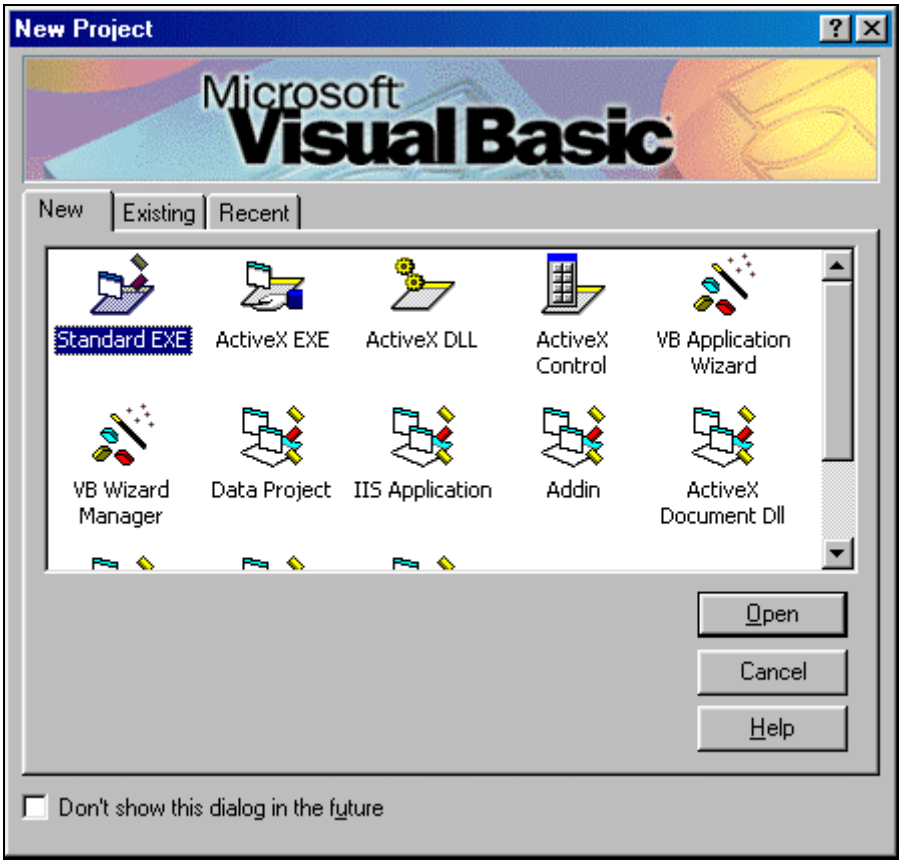
تشغيل بيئة التطوير المتكاملة (فجول بيسك):

هناك العديد من الطرق لتشغيل اللغة منها:
افتح قائمة (ابدأ Start) ثم ضع المؤشر فوق (البرامج Programs) ثم على المجلد Microsoft Visual Basic ٦،٠ ثم Microsoft Visual Basic ٦،٠.
أو إنشاء اختصار Short Cut للملف التنفيذي للغة على سطح المكتب Desk Top ثم النقر المزدوج عليه.
عند تنصيب اللغة على الجهاز يتم تسجيل الامتدادات Vbp ، frm ، Mod في سجلات النظام ، فعند النقر على إحدى هذه الملفات يتم تشغيل اللغة وفتح هذا الملف.

اختيار نوع المشروع Project:

عند تشغيل اللغة لأول مرة يتم إظهار مربع حوار Dialogue Box كما بالصورة يسألك عن نوع المشروع الذي تريد برمجته وتصميمه. هذا المربع به ثلاثة ألسنة تبويب Tabs هي:
New: يتيح لك اختيار نوع المشروع الجديد الذي تريد تصميمه.
Existing: تتيح لك فتح مشروع تم تصميمه من قبل.
Recent: ويتيح لك فتح آخر المشاريع التي اشتغلت عليها.

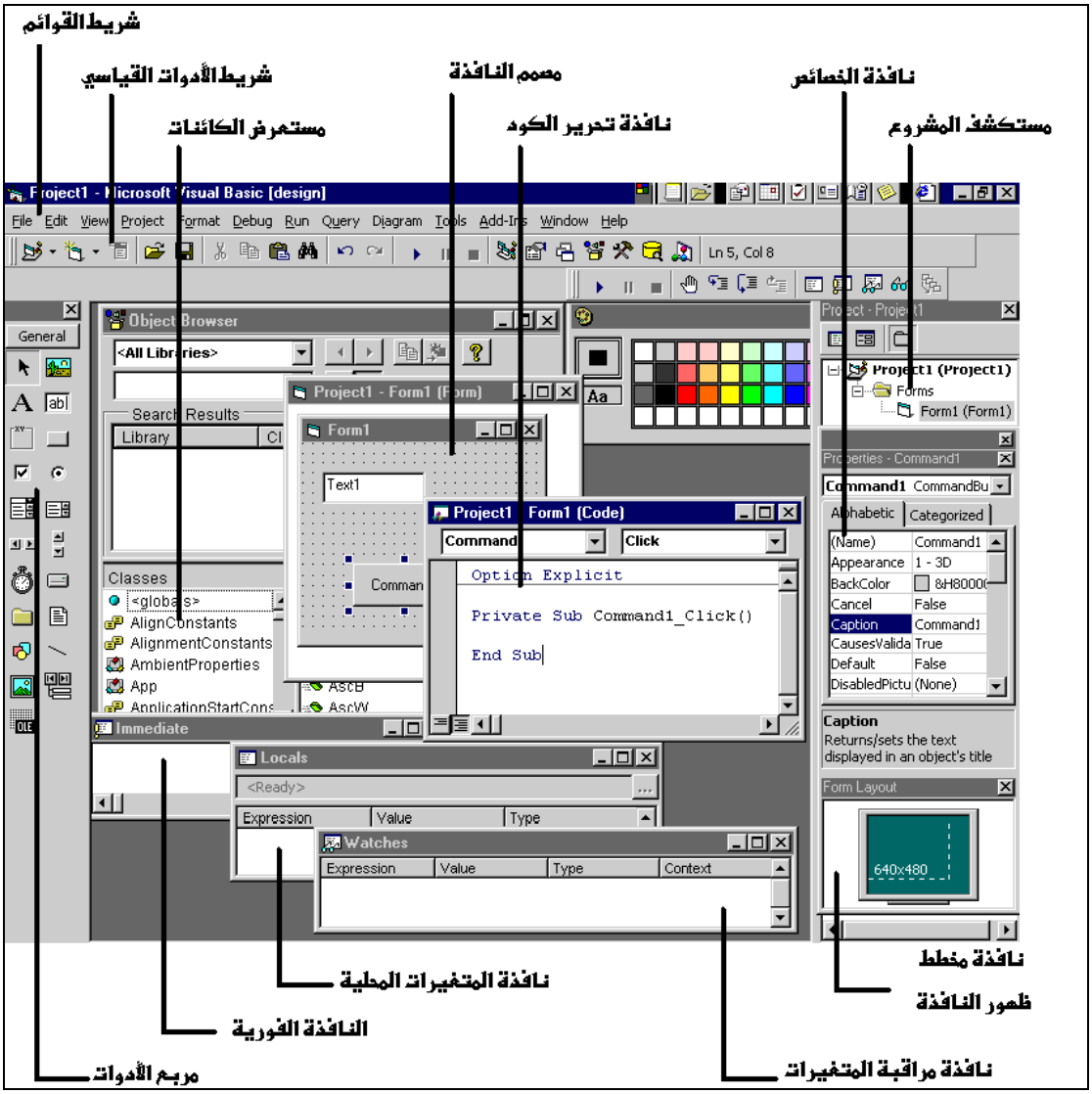
!! !ÐŒ¾ Ž!Œ!!! Œš! Œ !!!! !! !!!ƒ!!Î !



الآن سنختار اللسان الأول **new** لتصميم برنامج جديد ، بحيث يعرض عدة أنواع من المشاريع ، وفي هذا الفصل وفي الفصول القادمة نتحدث بالتفصيل عن النوع الأول (التلقائي) وهو **Standard EXE** فقط. لذا فبإمكانك الآن النقر المزدوج على هذا الخيار أو فقط الضغط على مفتاح الإدخال **Enter** للبدء في تصميم البرنامج.
وفي حالة رغبتك في عدم ظهور هذا المربع مجدداً عند بدء تشغيل اللغة انقر فوق الخيار **Don't show this dialog in the future** أسفل المربع ، وإذا أردت عرضه مجدداً افتح القائمة **Tools** ثم الأمر **Options** ثم افتح لسان التبويب **Environment** ثم الخيار **Prompt for project**.

مكونات لغة فجول بيسك ٦,٠:

بعد اختيار **Standard EXE** والنقر فوق **(Open)** يتم تحميل بيئة التطوير المتكاملة **IDE** ، والصورة توضح معظم نوافذ بيئة فجول بيسك:



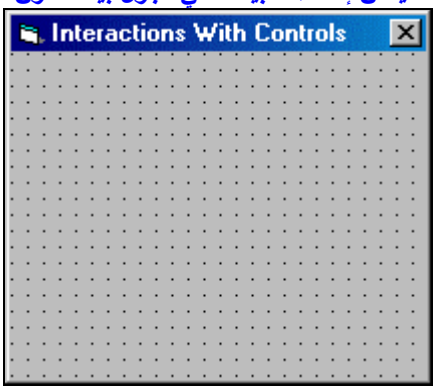
مكونات واجهة لغة فجول بيسك

النافذة الرئيسية Main window:

وتتكون من شريط العنوان Title bar وشريط القوائم menu bar وأشرطة الأدوات tool bars يحتوي شريط عنوان النافذة الرئيسية على اسم المشروع Project الحالي مضافاً إليه الوضع الحالي للبرنامج والنافذة form المعروضة حالياً ، بينما يحتوي شريط القوائم على مجموعة من القوائم التي من خلالها يتم التعامل مع بيئة فجول بيسك ، أما أشرطة الأدوات فتتكون من أزرار تمثل اختصاراً لأوامر القوائم أو تسهيل الوصول إلى أوامر القوائم. كما تُظهر النافذة الرئيسية لفجول بيسك موقع النافذة الحالية Current form بالنسبة للشاشة ومقدار ارتفاعها Height واتساعها Width بالبكسل.

مصمم نموذج النافذة Form Designer:

وهو أهم جزء في البرنامج ، فلا يمكن إنشاء تطبيقات في فجول بيسك دون مصمم النافذة (نموذج النافذة).



إضافة نافذة للمشروع:

لإضافة نافذة جديدة للمشروع نتبع الخطوات التالية:

افتح قائمة (project) ثم اختر الأمر (Add Form).

يظهر مربع حوار بعنوان (Add Form) اختر (Form).

انقر فوق (Open).

سيتم إضافة نافذة جديدة للمشروع ويتم عرضها في بيئة فجول بيسك.

ويمكن التبديل بين النوافذ من خلال النقر المزدوج على رمز في مستكشف المشروع.

إزالة نافذة من المشروع:

لإزالة نافذة من المشروع نتبع الخطوات التالية:

في نافذة مستكشف المشروع ، انقر على اسم النافذة بالزر الأيمن للفأرة.

تظهر قائمة منبثقة اختر منها الأمر (Remove FormName) علماً بأن FormName هو اسم النافذة البرمجي.

(قد) يسألك فجول بيسك هل تريد حفظ التغييرات عليه أم لا. أجب بأيها أحببت.

سيتم إزالة النافذة من المشروع.

ملاحظة /

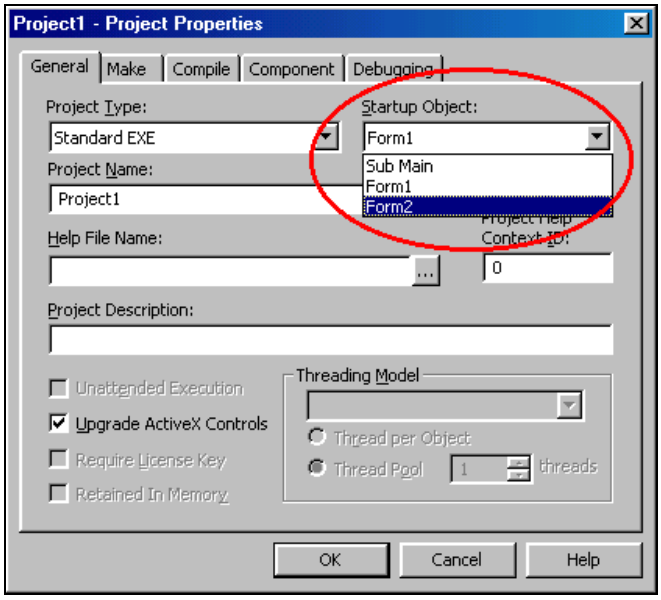
عند إزالة نافذة من المشروع هذا لا يعني حذف ملف النافذة ، بل يبقى الملف في الذاكرة لكن لا ينتمي لمكونات المشروع.

تحديد نافذة بدء التشغيل Setting Startup Form:

تستخدم فجول بيسك النافذة الأولى كنافذة بدء التشغيل تلقائياً ، لكن في حال رغبنا في تغيير نافذة بدء التشغيل نتبع الخطوات التالية:

افتح قائمة (Project) ثم اختر الأمر (ProjectName Properties) حيث أن ProjectName هو اسم المشروع الحالي.

يظهر مربع حوار بعنوان (Project Properties).



افتح القائمة المنسدلة المسماة (Startup Object) الموضحة بالصورة ، واختر منها اسم النافذة البرمجي لتكون هي نافذة بدء التشغيل.
انقر فوق الزر (Ok).

ملاحظة /
قد تحتوي قائمة Startup Object على عناصر مختلفة حسب نوع المشروع.

مربع الأدوات Tool Box:

وهي نافذة تحتوي على الأدوات الأساسية التي يتكون منها أي برنامج مصمم بلغة فجول بيسك ، وكما نلاحظ من الصورة فإن هناك ٢١ أداة تحكم Control في مربع الأدوات نستخدمها لتصميم واجهة المستخدم وهي كالتالي:

المؤشر Pointer: هو ليس بأداة وإنما يستخدم في حالة التراجع عن اختيار أداة معينة.

مربع التسمية Label: وهو أداة تعرض نصاً text لا يمكن للمستخدم تعديله بصورة مباشرة ، ويستخدم في عرض نتيجة عملية ما ، أو كعنوان يشرح وظيفة عنصر آخر.

مربع الرسم Picture Box: وهو أداة لعرض الصور ، وتستعمل كحاوية Container للعناصر والأدوات الأخرى.

مربع النص (صندوق النصوص Text Box): وهو أداة يستطيع المستخدم من خلالها إدخال قيم للبرنامج بصورة مباشرة.

الإطار Frame: وهو أداة تستخدم بصفة أساسية كحاوية للعناصر والأدوات الأخرى ، وكذلك في ضم العناصر المترابطة منطقياً أو وظيفياً مع بعضها ، وكذلك لتزيين واجهة المستخدم.

زر الأوامر Command Button: أداة تشبه الزر Button يؤدي كبسها (النقر عليها) إلى تنفيذ برمجة معينة كإنهاء تحميل البرنامج مثلاً.

زر الاختيار الأول Check Button: نستخدمها في حالة وجود عدة خيارات والمطلوب انتقاء خيار واحد أو أكثر منها ، وتسمح للمستخدم بالاختيار بين (نعم / لا) لنفس الخيار.

زر الاختيار الثاني Option Button: وهو أداة نستخدمها في حالة وجود عدة خيارات والمطلوب انتقاء خيار واحد فقط منها.

مربع التحرير والسرد Combo box: يسمح للمستخدم باختيار قيمة من قائمة منسدلة تشتمل على اختيارات محددة سلفاً أو بإدخال قيمة جديدة.

مربع السرد List Box: تسمح للمستخدم باختيار قيمة من قائمة تشتمل على عدة اختيارات محددة سلفاً.

شريط تمرير أفقي Horizontal ScrollBar يسمح للمستخدم باختيار قيمة بناءاً على موضع الزر على الشريط.

شريط تمرير رأسي Vertical ScrollBar مثل شريط التمرير الأفقي.

مؤقت Timer يتيح القيام بمهام معينة بعد مرور وقت معين.

مستعرض مشغلات Drive ListBox يتيح اختيار أحد مشغلات الأقراص.

مستعرض فهارس DirectoryListBox يتيح اختيار أحد الفهارس أو المجلدات الفرعية.

مستعرض ملفات FileListBox يتيح اختيار أحد الملفات.

شكل Shape يستخدم لرسم شكل هندسي على النافذة.

خط Line يستخدم لرسم خط على النافذة.

صورة Image مثل مربع الرسم إلا أنها تعرض صوراً فقط وتستهلك موارد أقل من النظام.

أداة ربط بيانات Data Control تربط بين قاعدة بيانات والبرنامج.

أداة ربط وتضمين OLE تربط بين البرنامج وخادم OLE.

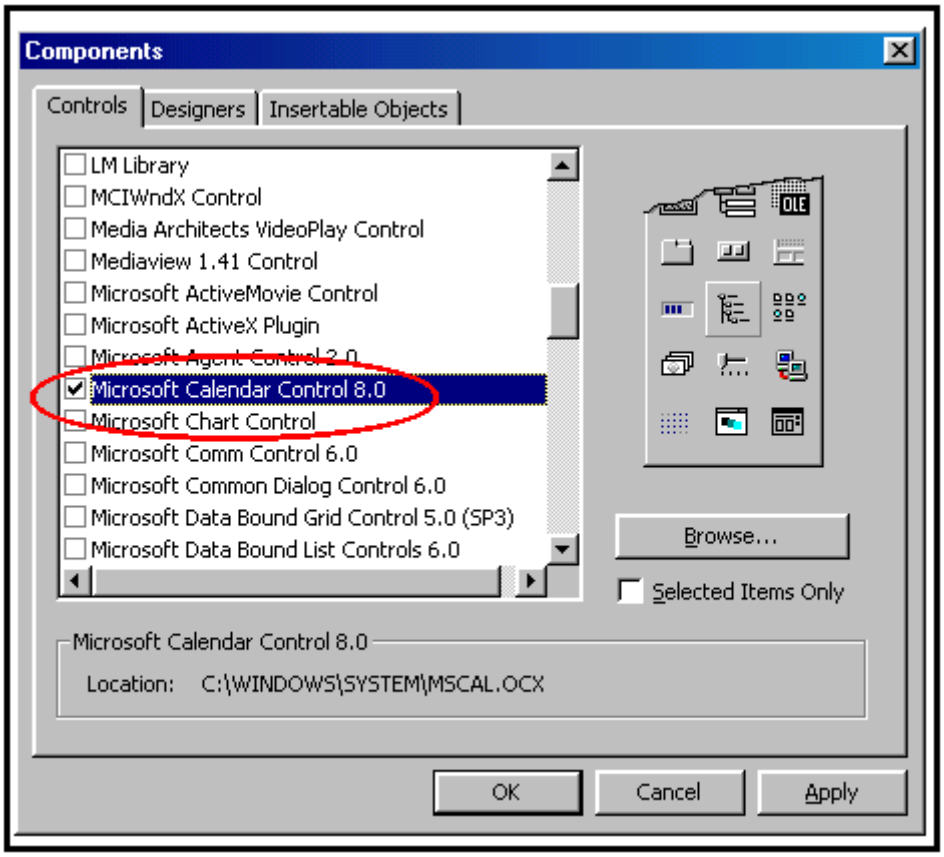
إضافة أداة تحكم إلى مربع الأدوات:

بالإمكان إضافة أي أداة أخرى غير الأدوات سالفة الذكر إلى مربع الأدوات وكالتالي:

انقر بالزر الأيمن للفأرة على أي ناحية في مربع الأدوات.

تظهر قائمة منبثقة Pop up menu اختر منها الأمر Components.

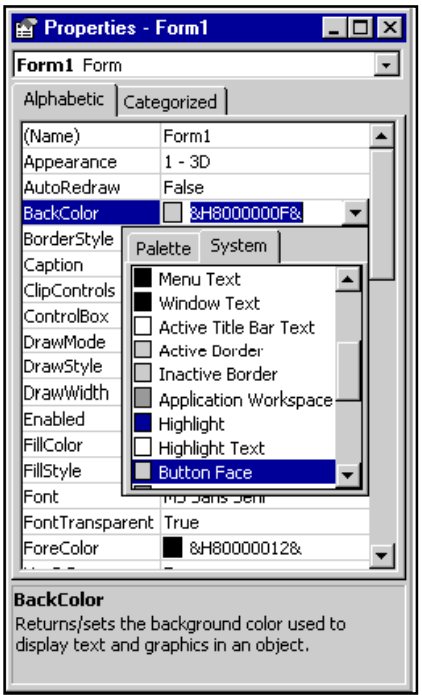
يظهر مربع حوار بعنوان Components وبه قائمة بالعناصر البرمجية التي يمكن إضافتها للبرنامج كما بالصورة ، وفي الصورة اخترنا الأداة ٦,٠ Microsoft Calendar Control:



اختر أحد هذه العناصر بتمكين علامة (صح) التي تقابلها.
انقر فوق الزر (موافق Ok). فيتم إدراج أيقونة في مربع الأدوات تمثل العنصر المضاف.

إذا كان عدد الأدوات الموجودة في المربع كبيراً جداً ، فيفضل أن تقوم بعملية تقسيم الأدوات إلى مجموعات تختفي وتظهر متى تشاء عن طريق النقر بزر الفأرة الأيمن على نافذة مربع الأدوات واختيار الأمر (Add Tab) من القائمة المنبثقة عن النقر ، ثم كتابة اسم المجموعة.
طريقة تنظيم محتويات كل مجموعة تتبع أسلوب السحب والإلقاء Drag & Drop وهو نفس الأسلوب الذي تتبعه لنقل أو نسخ ملفات جهازك. أخيراً إذا أردت حذف المجموعة قم بالنقر على اسم المجموعة بزر الفأرة الأيمن واختيار الأمر (delete Tab) من القائمة ، مع العلم بأن المجموعة الرئيسية والمسماة General لن تتمكن من حذفها.

إزالة أداة تحكم من المشروع:
لكي تحذف أداة تحكم Control من المشروع اتبع الخطوات التالية:
إذا كانت الأداة موضوعة على النافذة أزلها بتحديدها ثم الضغط على المفتاح Delete.
افتح القائمة (Project) ثم اختر الأمر (Components).
يظهر مربع حوار بعنوان (Components).
امسح مربع الاختيار الذي يمثل الأداة المراد حذفها.
انقر فوق الزر (Ok).
فيتم إزالة الأداة.



إذا لم يكن مربع الأدوات معروضاً أمامك افتح قائمة عرض View ثم اختر الأمر ToolBox.

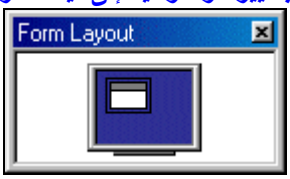
نافذة الخصائص Properties Window:
هذه النافذة تعرض جميع خصائص الكائن المختار سواء كان نافذة Form أو أداة (عنصر تحكم) Control مختارة ، وتتيح لك إمكانية تغيير هذه الخصائص وقت التصميم design Time. فعلى سبيل المثال : يمكنك تغيير لون خلفية النافذة (وهو إحدى خصائص النافذة) أو

لون خلفية أي أداة مختارة تدعم هذه الخاصية من خلال خاصية تسمى: BackColor.

تعرض هذه الخصائص بترتيب أبجدي عدا الخاصية Name فإنها تأتي في بداية قائمة الخصائص ، وعند اختيار خاصية معينة يتم عرض وصفها أسفل نافذة الخصائص.
وعند اختيار أي عنصر أو أداة يتم عرض جميع خصائصها ضمن هذه النافذة مباشرة وبصورة آلية. أما في حالة اختيار أكثر من أداة من نفس النوع (يتم عن طريق اختيار الأداة الأولى ثم الضغط المستمر فوق المفتاح Ctrl ثم النقر على باقي الأدوات الأخرى) فيتم فقط عرض الخصائص المشتركة فيما بينها. إذا لم تكن نافذة الخصائص ظاهرة أمامك اضغط المفتاح F٤ أو النقر على الأيقونة التي تمثل النافذة من شريط الأدوات Tool Bar.

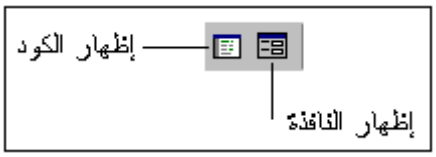
مربع تخطيط النافذة Form Layout Window:

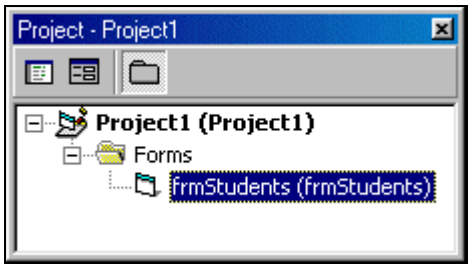
يظهر هذا المربع على شكل شاشة حاسوب وبه صور للنوافذ Forms التي يتضمنها المشروع الحالي ، ومن خلال هذا المربع نستطيع التحكم في مكان وكيفية ظهور النوافذ عند تشغيل البرنامج ، فقط قم بسحب الرسم الذي يمثل نافذة معينة إلى المكان المناسب. كما يمكنك التعرف من خلاله على درجة وضوحية الشاشة Screen resolution من خلال النقر بالزر الأيمن للفأرة على المربع واختيار الأمر Resolution Guide ، وبالتالي نعرف كيف ستظهر النوافذ فيما لو قام مستخدم البرنامج بتغيير الوضوحية إلى قيمة أخرى غير التي تم فيها تصميم البرنامج.



نافذة المشروع Project window:

تعرض قائمة بجميع مكونات المشروع من نماذج نوافذ Forms ووحدات نمطية Modules وملفات Files ، تعرض كل مجموعة متجانسة في مجلد مستقل فيكون هناك مجلد خاص بنماذج النوافذ وآخر للوحدات النمطية وهكذا.
ويمكن من خلال هذه النافذة التبديل بين صورة نموذج النافذة وبين شاشة محرر الكود Code editor من خلال أيقونتين ، الأولى تظهر النموذج Form والأخرى تظهر الكود.





نافذة محرر الكود Code Editor:

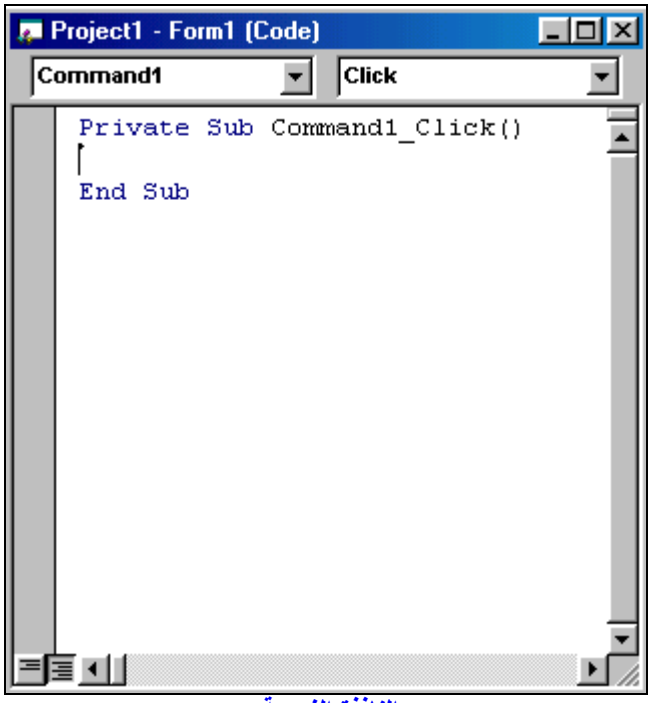
ومن خلالها يتم كتابة الكود الذي غالباً ما يكون ضمن إجراءات أحداث Event procedures ويتم تنفيذ كل إجراء عند إطلاق (أو استثارة أو تفجير) هذا الحدث ، ويكون الإجراء الحدثي مرتبطاً بطبيعة الحال مع نموذج النافذة أو أحد العناصر البرمجية المرسومة عليه. ولكي نقوم بكتابة البرمجة الخاصة بحدث النقر المفرد Click لزر الأوامر Commandbutton مثلاً نقوم بالنقر عليه نقرتين سريعتين أثناء وقت التصميم Design Time فيتم فتح نافذة تحرير الكود وبها السطران البرمجيان التاليان:

```
Private Sub Command1_Click( )

End Sub
```

فالسطر الأول يتكون من الكلمات المحجوزة Private Sub ثم الاسم البرمجي لزر الأوامر Command١ متبوعاً بحدث النقر Click ، والسطر الثاني عبارة عن نهاية الإجراء الحدثي End Sub. نكتب البرمجة التي نريد تنفيذها عند النقر المفرد على الزر أثناء تشغيل البرنامج بين هذين السطرين.

هناك مفاتيح اختصار تقوم بإظهار نافذة الكود إذا لم تكن ظاهرة أمامك وهي الضغط على المفتاحين Shift + F٧ معاً.
ولإخفاء نافذة محرر الكود وإظهار نافذة مصمم نموذج النافذة اضغط المفتاح F٧.

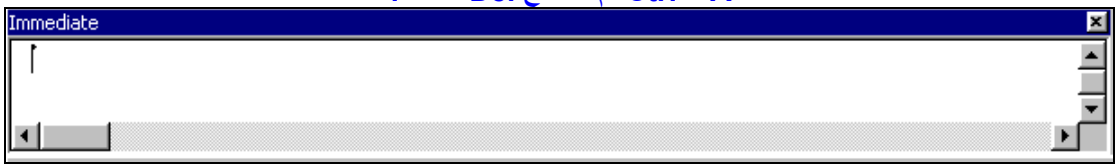


النافذة الفورية:

وهي أداة جيدة تتيح لك إدخال أمر أو تعبير فجول بيسك ومن ثم تعرض نتيجة تنفيذه باستخدام الأمر **Print** أو باستخدام علامة الاستفهام ؟ ، وكذلك تستخدم لعرض قيمة متغير أو تعبير أثناء وضع التوقف المؤقت **Break mode** للبرنامج. كذلك يمكن طباعة النتائج من البرنامج إلى هذه النافذة بكتابة **Debug.Print** أثناء فترة تجريب البرنامج.

هذه النافذة لا تتوفر بعد تحويل البرنامج إلى ملف تنفيذي ، لذا يجب إزالة العبارة السابقة من البرنامج لأنها ستسبب في حدوث رسالة خطأ ومن ثم إغلاق البرنامج اضطرارياً. لكن تستخدم فقط لغرض تجربة البرنامج وتنقيحه وتصحيح الأخطاء قبل تحويله إلى الملف التنفيذي.

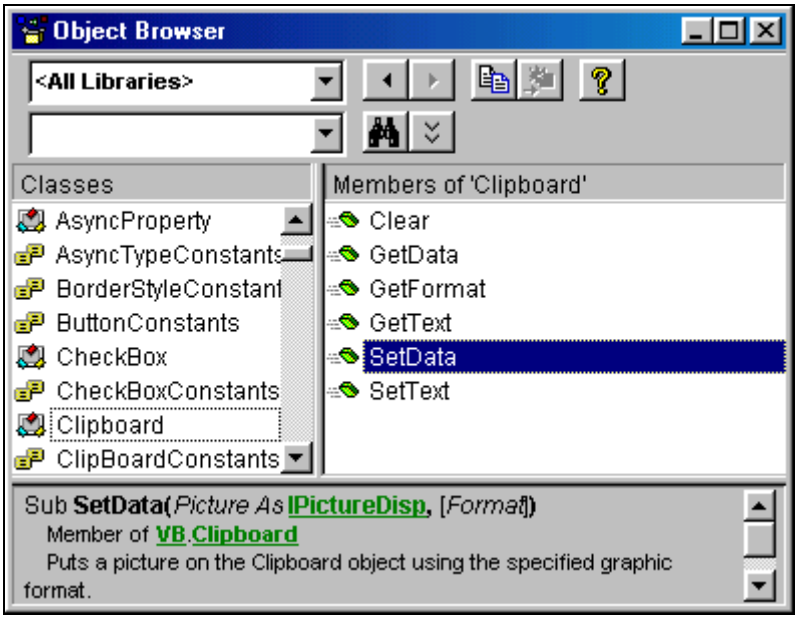
إذا لم تكن النافذة هذه ظاهرة أمامك اضغط المفتاحين **Ctrl + G** معاً ، ولمسح محتوياتها اضغط المفتاحين **Ctrl + A** ثم المفتاح **Del** للحذف.



النافذة الفورية

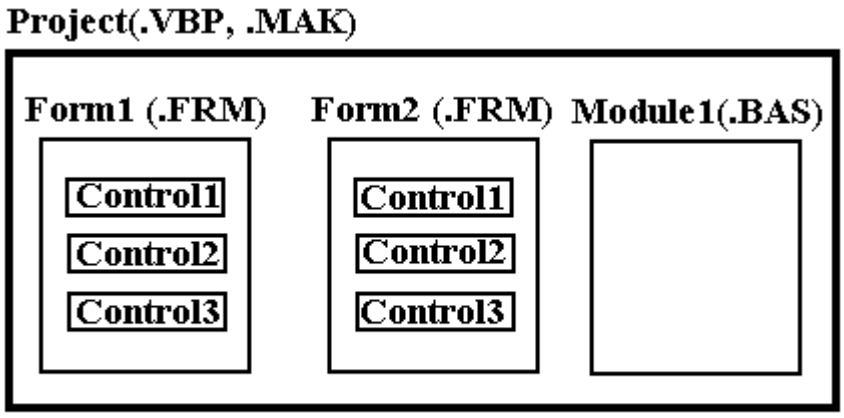
نافذة مستعرض الكائنات **Object browser**:

تعرض هذه النافذة جميع الفئات الموجودة في المكتبات المضمنة في برنامجك مع كافة وظائفها وطرقها وخصائصها وأحداثها ، لتعطيك فكرة عامة عن محتويات هذه المكتبات ، ولها ميزة أخرى وهي سهولة إيجاد المساعدة **help** المرفقة بكل خاصية أو حدث أو وظيفة عن طريق النقر بزر الفأرة الأيمن على العنصر المطلوب واختيار الأمر **help** من القائمة. إذا لم تكن النافذة معروضة أمامك اضغط المفتاح F٢.



مكونات برنامج تطبيقي مصمم بلغة فجول بيسك ٦,٠:

يتكون أي برنامج تطبيقي متكامل في لغة فجول بيسك ٦,٠ من الأجزاء التالية كما بالرسم التوضيحي:



مكونات مشروع قياسي

Forms نموذج النافذة: ومن خلاله يتم تصميم واجهة المستخدم.

Controls العناصر البرمجية: وهي عناصر برمجية (أدوات) يتم رسمها على النوافذ لبناء واجهة المستخدم التي تتيح للمستخدم التعامل مع البرنامج مثل (صندوق النصوص **TextBox** وعنصر التسمية **Label** وزر الأوامر **Commandbutton** وغيرها). إن النوافذ والعناصر البرمجية (الأدوات) يطلق عليها جميعاً اسم (كائنات **Objects**).

Properties الخصائص: لكل كائن مجموعة من الخصائص تحدد كيفية ظهوره وطريقة عمله. منها على سبيل المثال: الاسم البرمجي **name** ولون الخلفية **Backcolor** وغيرها. هناك خصائص يتم تحديدها وقت التصميم **Design Time** فقط من خلال نافذة الخصائص **Properties Window** ، وهناك خصائص أخرى تحدد وقت التشغيل فقط **Run Time** من خلال الكود (البرمجة) ، وهناك خصائص أخرى يمكن تحديدها (ضبطها) في الوقتين. ولضبطها برمجياً نكتب اسم الكائن متبوعاً بنقطة ، فتظهر قائمة بالخصائص والوظائف التي تتبع هذا الكائن ، فنختار الخاصية المطلوبة ونعدل قيمتها كما في المثال:

Object.Property = NewValue

Methods الوظائف: وهي دوال مبنية داخل الكائن نفسه تؤدي وظيفة معينة تكون متعلقة بسلوك وعمل هذا الكائن ، ويتم استدعاؤها كذلك بطريقة سهلة وبسيطة وهي بكتابة الاسم البرمجي للكائن ثم النقطة (.) ثم اسم الوظيفة كما بالمثال:

Form1.Show

في هذا المثال تم إظهار النافذة الأولى عن طريق استدعاء الوظيفة **Show** التابعة لها.

ملاحظة / عندما تكتب اسم الكائن (صحيحاً بالطبع) ثم تطبع النقطة يتم عرض قائمة بالخصائص والوظائف التي يدعمها هذا الكائن ، وعند كتابة الحرف الأول للخاصية ينتقل المؤشر إليها ، وإذا اشتركت عدة خصائص أو وظائف بنفس الحرف الأول يظهر المؤشر على أولها ، وعند كتابة الحرف الثاني للخاصية أو الوظيفة يتم الانتقال للخاصية أو الوظيفة التي يتكون اسمها من هذين الحرفين وهكذا. وإذا وقف المؤشر عند الخاصية أو الوظيفة المطلوبة نستخدم (المسطرة Space Tab) فيتم كتابتها كاملة ، هذه الخاصية تسمى خاصية الإكمال التلقائي Auto complete.

Events الأحداث: وهي عبارة عن برمجة يتم تنفيذها عند وقوع (إطلاق) حدث معين كما أسلفت.

General Procedures الإجراءات العامة: وهي برمجة غير مرتبطة بكائن ، ويتم استدعاؤها من أي مكان بالبرنامج.

Modules الوحدات النمطية: وهي مجموعة من الإجراءات العامة ، وجُمل تحديد نوع المتغيرات ، وجمل تعريف الثوابت المستعملة من قبل البرنامج وغيرها.

خطوات تصميم برنامج تطبيقي في فجول بيسك ٦,٠:

عند تصميم أي برنامج تطبيقي في فجول بيسك لا بد أن تمر مرحلة التصميم بالخطوات الثلاثة التالية:

وضع (رسم) عناصر التحكم Controls أو الأدوات Tools على نموذج النافذة form.

ضبط خصائص Sitting Properties نموذج النافذة والعناصر التي عليه.

كتابة الكود المرتبط بكل أداة.

الخطوة الأولى: رسم العناصر والأدوات على نموذج النافذة:

بعد تشغيل بيئة التطوير المتكاملة واختيار نوع المشروع نكون جاهزين لوضع العناصر البرمجية على النافذة التي ستكون فارغة بطبيعة الحال.

ولكي نضع (نرسم) العنصر على النافذة نقوم بالآتي:

ننقر على الأيقونة التي تمثل العنصر في مربع الأدوات نقرة واحدة فيتم اختياره.

نذهب إلى النافذة فيكون شكل مؤشر الفأرة كالإشارة (+).

نرسم العنصر على النافذة بنفس الطريقة التي نرسم بها مربعاً في برنامج الرسام.

بعد وضع الأداة على النافذة لا بد من تعديل خصائصها وموقعها وأبعادها بحيث تناسب احتياجات المستخدم من البرنامج.

وفي حالة أنك تريد وضع أكثر من عنصر من نفس النوع على النافذة ، قم بوضع العنصر الأول ، ثم انقر عليه نقرة واحدة واضغط المفتاحين Ctrl + C لأخذ نسخة منه ، بعد ذلك اضغط المفتاحين Ctrl + V للصقه. وفي هذه الحالة سيظهر لك مربع رسائل يسألك هل ستضع العنصر ضمن مصفوفة من نفس النوع أم لا ، اختر (لا) فيتم وضع العنصر النسخة في الركن الأيسر العلوي للنافذة.

تحريك العناصر البرمجية:

إذا لم يكن مكان تواجد العنصر مناسباً لاحتياجاتك تستطيع تحريكه إلى المكان المناسب كما يلي:

انقر فوق العنصر دون أن ترفع إصبعك عن زر الفأرة.

حرك الفأرة إلى المكان المناسب.

حرر زر الفأرة.

فيتم وضع العنصر في المكان المطلوب.

هذا عن طريق الفأرة ، أما عن طريق لوحة المفاتيح ، وفيتم اختيار العنصر بالنقر عليه نقرة واحدة ثم الضغط على المفتاح Ctrl وأحد مفاتيح الأسهم فيتم تحريك العنصر بالاتجاه المطلوب.

تغيير حجم الأداة:

أولاً : باستخدام الفأرة:

عند الرغبة في تغيير أبعاد الأداة نتبع الخطوات التالية:

انقر فوق العنصر على النافذة نقرة واحدة فيتم إظهار ثمانية مربعات صغيرة تحيط بالأداة تسمى بمربعات تغيير الحجم.

ضع مؤشر الفأرة على أحد هذه المربعات وانقر دون أن تحرر زر الفأرة.

حرك الفأرة في اتجاه ذلك المربع.

فيتم تغيير حجم الأداة.

ثانياً : باستخدام لوحة المفاتيح:

انقر فوق الأداة نقرة واحدة.

استخدم المفتاح Shift مع أحد مفاتيح الأسهم لتغيير الأبعاد.

الخطوة الثانية : ضبط الخصائص :

إن ضبط الخصائص يعني وضع المواصفات التي تتحكم في مظهر وسلوك الأدوات أثناء تشغيل البرنامج ، ويتم ضبط الخصائص وقت التصميم عن طريق نافذة الخصائص **Properties Window**.
كل خاصية تتبع أي كائن سواء كان نافذة أو عنصر تحكم تقابلها قيمة قد تكون عددية صحيحة أو قد تكون منطقية أو غير ذلك. وفي حالة وجد أكثر من قيمة محتملة للخاصية نلاحظ وجود زر عليه سهم متجه لأسفل على يمين الخاصية ، عند النقر عليه تنسدل قائمة بها كافة القيم المحتملة للخاصية يتم اختيار إحداها بالنقر عليها. وإذا أردت معرفة المزيد عن خاصية ما انقر عليها فيتم عرض وصف لهذه الخاصية أسفل نافذة الخصائص أو اضغط المفتاح F١.

الخطوة الثالثة : كتابة الكود (البرمجة):
كما سبق وأن أوضحت فإن تنفيذ البرمجة في فجول بيسك يتم عند إطلاق حدث معين ، لذا يجب تحديد أي حدث سيتم تنفيذ البرمجة عند حدوثه أولاً ، وكذلك متى سيتم إطلاق الحدث ، ثم يتم تنفيذ البرمجة المرتبطة بهذا الحدث والموجودة في إجراء حدثي **Procedure** المكتوب في نافذة تحرير الكود.

أوضاع البرنامج في فجول بيسك:
يمر البرنامج بالأوضاع التالية:
وضع التصميم **Design mode**: وفيه يتم تصميم البرنامج.
وضع التنفيذ **Run mode**: وفيه يتم تنفيذ البرنامج.
وضع التوقف المؤقت **Break mode**: وفيه يتم التوقف عن التنفيذ نظراً لوقوع خطأ ما ، وغالباً ما يُستفاد من هذا الوضع في تنقيح البرنامج واكتشاف وتصحيح الأخطاء.

قوائم بيئة التطوير المتكاملة:
١. القائمة **File**:
تحتوي هذه القائمة على أوامر أساسية خاصة للمشاريع بشكل عام ، كإنشاء مشروع جديد ، حفظ محتويات المشروع ، طباعة محتويات المشروع وترجمة المشروع وتحويله إلى ملف ثنائي وغيرها.

٢. القائمة **Edit**:
تحتوي على أوامر التحرير القياسية كالقص والنسخ واللصق ، ونستخدمها بشكل خاص عند التعامل مع نافذة محرر الكود.

٣. القائمة **View**:
تحتوي على أوامر لعرض مختلف النوافذ المكونة للواجهة مثل نافذة مربع الأدوات ونافذة الخصائص وغيرها.

٤. القائمة **Project**:
معظم أوامر خاصة بمحتويات المشروع ، فهي تمكنك من إضافة عنصر أو مجموعة عناصر من عناصر المشروع كالنوافذ **Forms** والوحدات النمطية (ملفات البرمجة) **Modules** والفئات **Classes** وغيرها.

٥. القائمة **Format**:
وهي خاصة بتنسيق الأدوات التي نضعها على النافذة من ناحية موقعها على النافذة. بالإضافة إلى تغيير ترتيب الأدوات ، أي وضع أداة فوق الكل أو أداة خلف الكل وغير ذلك.

٦. القائمة **Debug**:
وتحتوي على معظم أوامر التنقيح ، من هذه الأوامر اختيار طريقة تنفيذ البرنامج ، كتنفيذ سطر واحد منه **Step Into** ، إجراء كامل **Step Over** ، أمر سابق **Step Out** أو التنفيذ حتى السطر الذي يوجد فيه مؤشر الكتابة **Run To Cursor** ، وبالنسبة علامات القطع **Break Points** فهي علامات تظهر مبدئياً باللون الأحمر على سطر معين بحيث تتم عملية الإيقاف المؤقت للبرنامج عند الوصول إلى هذه العلامات.

٧. القائمة **Run**:
ومنها نستطيع تنفيذ البرنامج وتمكننا من اختيار الأوامر الأخرى كالإيقاف المؤقت **Break** أو إنهاء تنفيذ البرنامج **End**.

٨. القائمة **Tools**:
تحتوي على أوامر مختلف التصانيف كمحرر القوائم ، ومعالج إضافة إجراء **Procedure** وغيرها.

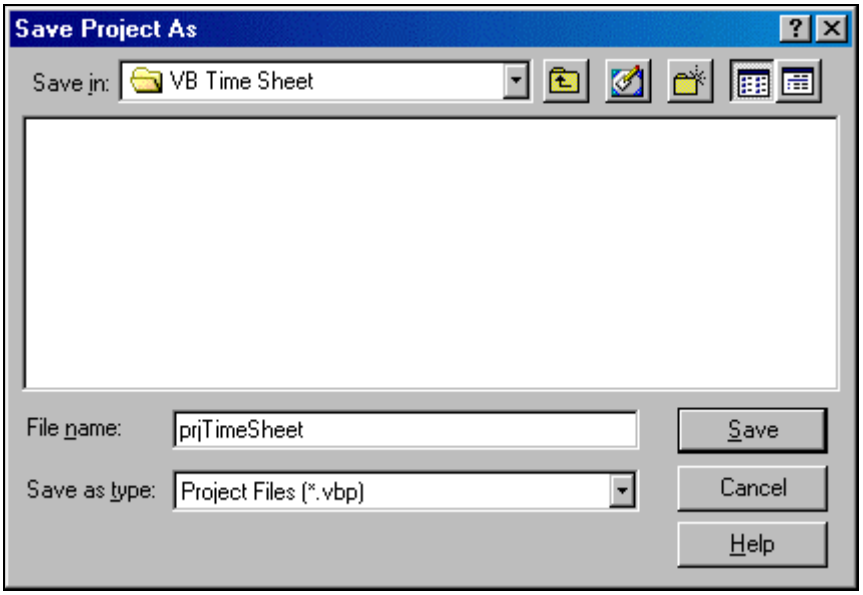
٩. القائمة **Add Ins**:

ومن خلالها يتم تشغيل برامج خارج اللغة وظيفتها خلق تكامل مع اللغة مثل برنامج Visual Data Manager وغيرها.

١٠. القائمة Windows:
ومن خلالها يتم وضع النوافذ بالشكل المطلوب ، وكذلك لعرض النافذة المطلوبة.

١١. القائمة Help:
ومن خلالها يتم استدعاء ملف التعليمات الخاص باللغة ، وكذلك التعريف باللغة ولمن هي مرخصة.

حفظ المشروع ومكوناته:
يفضل تخزين (حفظ) المشروع بما يحتويه من نوافذ ووحدات نمطية وغيرها وذلك باختيار الأمر Save Project من القائمة File فيظهر مربع حوار يحث المبرمج على تحديد مسار ملف المشروع وملفات النوافذ والوحدات النمطية كما هو موضح بالصورة التالية:



والآن إلى الجزء الثاني

الفصل الثاني : النافذة والأدوات الأساسية
The Form window and basic controls
النافذة The Form:
تعتبر النافذة بمثابة القلب النابض والهيكل الأساسي المكون لكل برنامج تطبيقي يعمل على منصة وندوز ، بحيث نضع عليها الأدوات والعناصر المختلفة بأسلوب منظم ومدروس لتشكل ما يسمى بواجهة المستخدم User Interface.
والصورة التالية توضح نافذة ومكوناتها:

واجهة المستخدم User interface:

وهي ما يقابل المستخدم من نوافذ يمكن أن يتعامل معها لأداء أعماله وما يطلبه من البرنامج. هذه الواجهة يجب أن تخضع لمجموعة من الشروط كي تحقق ما يطلق عليه: البرنامج المثالي الذي يجد فيه المستخدم السهولة والانسيابية والبساطة في التعامل معه.

شروط البرنامج الجيد:

لا بد للمبرمج من أن ينتبه إلى عدة شروط لكي يكون برنامجه جيداً مقبولاً لدى المستخدم ومنها:

أن يتقيد قدر الإمكان بالثوابت التي تتوفر في نظام وندوز من حيث السهولة والتلقائية وتسلسل الأحداث والأوامر وخضوعها لخوارزمية عمل محددة وصحيحة ، حيث أن المستخدم يتوقع أن يكون البرنامج مشابهاً في سلوكه لسلوك نظام التشغيل.

كما يجب أن يتوفر في البرنامج الواجهات الرسومية المتناسقة والفعالة حيث أن اللغة توفر أشكالاً هندسية (لأعمال الديكور) يمكن الاستفادة منها ، كذلك استخدام الرسومات والصور والأيقونات كرموز للعمليات المختلفة ، فالصورة - كما يقولون - بألف كلمة ، لكن يجب الانتباه إلى نقطة مهمة وهي أن نستخدم الرمز الواحد لعملية واحدة فقط فلا نستعمل الرمز ليرمز إلى أكثر من عملية مما يسبب ارتباكاً لدى المستخدم.

كما يجب الانتباه عند ترتيب العناصر على النافذة أن يكون العناصر المرتبطة منطقياً في عملها موضوعة في مجموعة واحدة أو جهة واحدة.

وأن يكون نمط واتجاه الكتابة عربيا (من اليمين إلى اليسار) في الواجهات العربية.

وكذلك ترتيب عناصر القوائم بصورة منطقية بناء على طريقة عملها وأن تكون القوائم شبيهة بقوائم وندوز قدر الإمكان.

كما يمكننا استعمال الألوان والخطوط Fonts وغير ذلك من أدوات تزيين واجهة البرنامج لكن يفضل ترك الألوان كما حددتها مايكروسوفت لأن هذه الألوان مريحة للعين ولا تسبب تشويشاً للمستخدم.

كذلك تجنب الإكثار من وضع العناصر والأدوات البرمجية على النافذة مما يسبب في بطء تحميلها.

كما يجب تجنب الاعتماد على الموقت Timer بكثرة ، مما يسبب أيضاً في بطء تحميل النافذة أو إظهار نتائج غير متوقعة.

لاحظ أن الإفراط في تزيين الواجهات يسبب نفور المستخدم ، فما زاد عن حده انقلب إلى ضده.

التعامل مع النافذة Working with Forms:

نتعامل مع النافذة عن طريق خصائصها Properties ووظائفها Methods وأحداثها Events.

خصائص النافذة Form Properties:

بعد إنشاء نافذة وتنسيقها بحيث تلائم متطلباتك ، نضبط خصائصها Sitting Properties لتحديد سلوكها وطريقة عملها ، وفيما يلي شرح موجز لأهم خصائصها:

الخاصية	عملها	الصورة العامة لها
Name	تحدد اسم النافذة البرمجي.	Obj.Name = String
Apearance	تظهر النافذة ثلاثية الأبعاد أو مسطحة.	Obj.Appearnace = [Value]
ملاحظة: لهذه الخاصية احتمالان ، انظر جدول ثوابت هذه الخاصية لاحقاً.		
BackColor	تحدد لون خلفية النافذة.	Obj.BackColor [= Color]
ملاحظة: لهذه الخاصية احتمالات كثيرة ، انظر جدول ثوابت الألوان لاحقاً (صفحة.....)		

BorderStyle	تحديد نوع إطار النافذة.	Obj.BorderStyle = Value
ملاحظة: لهذه الخاصية احتمالان ، انظر جدول ثوابت هذه الخاصية لاحقاً.		
ControlBox	إظهار وإخفاء أدوات التحكم.	Obj.ControlBox = Boolean
Enabled	تحدد هل النافذة ممكّنة أم غير ممكّنة.	Obj.Enabled = Boolean
Font	تحدد نوع وحجم خط الكتابة.	Obj.Font
ForeColor	لتحديد لون الكتابة أو الرسومات.	Obj.ForeColor = Color
MaxButton	تمكين الزر (تكبير) أو عدم تمكينه.	Obj.MaxButton = Boolean
MinButton	تمكين الزر (تصغير) أو عدم تمكينه.	Obj.MinButton = Boolean
Moveable	لجعل النافذة قابلة للتحريك أو جامدة.	Obj.Moveable = boolean
Picture	لتحميل ملف صورة على النافذة.	Obj.Picture [=Picture]

Name : لتحديد اسم النافذة البرمجي.
Appearance : لتحديد ظهورها بالشكل ثلاثي الأبعاد ٣D أو بالشكل المسطح Flat.
BackColor: لتغيير لون خلفيتها.
BorderStyle : لجعلها قابلة للتحجيم (تغيير حجمها وقت التشغيل) أو غير قابلة للتحجيم.
ControlBox : لإظهار إخفاء أدوات التحكم (تكبير واستعادة – تصغير – إغلاق).
Enabled : تحديد كون النافذة ممكّنة أو غير ممكّنة.
Font : لتحديد نوع وحجم خط الكتابة.
ForeColor : لتحديد لون الكتابة أو الرسومات.
MaxButton : تمكين الزر (تكبير) أو عدم تمكينه.
MinButton : تمكين الزر (تصغير) أو عدم تمكينه.
Movable : لجعل النافذة قابلة للتحريك أو جامدة.
Picture : وضع الصور عليها.
RightToLeft : لجعل نمط واتجاه الكتابة عربياً أو لاتينياً.
StartUpPosition : تحديد موقعها عند بدء التشغيل.
Visible : تحديد ظهور النافذة أو عدم ظهورها.
WindowState : تحديد حالتها عند التشغيل (مكبرة – عادية – مصغرة إلى شريط المهام).

وظائف النافذة Form Methods:
من الوظائف التي تدعمها النافذة:
الوظيفة Load: عند استدعائها يتم تحميل النافذة إلى الذاكرة دون إظهاره. والصيغة العامة لها:
Load FormName
حيث : Formname هو اسم النافذة البرمجي.

الوظيفة Unload: عند استدعائها يتم إنهاء تحميل النافذة من الذاكرة ، والصيغة العامة لها هي:
Unload FormName

الوظيفة Show: عند استدعائها يتم إظهار النافذة المحملة بالذاكرة عن طريق الوظيفة Load ، والصيغة العامة لها:
Object.Show [0 or 1]
حيث :
Object: يمثل الاسم البرمجي للنافذة.
[١ , ٠] : قيم اختيارية (تلقائيا = ٠) ، إذا كانت (١) يتم إظهار النافذة بحيث لا يمكن الرجوع إلى النافذة السابقة إلا بإخفائها. أما (٠) فيمكن الرجوع إلى النافذة السابقة حتى ولو لم يتم إغلاق النافذة الحالية.

الوظيفة Hide: عند استدعائها يتم إخفاء النافذة دون إنهاء تحميلها (أي تبقى في الذاكرة). لذا يجب أن تكون النافذة قبل استدعاء هذه الوظيفة محملة في الذاكرة وإلا فستظهر رسالة خطأ.
الصيغة العامة لهذه الوظيفة:
Object.Hide
حيث:
Object: الاسم البرمجي للنافذة.

الوظيفة PopUpMenu: وتستعمل عند الرغبة في إظهار قائمة Menu بصورة منبثقة. سيأتي شرحها عند التعامل مع القوائم.

الوظيفة PrintForm: ومن خلالها يتم طباعة النافذة وما تحتويه من عناصر وأدوات. والصورة العامة لها:
Object.PrintForm

الوظيفة Refresh: تعيد رسم النافذة والعناصر التي عليها (إعادة تنشيط النافذة) ، وفي العادة لا يلزمنا إعادة تنشيط النافذة برمجياً لأن فجول بيسك تقوم بذلك تلقائياً.
الشكل العام لها:
Object.Refresh
كما يوجد الكثير من الوظائف الأخرى.
ملاحظة:
يمكن استخدام الكلمة me لتمثيل اسم النافذة الحالية. فمثلاً لو كتبنا Unload me فيتم إنهاء تحميل النافذة الحالة.

والمخطط التالي يوضح ميكانيكية تنفيذ وظائف النافذة.

الوظيفة	Hard Drive	Memory	Screen	مثـــــال
Show				Me.Show frmMyForm.Show
Hide				Me.Hide frmMyForm.Hide
Load				Load frmMyForm
UnLoad				Unload Me Unload frmMyForm

أحداث النافذة Form Events:

مفهوم الحدث بصفة عامة:

إن الحدث بمفهومه العام كمصطلح في علم الحاسوب هو أي فعل يقوم به المستخدم كاستعمال لوحة المفاتيح للكتابة في عنصر معين مثلاً ، هنا: ضغط المفاتيح يعتبر حدثاً ، وللزيادة في الدقة ، إذا ضغط المستخدم مفتاحاً ولم يحرره (لم يرفع إصبعه من فوقه) يعتبر ذلك حدثاً يختلف عن الحدث الذي ينطلق عندما يرفع إصبعه من عليه ، وكذلك استعمال الفأرة يتولد عنه عدة أحداث ، فحركة الفأرة حدث ، والنقر بمؤشر الفأرة يطلق أحداثاً عدة.
وقد لا يكون المستخدم هو المسبب المباشر للحدث ، فمثلاً الأحداث التي تترتب عن الموقت **timer** تقع مرة كل فترة زمنية معينة.

وفيما يلي بعض الأحداث:

حدث التحميل Load Event:

يتم إطلاق هذا الحدث عند بدء تحميل النافذة في الذاكرة وظهورها للعيان ، يستفاد من هذا الحدث في عملية تعريف المتغيرات وتجهيز العناصر وغير ذلك.

الحدث Activate :

يتم إطلاق هذا الحدث عندما تكون النافذة هي النافذة النشطة.

حدث النقر المفرد Click والنقر المزدوج DblClick:

يتم إطلاق الحدث **Click** عند النقر بزر الفأرة على الكائن (نافذة أو عنصر تحكم) ، بينما يتم إطلاق الحدث **DblClick** عند النقر المزدوج السريع على الكائن.
قد يحدث إطلاق للحدث **Click** حتى بدون النقر الفعلي كتغيير قيمة الخاصية **Value** للعنصرين **Check box** و **Option box** من خلال الكود تماماً كما لو أن المستخدم نقر عليها بالزر الأيسر للفأرة ، ليس هذا فحسب ، بل يتم إطلاق الحدث **Click** عندما تتغير الخاصية **Listindex** للعنصرين **List box** و **Combo box** أيضاً.
هذان الحدثان **Click** و **DblClick** لا يمرران قيمة للبرنامج بحيث لا تعرف أين موقع المؤشر مثلاً ، فللحصول على مثل هذه المعلومات نستخدم الحدث **MouseDown** الذي سيأتي شرحه قريباً.

تنبيه / عندما تنقر على الكائن نقرتين سريعتين يتم إطلاق الحدثين **Click** ثم **Dblclick** معاً ، وهذا يسبب إرباكاً للبرنامج فلا يفرق بين النقر المفرد والمزدوج بسهولة لأن فجول بيسك سوف ينفذ الإجراء الحدثي **Click** ولا نعلم هل سينفذ إجراء الحدث **DblClick** أم لا. وعلى كل حال تجنب برمجة الإجرائين معاً على نفس الكائن.

الأحداث KeyPress ، KeyDown ، KeyUp:

تنطلق هذه الأحداث بالتسلسل التالي:
الحدث **KeyDown**: المستخدم يضغط المفتاح.
الحدث **KeyPress**: فجول بيسك يحول (يترجم) المفتاح إلى شفرة آنسي.
الحدث **KeyUp**: المستخدم يحرر (يرفع إصبعه عن) المفتاح.

يتم كل ذلك عندما يكون العنصر أو الأداة في بؤرة التحكم(١).

في حالة ضغط أحد المفاتيح (Ctrl + X) أو Space Bar أو Enter أو Escape أو أي مفتاح من مفاتيح حروف الطباعة يتم إطلاق الحدث KeyPress. أما في حالة باقي المفاتيح بما فيها مفاتيح الأسهم ومفاتيح الوظائف والمفتاحين (Alt + X) فيتم إطلاق الحدثين KeyDown و KeyUp.

إن الحدث KeyPress هو أبسط هذه الأحداث الثلاثة ، فهو يمرر شفرة Ansi للمفتاح المضغوط من قبل المستخدم ، فلتحديد هذا المفتاح برمجياً وجب علينا تحويله إلى سلسلة رموز بدلاً من شفرة آنسي باستخدام الدالة Chr$( ) كما في المثال التالي الذي يعرض اسم المفتاح الذي تم ضغطه:

```
Private Sub Text1_KeyPress(KeyAscii as integer)
Msgbox "User Pressed:" & Chr$(KeyAscii)
End Sub
```

أما الحدثان KeyDown و KeyUp فإنهما يمرران معاملين (بارامترين) هما KeyCode وShift ، حيث يشير الأول إلى المفتاح المضغوط والثاني يحدد ما إذا كان أحد الأزرار الثلاثة (Shift – Ctrl – Alt) قد تم ضغطه أيضاً.

الأحداث MouseDown و MouseUp و MouseMove:

ينطلق الحدث الأول عند النقر على العنصر دون تحرير زر الفأرة.

ينطلق الحدث الثاني عند تحرير زر الفأرة بعد النقر على العنصر.

ينطلق الحدث الثالث عند مرور مؤشر الفأرة فوق العنصر.

جميع هذه الأحداث لها نفس المعاملات :

أي زر من أزرار الفأرة (الأيسر – الأيمن) تم النقر به.

حالة المفاتيح (Shift – Ctrl – Alt) هل تم ضغطها أثناء النقر أم لا.

الإحداثيات السينية X والصادية Y لمؤشر الفأرة.

يبدأ العد للإحداثيات السينية والصادية من الركن الأيسر العلوي لكل عنصر أو أداة.

عند كتابة برمجة كل حدث من هذه الأحداث يجب الانتباه إلى النقاط التالية:

إن قيم X وY تمثل موقع مؤشر الفأرة بالنسبة للعنصر أو النافذة ، ولا تدخل حدود العنصر أو النافذة ضمن قيمها ، وتكون الزاوية العليا اليسرى للعنصر أو النافذة هي نقطة الأصل (٠،٠) ، وعند تحريك مؤشر الفأرة خارج العنصر أو النافذة من جهة اليسار أو الأعلى ستكون القيم سالبة.

عندما يضغط المستخدم زر الفأرة دون أن يحرره ثم يحرك مؤشر الفأرة خارج حدود العنصر أو النافذة فإن الأحداث الثلاثة لا تزال تعمل ومن المحتمل أن تكون قيم X و Y سالبة مما قد يسبب بعض المشاكل في البرنامج ، ولن يتم تنفيذ برمجة الحدث إلا بعد تحرير زر الفأرة فوق العنصر تماماً.

إن الحدثين MouseDown و MouseUp ينطلقان عندما يقوم المستخدم بضغط أو تحرير زر من أزرار الفأرة فوق العنصر. فعلى سبيل المثال: إذا ضغط المستخدم الزر الأيسر ولم يحرره ثم ضغط الزر الأيمن فإن العنصر يستقبل حدثين من MouseDown وبالتالي حدثين من MouseUp.

عندما يحرر المستخدم زر الفأرة يتم إطلاق الحدث Mouseup ومن ثم إطلاق الحدث MouseMove حتى ولو لم يتحرك مؤشر الفأرة.

العناصر البرمجية (أدوات التحكم)

Basic Controls

١ - زر الأوامر CommandButton:

ويستخدم لبدء أو مقاطعة أو إنهاء عملية معينة.

أهم خصائص زر الأوامر:

Appearance : لعرضه على صورة ثلاثية الأبعاد أو الصورة المسطحة.

Cancel: وتقوم بتنفيذ برمجة الحدث click التابع لهذا العنصر عند ضغط مفتاح الهروب Escهناك زر أوامر واحد فقط على النافذة تكون قيمة الخاصية Cancel له بـ True.

!! ž!CommandButton!! !!! ! !!! ! !! ! !!!! !! ž!!!! !!! ! !! ¿!!!!! ! !!!! ! !Š!!! !!! !! ž!! ! !Ÿ!!! !!!! !!! ! !!!!š!!Œ! ! º !!!Î !
!!! ! !!! ž!Text Box ! !! !!!!! !!!! !! !!!! !!!!!!! !! !!!!! !!!! !!!!! !! !!¿!! !! !!¿!! !! !! !! !!!!!! !!! !! !! !!!!!! !!!! !!!!!! ! !
!! ! !! !!!! !!!! !!!!! !! !!!!! !! ! !! ! !!!! !!!!! ! !!! ž!! ! !Ÿ!!! ! !!!!!!!!Ÿ!!! ž!!!!! !!!!!!! !!!! ! !!!! ! »!!!!!!» ! !!!! !»!!
!Tabindex!!!! !! !!!! !!!! !!! !!!! Š! ! !!!!! !! !

Caption: تمثل النص الظاهر على هذا العنصر. وتستخدم كذلك لإنشاء مفتاح الاختصار Short Cut ، وهو المفتاح الذي يضغط مع المفتاح Alt لتنفيذ برمجة حدث النقر المفرد لزر الأوامر.
Default: وتقوم بتنفيذ برمجة الحدث click التابع لهذا العنصر عند ضغط مفتاح الإدخالEnter. هناك زر أوامر واحد فقط على النافذة تكون قيمة الخاصية Default له بـ True.
Font: ومن خلالها يتم ضبط نوع وحجم الخط الذي يتم به كتابة النص Caption.
ToolTipText: وتعرض نصاً في مربع صغير على هيئة تلميح ، يظهر عند توقف مؤشر الفأرة على العنصر.

أهم أحداث زر الأوامر:
Click: يتم إطلاق هذا الحدث عند النقر على زر الأوامر.

إنشاء مفتاح الاختصار Short Cut:
تتميز العناصر التي تدعم الخاصية Caption عن تلك التي تدعم الخاصية Text في أنها يمكن أن نسند لها مفتاح اختصار يتكون من المفتاح Alt مضافاً إليه أحد الحروف المكتوبة في الخاصية Caption ، وتتم بكتابة الرمز (&) قبل الحرف الذي سيكوّن مفتاح الاختصار.
فمثلاً عند كتابة (&Open) في الخاصية Caption لزر الأوامر ستظهر الكلمة كالتالي (Open) وعند الضغط بلوحة المفاتيح على كل من (Alt + O) سيتم تنفيذ برمجة الحدث Click لهذا الزر وهكذا.

٢ - عنصر التسمية Label:
ويستخدم لعرض النصوص التي لا يمكن للمستخدم تعديلها بصور مباشرة ، وكذلك يستعمل كعنوان يوضح الغرض من عنصر آخر يكون بجانبه ، أو يستخدم لإظهار نتيجة عملية ما.

أهم خصائص عنصر التسمية :
Alignment: (المحاذاة) : محاذاة النص الظاهر عليه إما إلى اليمين أو إلى اليسار أو في منتصف العنصر.
Appearance: وتحدد ظهور العنصر إما بالصورة ثلاثية الأبعاد أو بالصورة المسطحة.
AutoSize: إذا كانت True يتم جعل حجم العنصر مطابقاً للكتابة ، أما إذا كان False فيتم جعل حجم العنصر كما حدده المستخدم تماماً(١).
BorderStyle: يحدد نوع حد العنصر (بارز – عادي).
Caption: يحدد النص الظاهر على العنصر.
Font: ومن خلالها يتم ضبط نوع وحجم الخط الذي يتم به كتابة النص Caption.
ToolTipText: وتعرض نصاً في مربع صغير على هيئة تلميح ، يظهر عند توقف مؤشر الفأرة على العنصر.

أهم أحداث عنصر التسمية :
Click: وينطلق عند النقر المفرد عليه.
DblClick: وينطلق عند النقر المزدوج عليه.
هذه الأداة تدعم المعامل (&) في الخاصية Caption.

صندوق النصوص TextBox:
ويستخدم لإدخال القيم (على هيئة سلسلة رموز) ، وكذلك لعرض نصوص قابلة للتعديل بصورة مباشرة.

أهم خصائص صندوق النصوص:
Appearance: وتحدد ظهور العنصر إما بالصورة ثلاثية الأبعاد وإما بالصورة المسطحة.
BorderStyle: يحدد نوع حد العنصر (بارز – عادي).
Font: ومن خلالها يتم ضبط نوع وحجم الخط الذي يتم به كتابة النص في الخاصية text.
MaxLength: ومن خلالها يتم تحديد عدد الحروف التي يستوعبها العنصر.
Multiline: وتحدد ما إذا كان العنصر متعدد الأسطر أم بسطر واحد.
PasswordChar: وتستعمل في حالة إدخال كلمات المرور للبرنامج بحيث يظهر حرف ما بدلاً عن الكتابة وفي المعتاد تستخدم النجمة (*).
ScrollBars: تحدد هل ستظهر أشرطة التمرير أم لا.
SelStart: تحدد مكان البدء في تحديد النص Selecting (لغرض نسخه أو قصه أو لصق نص آخر بدلاً منه). هذه الخاصية متوفرة فقط وقت التشغيل (بالكود).
SelText: تمثل النص المحدد (وقت التشغيل).

!!!!!!!!! Ÿ!!!!!! !!! !!!ƒ!!!!!!!!! !Ÿ!!!Š!!!!!!!!!!!!!!!!!!!!Š!!!!!!!!!!!!!!!!!!!!!!!!!ž!!Ï!

Text: من خلالها يتم إدخال النصوص وعرضها.
ToolTipText: وتعرض نصاً في مربع صغير على هيئة تلميح ، يظهر عند توقف مؤشر الفأرة على العنصر.

أهم أحداث صندوق النصوص:
Change: ينطلق في كل لحظة يتم فيها تغيير قيمة الخاصية **Text**.
LostFocus: وينطلق عندما يغادر المستخدم صندوق النصوص إلى عنصر تحكم آخر.
KeyPress: ينطلق كلما تم الضغط على أحد المفاتيح.
أهم وظائف صندوق النصوص:
SetFocus: ومن خلالها يتم وضع صندوق النصوص في بؤرة التحكم.

الأداة **CheckBox** (مربع الاختيار ١):
ويسمح للمستخدم باختيار بين حالتين نعم أو لا.

أهم خصائص مربع الاختيار:
Caption: وتمثل النص المعروض على الأداء.
Font: تحدد نوع وحجم الخط الذي يتم به كتابة النص في الخاصية **Caption**.
Value: وتمثل حالة الأداة (إذا كانت مساوية للصفر تعني عدم ظهور العلامة (صح) ، وإذا كانت مساوية للواحد تعني ظهور العلامة (صح) ، وإذا كانت مساوية ٢ فتظهر الداء بلون هافت (رمادي)).

أهم أحداث مربع الاختيار:
Click: وينطلق عند النقر المفرد عليه.

الأداة **Option Box** (مربع الاختيار ٢):
وتمكن المستخدم من انتقاء خيار واحد فقط من مجموعة اختيارات.

أهم خصائص مربع الاختيار ٢:
Caption: وتمثل النص المعروض على الأداء.
Font: تحدد نوع وحجم الخط الذي يتم به كتابة النص في الخاصية **Caption**.
Value: وتحدد ما إذا كانت الأداة مختارة **Selected** (أي ظهور النقطة السوداء على الأداة) أم لا. هناك أداة اختيار واحدة فقط يمكن أن تظهر النقطة عليها.

أهم أحداث مربع الاختيار ٢:
Click: وينطلق عند النقر المفرد عليه.

الأداة **Frame** (الإطار):
تستخدم هذه الأداة لتزيين الواجهة ، وكذلك لضم العناصر المرتبطة منطقياً أو وظيفياً في مجموعة واحدة.
تعرف هذه الأداة بالحاوية **container**.

أهم خصائص الإطار:
Caption: يعرض النص أعلى الإطار.
Font: تحدد نوع وحجم الخط الذي يتم به كتابة النص في الخاصية **Caption**.

الإطار كحاوية **Frame as container**:
عندما يكون الإطار حاوية ، ما ينطبق على الإطار ينطبق بالنتيجة على العناصر التي يضمها ، فإن أخفينا الإطار مثلاً باستخدام الخاصية **Visible = False** يتم إخفاء العناصر التي يحتويها.

لضم العناصر في الإطار:
نضع الإطار على النافذة ونحدد أبعاده إلى الحجم المطلوب.
ننقر على العنصر نقرة واحدة في شريط الأدوات ثم نرسمه رسماً على الإطار (وليس بالنقر المزدوج على الأداة لوضعها).

ملاحظات:

عند رسم الأداة خارج الإطار ثم سحبها إليه لا يضم الأداة إلى الإطار ، ولا يكون الإطار هو الحاوية التي تضم الأداة ، بل تكون النافذة هي الحاوية.
عند وضع مجموعة من مربعات الاختيار ٢ في إطار ، ومجموعة أخرى في إطار آخر ، تعمل كل مجموعة لوحدها ، أي يمكن اختيار أحد المربعات في الإطار الأول واختيار آخر من الإطار الثاني. في حين لو وضعنا مربعات الاختيار ٢ على النافذة مباشرة يتم اختيار أحدها فقط.

الأداة **ListBox**:
هذه الأداة تعرض قائمة بالاختيارات بحيث يمكن للمستخدم من انتقاء اختيار واحد أو أكثر ، إذا كان عدد الاختيارات أكبر من ارتفاع الأداة تختفي بعض الاختيارات ويظهر شريط تمرير للوصول إلى باقي الاختيارات.

أهم خصائص الأداة **ListBox**:
Appearance: وتحدد ظهور العنصر إما بالصورة ثلاثية الأبعاد وإما بالصورة المسطحة.
List: ومن خلالها يتم كتابة الاختيارات التي تظهر عليها.
ListCount: وتعطي عدد الاختيارات التي تضمها.
ListIndex: وتعطي رقم ترتيب الاختيار في الأداة.
Multiselect: وتحدد هل يمكن للمستخدم من انتقاء أكثر من خيار أم لا.
Selected: هي مجموعة من القيم (**True** أو **False**) تحدد الاختيارات التي تم انتقاؤها.
Sorted: هل سيتم ترتيب العناصر حسب جدول **ASCII** أم تبقى كما حددها المستخدم.
Text: وتمثل الخيار النشط.

أهم أحداث الأداة **ListBox**:
Click: ينطلق عند النقر المفرد على الأداة.
DblClick: ينطلق عند النقر المزدوج على الأداة.

أهم وظائف الأداة **ListBox**:
AddItem: لإضافة اختيار جديد للأداة برمجياً.
Clear: محو كافة الخيارات.
DeleteItem: محو الخيار المطلوب.

ملاحظة:
يفضل تعبئة الأداة بالاختيارات في الحدث **Form_Load**.

الأداة **ComboBox**:
هذه الأداة شبيهة بالأداة السابقة إلا أنها تختلف عنها في أن الأولى تعرض أكثر من اختيار في نفس الوقت ، أما هذه الأداة فتعرض اختياراً واحداً فقط ، تتميز هذه الأداة عن الأولى بإمكانية الكتابة فيها مباشرة.

أهم خصائص الأداة **ComboBox**:
Appearance: وتحدد ظهور العنصر إما بالصورة ثلاثية الأبعاد وإما بالصورة المسطحة.
List: ومن خلالها يتم كتابة الاختيارات التي تظهر عليها.
ListCount: وتعطي عدد الاختيارات التي تضمها.
ListIndex: وتعطي رقم ترتيب الاختيار في الأداة.
Sorted: هل سيتم ترتيب العناصر حسب جدول **ASCII** أم تبقى كما حددها المستخدم.
Style: وتحدد شكل ظهور الأداة وهل يمكن الكتابة فيها أم لا.
Text: وتمثل الخيار المختار ويتم عرضه أعلى الأداة.

أهم أحداث الأداة **ComboBox**:
Click: ينطلق عند النقر المفرد على الأداة.
DblClick: ينطلق عند النقر المزدوج على الأداة.

أهم وظائف الأداة **ComboBox**:
AddItem: لإضافة اختيار جديد للأداة برمجياً.
Clear: محو كافة الخيارات.
DeleteItem: محو الخيار المحدد برقم فهرسه ، ويبدأ الفهرس من القيمة (٠) والتي تمثل الخيار الأول في الأداة.

أمثلة:

CboExample.AddItem "This is an example"
CboExample.AddItem "هذا مثال"
يتم إضافة خيارين للأداة هما: this is an example و هذا مثال إلى الأداة.
CboExample.DeleteItem 1
يتم حذف الخيار الثاني وهو (هذا مثال).
CboExample.Clear
يتم حذف جميع الخيارات وتبقى الأداة فارغة. وهذه الأمثلة تنطبق على الأداة ListBox.

الأداة Line (خط):
هذه الأداة ترسم خطاً مستقيماً على النافذة.

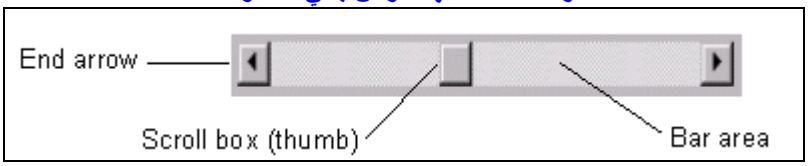
أهم خصائص الأداة Line:
BorderColor: وتحدد لون الخط.
BorderStyle: وتحدد الخط هل هو متصل أم متقطع أم غير ذلك.
BorderWidth: وتحدد سمك الخط.

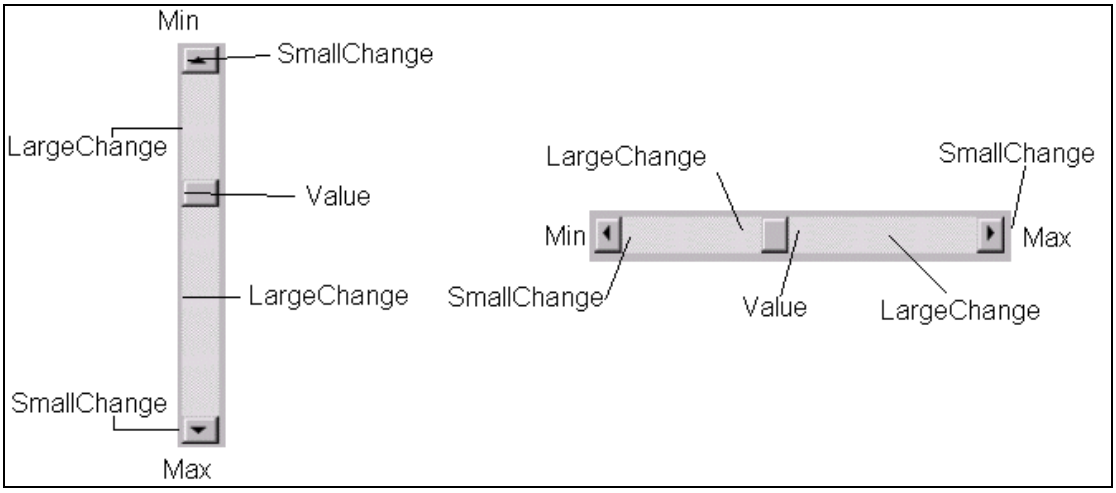
ليس هناك أحداث أو وظائف لهذه الأدوات.
الأداة Shape:
هذه الأداة ترسم مربعاً أو دائرة أو مستطيلاً.

أهم خصائص الأداة shape:
BackColor: وتحدد لون خلفية الأداة.
BackStyle: وتحدد ما إذا كانت خلفية الأداة شفافة أم مصمتة.
BorderColor: وتحدد لون حواف الأداة.
BorderStyle: وتحدد شكل حواف الأداة هل هي منقطة أم متصلة أم غير ذلك.
BorderWidth: وتحدد سمك حواف الأداة.
FillColor: وتحدد لون التعبئة.
FillStyle: وتحدد طريقة التلوين.
Shape: وتحد هل الأداة بشكل مربع أم دائرة أم مستطيل أم غير ذلك.

هذه الأداة ليس لها أحداث أو وظائف.

أشرطة التمرير الأفقية والرأسية:
تستعمل أشرطة التمرير بكثرة في نظام النوافذ windows. عندما تختفي بعض المعلومات في النافذة تظهر الأشرطة فنستخدمها لعرض باقي المعلومات.





أهم خصائص أشرطة التمرير:
LargeChange: وتغير قيمة الخاصية **value** بمقدار معين عندما ينقر المستخدم على **Bar Area**.
Max: وتحدد أقصى قيمة عددية صحيحة يمكن أن تصل إليها الخاصية **value**.
Min: وتحدد أقل قيمة عددية صحيحة أن تصلها الخاصية **Value**.
SmallChange: وتغير قيمة الخاصية **Value** بمقدار معين عندما ينقر المستخدم على أسهم شريط التمرير.
Value: وهي قيمة موقع زر التمرير **Scroll Box** على الشريط.

ملاحظة: إذا تم تغيير قيم الخصائص **max** و **Min** و **Value** برمجياً فيجب الانتباه إلى أنها تقع بين قيمتي الخاصيتين **Max** و **Min** ولا تتعداها.

أهم أحداث أشرطة التمرير:
Change: وينطلق بعد أن يتغير موقع مربع التمرير.
Scroll: وينطلق باستمرار عندما يتحرك مربع التمرير.

الأداة **PictureBox** (أداة عرض الصور):
تسمح هذه الأداة بعرض الصور عليها ، وهي الوسط المفضل لعرض الصور المتحركة **Animation**. وهي تشبه بحد كبير نموذج النافذة في الكثير من الخصائص ، وتعتبر حاوية لغيرها من العناصر البرمجية الأخرى.

أهم خصائص أداة الصور:
AotuSize: وتحدد هل سيكون حجم أداة الصور مطابقاً لحجم الصورة المعروضة أم أن حجم الصورة المعروضة مطابق لحجم الأداة.
Font: لضبط نوع وحجم الخط.
Picture: وفيها يتم تحميل ملف الصورة.

أهم أحداث أداة الصور:
Click: وينطلق عند النقر المفرد عليها.
DblClick: وينطلق عند النقر المزدوج على الأداة.

أهم وظائف أداة الصور:
Cls: لمسح محتويات الأداة.
Print: لطباعة المعلومات على الأداة.

ولكي نعرض الصورة على أداة الصور نستخدم الدالة **LoadPicture** وكما يلي:
PicExample.Picture = LoadPicture("مسار ملف الصورة")

ويمكن تحميل الملفات التي تنتمي للأنواع التالية فقط على أداة الصور: **Bmp** ، **Ico** ، **Wmf** ، **Jpeg** ، **Jpg**.

الأداة **Image** (أداة الصور ٢):

وهي أداة تشبه الأداة السابقة في أنها تعرض صورة على النافذة ، لكنها لا تتعامل مع الصور المتحركة ، وكذلك لا تصلح لأن تكون حاوية لغيرها من العناصر البرمجية الأخرى.

أهم خصائص الأداة:
Picture: لتحميل ملف الصورة.
Streatch: وتحدد هل سيكون حجم الصورة مطابقاً لحجم الأداة أم أن حجم الأداة سيكون مطابقاً لحجم الصورة.

أهم أحداث الأداة:
Click: وينطلق عند النقر المفرد على الأداة.
DblClick: وينطلق عند النقر المزدوج على الأداة.

ملاحظات:
هذه الأداة لا تدعم التعامل مع أي وظيفة **method**.
لكنها تتعامل مع الدالة **LoadPicture** ، بالإضافة إلى أنها تدعم تحميل ملفات الصور من الأنواع السابقة.

الأداة **DriveListBox** مستعرض السواقات:
هذه الأداة تتيح للمستخدم اختيار إحدى سواقات الأقراص الموجودة بجهاز الحاسوب وقت التنفيذ. هذه الأداة تعرض الحرف الذي يمثل السواقات في **Combo** ، ولا حاجة للكود في عملية تعبئة الأداة بالعناصر.

أهم خصائص مستعرض السواقات:
Drive: وتمثل حرف السواقة المختارة.

أهم أحداث مستعرض السواقات:
Change: وينطلق كلما غير المستخدم أو البرنامج الحرف الذي يمثل السواقة المختارة.

الأداة **DirListBox** مستعرض المجلدات:
هذه الأداة تعرض المجلدات بصورة هرمية منظمة ، وتتيح اختيار أحد المجلدات أو المجلدات الفرعية.
ولكي تحدد المجلدات التي ستظهر على الأداة نحتاج لبعض البرمجة التي تربط هذه الأداة بالأداة السابقة ، أي عرض المجلدات التي تتبع السواقة المختارة.

أهم خصائص مستعرض المجلدات:
Path: وتمثل مسار المجلد المختار.

أهم أحداث مستعرض المجلدات:
Change: وينطلق عند اختيار مجلد معين.

الأداة **FileListBox** مستعرض الملفات:
وتعرض الملفات بشكل هرمي منظم ، ولكي نعرض الملفات لا بد من ربط هذه الأداة بمستعرض المجلدات المربوط بمستعرض السواقات.

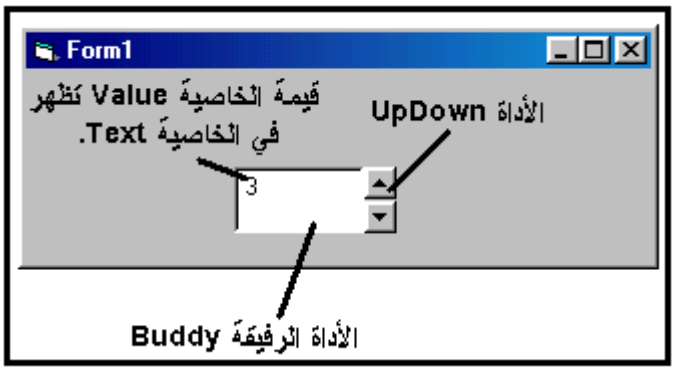
أهم خصائص مستعرض الملفات:
FeileNam: تمثل اسم الملف المختار.
MultiSelect: وتحدد هل يمكن اختيار أكثر من ملف أم لا.
Path: وتمثل مسار الملف الحالي.
Pattern: وهي سلسلة تحدد أنواع الملفات المطلوب عرضها في هذه الأداة ، فعلى سبيل المثال لو كتبنا **Dat.*** سيتم عرض أسماء الملفات التي امتدادها **Dat** فقط.
أهم أحداث مستعرض الملفات:
DblClick: وينطلق عند النقر المزدوج على اسم الملف.
PathChange: وينطلق عندما يتغير المسار.

عناصر وأدوات برمجية أخرى:

الأداة UpDown:

وتحتوي هذه الأداة على زرين للأسهم نستخدمها لزيادة أو إنقاص قيمة معينة يتم إظهارها على أداة أخرى ، قد تكون هذه الأداة صندوق نصوص أو أي أداة أخرى ، وفي هذه الحالة يطلق على الأداة المرتبطة بالأداة Updown بالأداة الرفيقة (Buddy).

والصورة التالية توضح الأداة UpDown مقترنة بالأداة الرفيقة (صندوق النصوص):



أهم خصائص الأداة UpDown:

الخاصية	عملها	الصورة العامة لها
Alignment	محاذاة UpDown بالنسبة للأداة الرفيقة	Obj.Alignment [= Value]
ملاحظة: لهذه الخاصية احتمالان ، انظر جدول ثوابت هذه الخاصية لاحقاً.		
BuddyControl	تمثل اسم الأداة الرفيقة البرمجي.	Obj.BuddyControl = [Value]
ملاحظة: يمكن أن تكون قيمتها = nothing وفي هذه الحالة ليس لـ UpDown أداة رفيقة.		
BuddyProperty	اسم الخاصية التي ستظهر عليها قيمة الأداة Updown.	Obj.BuddyProperty [= Value]
ملاحظة: في المعتاد أن تكون قيمتها مساوية لـ Text أو Caption أو حتى Value.		
Increment	مقدار زيادة أو نقصان قيمة الخاصية Value.	Obj.Increment [= Value]
ملاحظة: هذه الخاصية تقبل قيمة عددية صحيحة موجبة.		
Max	أقصى قيمة تصل إليها الخاصية Value	Obj.Max [= Value]
Min	أدنى قيمة تصل إليها الخاصية Value	Obj.Min [= Value]
ملاحظة: هاتان الخاصيتان تقبلان قيمة عددية صحيحة موجبة.		
Value	وتمثل القيمة الحالية للأداة Updown.	Obj.Value [=Number]
ملاحظة: هذه الخاصية تقبل قيمة عددية صحيحة موجبة تقع بين Min وMax ولا تتعداهما.		

الثوابت الرمزية للخاصية Alignment:

الثابت الرمزي	القيمة	الوصف
Cc2AlignmentLeft	٠	محاذاة النص إلى اليسار.
Cc2AlignmentRight	١	محاذاة النص إلى اليمين.

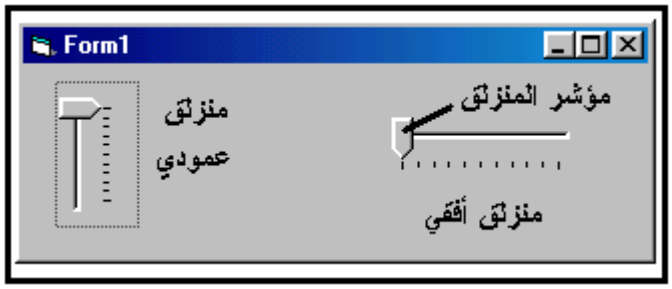
أهم أحداث الأداة Updown:

الحدث	وقت انطلاقه
Change	+
UpClick	وينطلق عند النقر فوق الزر العلوي للأداة.
DownClick	وينطلق كلما تم النقر فوق الزر السفلي للأداة.

الأداة Slider (المنزلق):

وهي عبارة عن شريط به زر (مؤشر) وتدريج (اختياري) ، يمكن من خلالها اختيار قيمة معينة بناء على موقع المؤشر على الشريط.

والصورة التالية توضح المنزلق بشكليه الأفقي والعمودي:



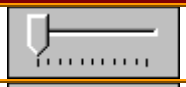
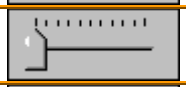
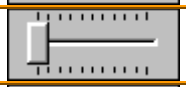
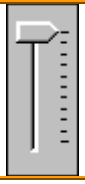
أهم خصائص المنزلق:

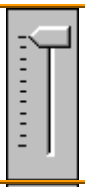
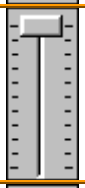
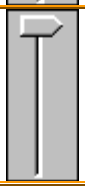
الخاصية	عملها	الصورة العامة لها
Max	وهي أقصى قيمة تصل إليها الخاصية Value.	Obj.Max [= Number]
Min	وهي أدنى قيمة تصل إليها الخاصية Value.	Obj.Min [= Number]
Value	وتمثل القيمة التي وصل إليها مؤشر الأداة.	Obj.Value [= Number]
ملاحظة: هذه الخاصية تقبل قيمة عددية صحيحة موجبة (تقع بين Min و Max) تمثل موقع مؤشر الأداة.		
Orientation	تحدد هل المنزلق أفقي أم عمودي.	Obj.Orientation = Number
ملاحظة: لهذه الخاصية احتمالان ، انظر جدول ثوابت هذه الخاصية لاحقاً.		
TickStyle	وتحدد شكل المنزلق وظهور التدريج أو إخفاؤه اعتماداً على الخاصية Orientation.	Obj.TickStyle [= number]

ثوابت الخاصية Orientation الرمزية:

الثابت الرمزي	القيمة	الوصف
SldHorizental	٠	(تلقائي) المنزلق أفقي.
SldVertical	١	المنزلق عمودي (رأسيّ).

ثوابت الخاصية TickStyle الرمزية بالتزامن مع قيم الخاصية Orientation:

الثابت الرمزي	القيمة	قيمة الخاصية Orientation	شكل المنزلق
SldButtomRight	٠	٠	
SldTopLeft	١	٠	
SldBoth	2	0	
SldNoTicks	3	0	
SldButtomRight	0	1	

	1	1	SldTopLeft
	1	2	SldBoth
	1	3	SldNoTicks

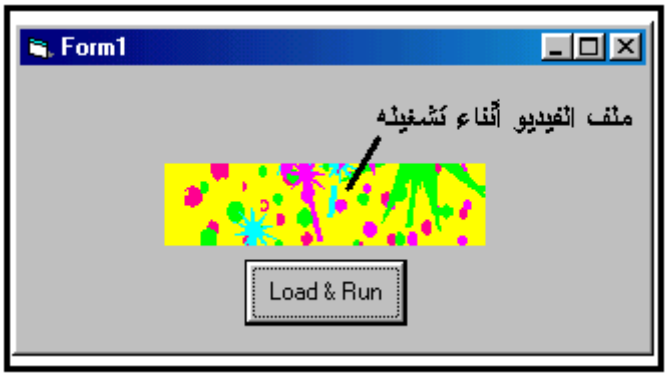
أهم أحداث المنزلق:

الحدث	وقت انطلاقه
Click	ينطلق عند النقر المفرد على المنزلق.
Scroll	وينطلق كلما تم الإمساك بزر المنزلق وسحبه.

الأداة Animation:

تستخدم هذه الأداة لتشغيل ملفات الفيديو من النوع (Avi). حيث يتم تحميل الملف برمجياً وليس من خلال نافذة الخصائص لأن أمر التحميل يعتبر وظيفة وليس خاصية.

والصورة توضح ملف فيديو أثناء تشغيله من خلال نقر الزر (Load & Run):



أهم خصائص الأداة Animation:

الخاصية	عملها	الصورة العامة لها
AutoPlay	هل سيتم تشغيل الملف بمجرد تحميله في الأداة أم لا.	Obj.AutoPlay[= Boolean]
Center	وتحدد مكان عرض الملف في الأداة.	Obj.Center [= Boolean]
ملاحظة: إذا كانت مساوية لـ True يظهر الفيديو بمنتصف الأداة ، وإلا سيظهر بركنها الأيسر العلوي.		

أهم أحداث الأداة Animation:

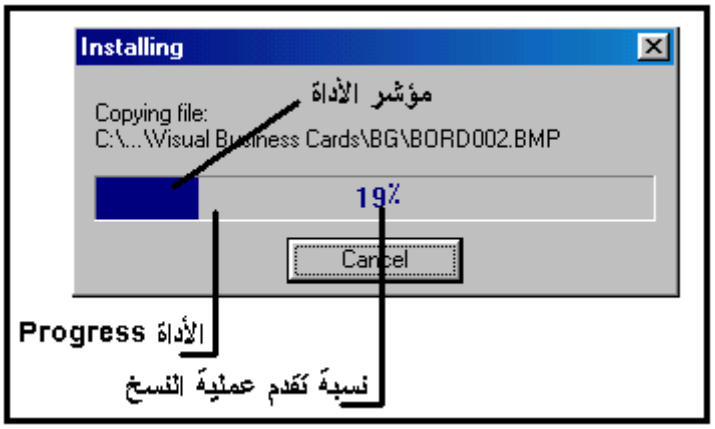
الحدث	وقت انطلاقه
Click	ينطلق عند النقر المفرد على المنزلق.
DblClick	وينطلق عند النقر المزدوج على الأداة.

أهم وظائف الأداة Animation:

الوظيفة	عملها	الصورة العامة لها
Open	ومن خلالها يتم تحميل ملف الـ Avi في الأداة.	Obj.Open (Path)
Play	ومن خلالها يتم تشغيل الملف.	Obj.Play
Stop	وتستخدم لغرض إيقاف تشغيل الملف.	Obj.Stop

الأداة Progress:

تستخدم هذه الأداة في حالة نسخ الملفات أو أثناء عمليات إنزال البرامج ، بحيث توضح للمستخدم تقدم عملية الإنزال أو النسخ بشريط أزرق يبدأ من اليسار وينتهي باليمين.
والصورة التالية توضح هذه الأداة وقت التشغيل:



أهم خصائص الأداة Progress:

الخاصية	الوظيفة	الصورة العامة لها
Value	وتمثل القيمة التي وصل إليها مؤشر الأداة.	Object.Value [= Number]
ملاحظة: هذه الخاصية تقبل قيمة عددية صحيحة موجبة تمثل موقع مؤشر الأداة.		
Max	وهي أقصى قيمة تصل إليها الخاصية Value.	Object.Max [= Number]
Min	وهي أدنى قيمة تصل إليها الخاصية Value.	Object.Min [= Number]

أهم أحداث الأداة Progress:

الحدث	وقت انطلاقه
Click	ينطلق عند النقر المفرد على الأداة.

ملاحظة : توصي شركة مايكروسوفت أن يكون اتساع Width الأداة يكافئ ارتفاعها Height بمقدار اثني عشر مرة (١٢ مرة).

الأداة StatusBar شريط الحالة:

وتستخدم في توضيح حالة البرنامج أو بعض المفاتيح الخاصة مثل CAPS Lock وIns و Num Lock و Scroll أو لعرض تعليمات معينة أو التاريخ والوقت وغير ذلك.
صورة الأداة أثناء التشغيل

أهم خصائص شريط الحالة:

الخاصية	عملـــها	الصورة العامة لها
Align	وتحدد مكان محاذاة الشريط بالنسبة للنافذة.	Obj.Alignment [= Number]
ملاحظة: لهذه الخاصية خمس احتمالات ، انظر جدول ثوابت هذه الخاصية لاحقاً.		
Style	وتحدد شكل الشريط: (عادي أم بسيط).	Obj.Style [= Number]
ملاحظة: لهذه الخاصية احتمالان ، انظر جدول ثوابت هذه الخاصية لاحقاً.		
SimpleText	وتمثل النص الذي يظهر على الشريط.	Obj.SimpleText [= String]
ملاحظة: تعمل هذه الخاصية إذا كانت الخاصية Style = True.		
ShowTips	تتحكم في ظهور التلميحات في أشرطة الأدوات.	Obj.ShowTips [= Boolean]
Font	لتحديد نوع وحجم الكتابة في قطاعات الشريط.	Obj.Font

الثوابت الرمزية للخاصية Align:

الثابت الرمزي	القيمة	الوصف
VbAlignNone	٠	كما حددها المبرمج.
VbAlignTop	١	الشريط أعلى النافذة.
VbAlignButtom	٢	الشريط أسفل النافذة.
VbAlignLeft	3	الشريط يمين النافذة.
VbAlignRight	4	الشريط يسار النافذة
ملاحظة : في جميع الحالات يكون اتساع الشريط مساوياً للخاصية ScaleWidth للنافذة.		

الثوابت الرمزية للخاصية Style:

الثابت الرمزي	القيمة	الوصف
SbrNormal	٠	تظهر جميع قطاعات الشريط.
SbrSimple	١	يظهر قطاع واحد بحجم الشريط.

القطاعات Panels التي يحتوي عليها الشريط:
يحتوي شريط الحالة على (تجمع Collection) يسمى (القطاعات Panels) ، هذه القطاعات لها خصائص ووظائف إليك شرحها بالتفصيل:

أهم خصائص الكائن Panels التابع لشريط الحالة:

الخاصية	عملـــها	الصورة العامة لها
Count	وترجع عدد القطاعات التي يتضمنها الشريط.	Obj.Count
Item	ترجع العضو المحدد من خلال قيمة فهرسه أو موضعه.	Obj.Item(index)

أهم وظائف الكائن Panels التابع لشريط الأدوات:

الوظيفة	عملـــها	الصورة العامة لها
Add	تستخدم لإضافة قطاع جديد للشريط.	ObjName.Panels.Add _ ([index],[Key],[Text],[Style],[Picture])
ملاحظة: جميع هذه البارامترات اختيارية ، وفي حالة تجاهلها كلها يُضاف قطاع فارغ.		
Remove	تستخدم لإزالة قطاع من الشريط.	ObjName.Panels.Remove (index)
ملاحظة: يتم إزالة قطاع معين بواسطة رقم فهرسه ، علماً بأن أول قطاع يحمل فهرساً رقمه (٠).		
Clear	تستخدم لإزالة جميع قطاعات الشريط.	ObjName.Panels.Clear

أهم أحداث شريط الحالة:

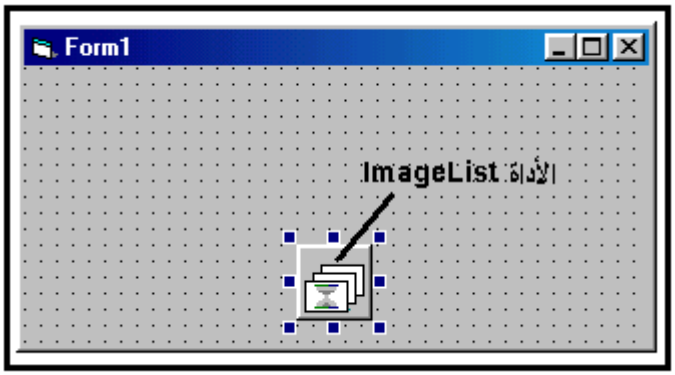
الحـــدث	وقت انطلاقه
PanelClick	ينطلق عند النقر المفرد على أحد قطاعات شريط الحالة.
PanelDblClick	ينطلق عند النقر المزدوج على أحد قطاعات شريط الحالة.
ملاحظة: كلا الحدثين يحتويان على بارامتر يحدد القطاع الذي تم نقره مفرداً أو مزدوجاً.	

أهم وظائف شريط الحالة:

الوظيفة	عملـها	الصورة العامة لها
Refresh	تنشيط الشريط في حالة تغيير نص أحد القطاعات.	StatusBarName.Refresh

الأداة ImageList:

هذه الأداة تستخدم كحاوية للصور والأيقونات التي تستخدم من قبل الأدوات الأخرى مثل TreeView و ListView و TabStrip و ImageCombo وغيرها من الأدوات الأخرى.
هذه الأداة غير مرئية وقت التشغيل ، ولا يتم التعامل معها إلا برمجياً من خلال خصائصها ووظائفها.
إن استعمال هذه الأداة كمستودع للصور التي تستعمل فيما بعد من قبل الأدوات الأخرى له فوائد جمة ، فلولاها لكابدت الأمرّين من استعمال الدالة LoadPicture التي تسبب البطء الشديد في تحميل الصور وخاصة الكبيرة منها ، وبالتالي البطء في تحميل النافذة. كما أنه من الأفضل أن يتم تحميل كل الصور في ImageList وقت التصميم ومن ثم الإشارة إليها من خلال العناصر والأدوات الأخرى من خلال نافذة الخصائص أو بالكود.
والصورة توضح الأداة وقت التصميم:



تحتوي هذه الأداة على كائنات مضمنة داخلها تسمى ListImages كل واحدة منها تمثل صورة (غالباً ما تكون الصورة من نوع Bmp – Ico – Cur – Jpg –Gif) ، بحيث نصل إلى الصورة المطلوبة من خلال مفتاح يسمى (Index).

أهم خصائص الكائن ListImage:

الخاصية	عملها	الصورة العامة لها
Count	وترجع عدد الصور التي تتضمنها الأداة.	Obj.Count
Item	ترجع العضو المحدد من خلال قيمة فهرسه أو موضعه.	Obj.Item(index)
MaskColor	تحديد اللون الذي سيكون شفافاً في الصورة المعروضة.	Obj.MaskColor = Color
Usemaskcolor	تحدد هل ستعمل الخاصية السابقة أم لا.	Obj.UseMaskColor = Boolean

أهم وظائف الكائن ListImages:

الوظيفة	عملـــها	الصورة العامة لها
Add	تستخدم لإضافة صورة جديدة للأداة.	Set I = Obj.Add ([index],[Key],[Pict])
ملاحظة: حيث Pict تمثل مسار ملف الصورة.		
Remove	تستخدم لإزالة صورة من الأداة.	Obj.Remove (index)
ملاحظة: يتم إزالة صورة معينة بواسطة رقم فهرسها ، علماً بأن أول صورة تحمل فهرساً رقمه (٠).		
Clear	تستخدم لإزالة جميع الصور في الأداة.	Obj.Clear

إضافة الصور وقت التصميم:
ضع الأداة ImageList على النافذة.
انقر بالزر الأيمن للفأرة على الأداة فتظهر قائمة منبثقة اختر منها (خصائص Properties).
يظهر مربع حوار الخصائص ، اختر التبويب حتى يظهر المربع كما بالصورة:

انقر الزر Insert Picture (إدراج صورة) فيفتح مربع حوار لاختيار اسم ملف الصورة.
يتم كتابة رقم الصورة في الحقل Index تلقائياً.

ملاحظة:
عند حذف صورة يتم إعادة ترتيب الفهرس مجدداً.

إضافة الصور وقت التشغيل:
لإضافة صورة للأداة وقت التشغيل Run Time نستعمل الوظيفة Add كما بالمثال التالي:

```
Dim I as ListImage
Set i = ImageList.ListImages.Add( ,"Cut",LoadPicture("D:\Cut.Bmp"))
```

وكم تلاحظ من المثال تم استعمال الوظيفة Add التابعة للكائن ListImages واستعمال الدالة LoadPicture لتحميل ملف الصورة بالأداة.
مازالت تحتاج لشرح كويس

استخراج الصور من الأداة ImageList:
بعد وضع الصور بالأداة وكتابة فهارسها ومفاتيحها يمكن إظهار تلك الصور على عناصر أخرى تدعم الصور مثل PictureBox أو ImageBox وغيرها مكن خلال الخاصية picture كما بالمثال:

```
Picture1.Picture = ImageList.ListImages("Cut").Picture
```

وكما تلاحظ فقد استعملنا المفتاح Key ("Cut") للوصول إلى الصورة المطلوبة وعرضها على أداة الصور Picture١.

تخزين (حفظ) صورة موجودة على الأداة في القرص:
يمكننا استخدام الدالة SavePicture لتخزين صورة معينة موجودة على الأداة في القرص بالصورة التالية:

```
SavePicture (ImageList.ListImages("Cut").Picture , "C:\Cut.Bmp")
```

إظهار الصور شفافة:
كما أوضحت في جدول الخصائص ، هناك خاصية تتبع هذه الأداة هي MaskColor وظيفتها تحديد اللون الذي سيكون شفافاً في الصورة المعروضة على العناصر والأدوات الأخرى (بشرط أن تكون الخاصية UseMaskColor = True).
بصورة تلقائية يكون اللون الرمادي شفافاً ، لكن يمكنك تغيير هذه قيمة الخاصية إلى اللون الذي ترغبه من خلال نافذة الخصائص وقت التصميم أو بالبرمجة وقت التشغيل كما يلي:

```
ImageList1.MaskColor = VbWhite
ImageList1.UseMaskColor = True
```

الخصائص التلقائية للعناصر البرمجية:

من المستحسن كتابة اسم الخاصية عند استدعائها ، إلا أن فجول بيسك يدرك خصائص معينة لعناصر برمجية معينة على أنها افتراضية فتدرج آلياً إذا لم تذكر خاصية أخرى بعد اسم الكائن ، فمثلاً الخاصية Text لتابعة لصندوق النصوص هي الافتراضية فلو كتبنا a = Text١ فمعناها a = Text١.Text وكذلك الخاصية Caption التابعة لمربع التسمية Label وغيرها من الخصائص الأخرى.

والجدول التالي يبين الخصائص التلقائية لبعض العناصر والدوات البرمجية:

الأداة	الخاصية التلقائية
CheckBox	Value
ComboBox	Text
DirListBox	Path
DriveListBox	Drive
FileListBox	FileNmae
HscrollBar	Value
Image	Picture
Label	Caption
Option Button	Value
Picture Box	Picture
TextBox	Text
VscrollBar	Value

قواعد تسمية العناصر:

هناك خاصية واحدة فقط تتوفر في كافة العناصر والأدوات البرمجية مهما كان نوعها وهي الخاصية Name ، وهي تلك المجموعة من الحروف التي تمثل الاسم البرمجي للأداة في البرمجة (الكود). هذه الخاصية لا يمكن أن تكون فراغاً ، أو أن يطلق نفس الاسم على أكثر من عنصر لأكثر من نوع.

عندما نضع عنصراً على النافذة ، يقوم فجول بيسك بتسميته تلقائياً ، فعلى سبيل المثال عند وضع أول صندوق نصوص على النافذة يكون اسمه البرمجي Text١ تلقائياً ، وعند وضع الصندوق الثاني يكون اسمه البرمجي Text٢ وهكذا. وفي حالة وضع زر الأوامر الأول يكون اسمه البرمجي Command١ والثاني Command٢.

هذه الأسماء التلقائية تتيح لنا إضافة العناصر إلى النافذة وتضمن عدم تكرار نفس الاسم. لاحظ أن قيمة الخاصية Caption بالنسبة لزر الأوامر وعنصر التسمية Label منطبقة كما هو الحال بالنسبة للخاصية Text لصندوق النصوص مع قيمة الخاصية Name رغم أن كل خاصية مستقلة عن الأخرى.

ولأن الخاصية Name تمثل الاسم البرمجي للأداة في الكود ، فإنه من العادات الحسنة أن نسمي الأداة باسم يبين نوعها وعملها ، هذه العملية مهمة بقدر ما لاختيار الأسماء المناسبة للمتغيرات والمعرفات من أهمية.

قامت شركة مايكروسوفت بوضع جدول لتسمية العناصر ، بحيث لو أخذنا ثلاثة أحرف من اسم نوع العنصر ووضعناها في بداية الاسم البرمجي وزدنا عليه بعض الحروف التي توضح وظيفة وعمل العنصر لنتج لنا الاسم النموذجي للعنصر ، وهذا الجدول يوضح البادئات التي وضعتها مايكروسوفت لتسمية بعض العناصر البرمجية:

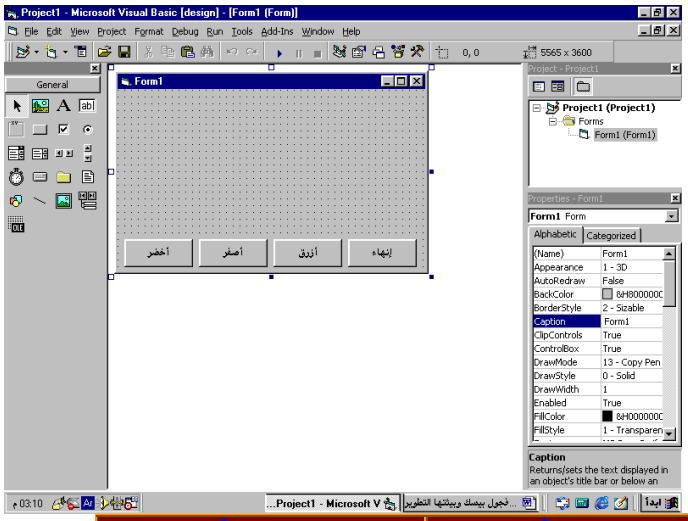
العنصر	البادئة	مثـــــال
CommandButton	Cmd	CmdExit
Textbox	Txt	TxtStdName
Label	Lbl	LblTime
PictureBox	Pic	PicUser
OptionBox	Opt	OptShutDown
CheckBox	Chk	ChkAgree
ComboBox	Cbo	CboItems
ListBox	Lst	LstBooks
Timer	tim	TimTimeDate
Frame	Fra	FraInfo
Data	Dat	DatSales
HscrollBar	Hsb	Hsbvalues

VscrollBar	Vsb	VsbValues
DriveListBox	Drv	DrvPro
DirListBox	Dir	DirProDir
FileListBox	Fil	FilInformation
Line	Lin	LinSeparater
Shape	Shp	ShpRectangle
OLE	Ole	OleWord
Form	Frm	FrmMain

مع تحياتي
أبوبكر شرف الدين (الهاكر) / ليبيا

مثال تطبيقي

مثال:
صمم برنامجاً تطبيقياً بلغة فجول بيسك بحيث تكون واجهته كما بالصورة ويقوم:



بتلوين خلفية النافذة باللون الأزرق عند النقر على الزر (أزرق).
بتلوين الخلفية باللون الأصفر عند النقر فوق الزر (أصفر).
بتلوين الخلفية باللون الأخضر عند النقر فوق الزر (أخضر).
بإنهاء البرنامج عند النقر فوق الزر (إنهاء).

طريقة تصميم البرنامج:
التصميم النظري:
نضع عدد ٤ أزرار تحكم على النافذة.
نضبط الخصائص كما بالجدول التالي:

العنصر	الخاصية	القيمة
إنهاء	Name	CmdExit
	Caption	إنهاء
أزرق	Name	CmdBlue
	Caption	أزرق
أصفر	Name	CmdYellow
	Caption	أصفر
أخضر	Name	CmdGreen
	Caption	أخضر

البرمجة:
برمجة الزر إنهاء:
ننقر فوق الزر إنهاء نقرتين سريعتين فيظهر لنا السطران التاليان:

```
Private Sub CmdExit_Click( )
End Sub
```

فنكتب بينهما الأمر **End** وهو أمر إنهاء البرنامج.

برمجة الزر أزرق:
ننقر فوق الزر أزرق نقرتين سريعتين ثم نكتب:
Form1.BackColor = VbBlue
برمجة الزر أصفر:
ننقر فوق الزر أصفر نقرتين سريعتين ثم نكتب:
Form1.BackColor = VbYellow
برمجة الزر أخضر:
ننقر فوق الزر أخضـر نقرتين سريعتين ثم نكتب:
Form1.BackColor = VbGreen

بعد ذلك نضغط المفتاح F٥ لتنفيذ البرنامج ، ننقر أحد الأزرار ونلاحظ ما يحدث.

أسئلة:
١- ما هي واجهة المستخدم User Interface؟
٢- لماذا يفضل استخدام الأيقونات والصور في البرنامج؟
٣- بأي شيء نتعامل مع النافذة؟
٤- لماذا لا يلزمنا تنشيط النافذة برمجياً؟
٥- ما الفرق بين الحدثين Load و Activate.
٦- ما هي بؤرة التحكم؟
٧- كيف نعرف ما هو المفتاح المضغوط على العنصر عندما يكون في بؤرة التحكم؟
٨- كيف تتم تسمية العناصر حسب الجدول الذي وضعته شركة مايكروسوفت؟

استخدام مربعات الرسائل

الدالة (أو العبارة) Msgbox:

توفر لنا لغة فجول بيسك دالة مهمة هي **Msgbox** ، هذه الدالة تعرض لنا مربع رسائل يقوم بعرض رسالة معينة على هيئة نافذة بها زر أو أكثر.
عند استخدام **Msgbox** كعبارة (وليس كدالة) فإنها لا ترجع أية قيمة ، فقط تقوم بعرض الرسالة المطلوبة والزر (موافق ok).

Msgbox Message, Type, Title

حيث:

Message: هي الرسالة المطلوب عرضها.
Type: الأزرار والأيقونات المطلوب عرضها.
Title: النص الذي سيظهر على شريط عنوان الرسالة.

وعند ظهور مربع الرسائل هذا لا تستطيع عمل أي شيء في البرنامج حتى تقوم بإغلاقها.
وعند استخدام **Msgbox** كدالة فإنها ترجع قيمة موجبة صحيحة تحدد أي زر تم الضغط عليه ويتم استدعاؤها بالطريقة التالية:

```
Dim Ansr as integer

Ansr = Msgbox ( Message, type, Title )
```

يستخدم البارامتر **Type** لتحديد أربعة مكونات وكما يلي:

المكون الأول: ومن خلاله يتم تحديد الأزرار التي ستظهر على المربع:

الـــزر	القيمة	الزر الذي سيعرض
vbOKOnly	0	OK
vbOKCancel	1	OK Cancel
vbAbortRetryIgnore	2	Abort Retry Ignore
vbYesNoCancel	3	Yes No Cancel
vbYesNo	4	Yes No
vbRetryCancel	5	Retry Cancel

المكون الثاني: ومن خلاله يتم تحديد الأيقونات التي ستظهر في مربع الرسالة:

الخيـــار	القيمة	الأيقونة التي ستظهر
vbCritical	16	
vbQuestion	32	
vbExclamation	48	
vbInformation	64	

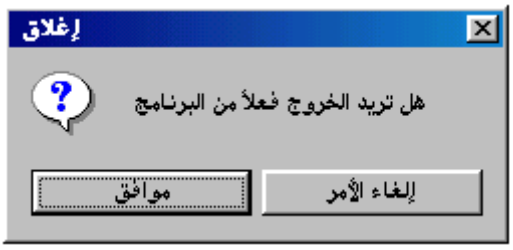
المكون الثالث : يحدد أي زر يكون هو الزر التلقائي بحيث لو تم الضغط على المفتاح Enter يتم اختياره.

الخيـــار	القيمة
vbDefaultButton1	0
vbDefaultButton2	256
vbDefaultButton3	512
vbDefaultButton4	768

لاحظ أن هناك قيمة عددية مقابل كل ثابت رمزي في الجداول السابقة ، فمثلاً : لو كتبت ٢٥٦ هو نفسه لو كتبت VbDefaultButton٢ في جملة الدالة Msgbox ومعناها جعل الزر الثاني هو الزر النشط لكن من الأفضل كتابة الثابت الرمزي بدلاً من القيمة الرقمية.
وكما قلت ، فإن الدالة Msgbox ترجع قيمة رقمية صحيحة موجبة ، هذه القيمة ستكون إحدى القيم المحتملة في الجدول التالي:

الـــزر	الثابت الرمزي المرجع	القيمة
OK	VbOK	1
Cancel	VbCancel	2
Abort	VbAbort	3
Retry	VbRetry	4
Ignore	VbIgnore	5
Yes	VbYes	6
No	VbNo	7

مثال:
لو كتبنا السطر البرمجي التالي:
A = Msgbox("هل تريد الخروج فعلاً من البرنامج",VbOkCancel + VbQuestion, "إغلاق")
سيتم إظهار مربع رسالة يعرض النص : هل تريد الخروج فعلاً من البرنامج ، ويعرض علامة استفهام VbQuestion والزرين موافق وإلغاء الأمر VbOkCancel ، وأخيراً يعرض النص : إغلاق على شريط عنوان مربع الرسالة كما بالصورة:



فإذا اختار المستخدم الزر (موافق) فإن قيمة المتغير a ستكون (١) باعتبار أن (موافق = ok) وإذا اختار الزر (إلغاء الأمر) فتكون a = ٢.

استخدامات مربع الرسائل:

باعتبارك من مستخدمي النظام وندوز فلا بد أنك استعملت هذه المربعات بكثرة ، فمثلاً ، قد واجهك مربع رسالة يحثك على تخزين التغييرات التي أجريتها على مستند عندما أردت الخروج من برنامج الرسام Paint ، أو ذلك المربع الذي واجهك في برنامج الطباعة Word وغيرها.

مثال/ لو عدنا إلى المثال السابق ، وقمنا بتعديل برمجة الزر (إنهاء) إلى البرمجة التالية:

```
Dim a as integer
A = Msgbox ("خروج", VbYesNo + VbQuestion, "هل تريد إنهاء البرنامج")
If a = 6 then
End
End if
```

فماذا يحدث؟

يتم اختبار قيمة المتغير a الذي يمثل الزر المضغوط في مربع الرسائل ، وبالنظر إلى الجدول السابق ، فإذا كانت قيمة a = ٦ أي أن الزر المضغوط هو الزر (نعم = yes) يتم إنهاء البرنامج ، وإلا ، سيتم إخفاء مربع الرسائل دون إنهاء البرنامج.

أعد قراءة الكتاب مرة أخرى

بالتوفيق: