الفصل الثاني

الخصائص وعناصر التحكم والكائنات

يركز هذا الفصل على عناصر تحكم لغة فيجول بيسك، مثل شريطي التمرير الأفقي والعمودي Scroll Bar، ومربعات النص Text Boxes، وأزرار الخيارات Option Button، وأزرار الأوامر Command Button. ستتعلم من هذا الفصل كيف تضع هذه العناصر داخل برامجك، وكيفية تغيير خصائصها، وكيفية ربط النصوص البرمجية بها، (أي ربطها ببرامج متعلقة بها).

تُقدم معظم البرامج، معلومات إلى المستخدم، وتتلقى منه معلومات أيضاً. تُدعى عملية تبادل المعلومات بين التطبيق وبين المستخدم، بواجهة المستخدم User Interface. تَستخدم جميع برامج الويندوز، عناصر التحكم Controls، لتزويد المستخدم بواجهة سهلة ومفهومة (هذا من أهم أسباب شيوع النظام ويندوز). يوضح هذا الفصل مدى سهولة بناء واجهة استخدام جذابة في لغة فيجول بيسك.

عنصر تحكم شريط التمرير

يستخدم شريط التمرير Scroll Bar بكثرة في برامج ويندوز. يمكّنك استخدام هذا العنصر، من اختيار قيمة معينة، بوضع مؤشّر شريط التمرير عند موقع محدد منه، بدلاً من كتابة القيمة.

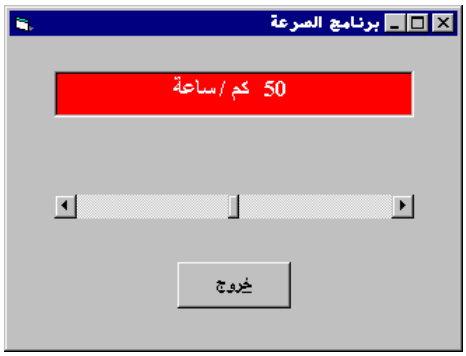
ملاحظة
أطلقنا على شريط التمرير في الفصل الأول كلمة كائن Object، لكن اعتباراً من هذا الفصل سنشير إليه بمصطلح عنصر تحكم Control. في معظم الأحوال، يعتبر الكائن هو نفسه عنصر تحكم، لكن ليس دائماً، فالنموذج Form هو كائن، لكنه ليس عنصر تحكم. نستطيع استخدام كلمة كائن للدلالة على عنصر تحكم، إذا كان هذا العنصر سيوضع في نموذج.

برنامج السرعة

يوضح برنامج السرعة، كيفية استخدام شريط التمرير للحصول على قيمة معينة من المستخدم.

يُفترض في برنامج السرعة إنجاز ما يلي:

- يظهر الإطار المبين في الشكل ٢-١ عند بدء تشغيل برنامج السرعة، يُفترض أن يوضع مؤشّر شريط التمرير عند مركز شريط التمرير (الموقع الافتراضي)، وأن تظهر الرسالة "٥٠ كم / ساعة" (قيمة السرعة) ضمن مربع النص.



الشكل ٢-١

نافذة برنامج السرعة.

- ينبغي على مربع النص، إظهار التغير في السرعة عند تغيير موضع مؤشّر شريط التمرير. فمثلاً يجب إظهار القيمة صفر عندما يوضع المؤشّر عند أقصى اليمين، أما عند وضعه عند أقصى اليسار فيجب إظهار القيمة ١٠٠.

❑ يؤدي نقر الزر **خروج** لإنهاء البرنامج.

التمثيل المرئي لبرنامج السرعة

يَستخدم برنامج السرعة عنصر تحكم شريط التمرير الأفقي. يبين الشكل ٢-٢ شكل عنصر التحكم هذا. طبعاً يختلف موضع هذا العنصر ضمن مربع الأدوات Toolbox تبعاً لاختلاف إصدار لغة فيجول بيسك المستخدم. ويؤدي وضع مؤشّر الفأرة فوق رمز شريط التمرير الأفقي، دون النقر عليه إلى ظهور مستطيل أصفر يحمل الرسالة HscrollBar داخله.

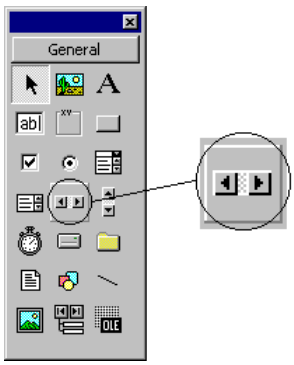
بهذه الطريقة، تتمكن من التحقق من وجود رمز شريط التمرير الأفقي في إطار مربع الأدوات.

❑ نفّذ فيجول بيسك، ثم انقر الزر **إلغاء الأمر** في الإطار New Project في حال ظهوره لإغلاق هذا الإطار، ثم اختر البند **New Project** من قائمة **File** للغة فيجول بيسك.

يستجيب فيجول بيسك بإظهار الإطار New Project.

❑ اختر الرمز Standard EXE من ضمن الإطار New Project، ثم انقر الزر **موافق**.

يستجيب فيجول بيسك بإنشاء مشروع جديد.



الشكل ٢-٢

رمز شريط التمرير الأفقي

داخل إطار مربع الأدوات.

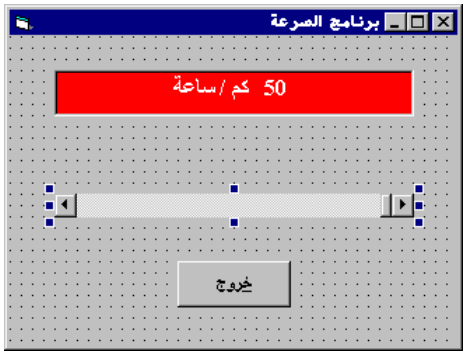
سنحفظ الآن المشروع الجديد الذي أنشأناه:

- ❑ أنشئ الدليل C:\VB5Prg\Ch02.
- ❑ تحقق من اختيار النموذج Form1، ثم اختر البند **Save Form1 As** من قائمة **File**.

يستجيب فيجول بيسك بإظهار مربع الحوار Save File As.

- ❑ استخدم مربع الحوار Save File As لحفظ النموذج باسم Speed.Frm في الدليل C:\VB5Prg\Ch02 واختر البند **Save Project As** من قائمة **File**، ثم استخدم مربع الحوار Save Project As لحفظ المشروع باسم Speed.vbp في الدليل نفسه.
- ❑ عدّل النموذج Form1 طبقاً للجدول ٢-١.

يُفترض أن يبدو النموذج لدى اكتماله، كما في الشكل ٢-٣.



الشكل ٢-٣

النموذج frmSpeed

في طور التصميم.

الجدول ٢-١. جدول خصائص النموذج frmSpeed.

الكائن	الخاصية	القيمة
Form	**Name**	**FrmSpeed**
	BackColor	Light gray
	Caption	برنامج السرعة
	RightToLeft	True
CommandButton	**Name**	**cmdExit**
	Caption	&خروج
	RightToLeft	True
Horizontal Scroll Bar	**Name**	**hsbSpeed**
	Min	0
	Max	100
	RightToLeft	True
TextBox	**Name**	**txtSpeed**
	Alignment	2-Center
	Font	(اختر ما شئت)
	BackColor	Red
	ForeColor	White
	Text	٥٠ كم / ساعة
	MultiLine	True
	RightToLeft	True

❑ ذكرنا في الفصل الأول، أن النقر المزدوج على رمز عنصر التحكم ضمن مربع الأدوات، يؤدي إلى وضع عنصر التحكم ذاك، ضمن النموذج الحالي.

يستجيب فيجول بيسك بوضع عنصر التحكم في وسط النموذج الحالي. تستطيع بعد ذلك نقله إلى مكان آخر عن طريق سحبه بمؤشّر الفأرة. كما تستطيع تكبيره أو تصغيره بسحب المقابض التي تظهر حوله.

❑ للولوج إلى خصائص عنصر التحكم، تأكد من اختيار هذا العنصر على النموذج (أي توضّع المقابض حوله)، ثم اختر البند **Properties Windows** من قائمة

View. أو تستطيع بدلاً من ذلك، النقر بالزر الأيمن للفأرة على العنصر، ثم اختيار البند **Properties** من القائمة الفرعية السريعة التي ظهرت.

يستجيب فيجول بيسك بإظهار إطار الخصائص Properties لعنصر التحكم المختار.

تستطيع الآن تغيير خصائص هذا العنصر.

❑ احفظ المشروع باختيار البند **Save Project** من قائمة **File** التابعة لفيجول بيسك.

كتابة نص برنامج السرعة

سنكتب الآن نص برنامج السرعة:

❑ اكتب النص التالي ضمن الإجراء cmdExit_Cilck() التابع للنموذج frmSpeed:

```
Private Sub cmdExit_Click()

    End
End Sub
```

يُنفذ نص البرنامج السابق آلياً، عند نقر الزر **خروج**، ويُنهي تنفيذ برنامج السرعة.

ملاحظة
لإدخال نص البرنامج للزر **خروج**، انقر نقراً مزدوجاً على الزر **خروج** في مرحلة التصميم، فيظهر إطار البرنامج للإجراء cmdExit_Cilck(). ويكون هذا الإجراء جاهزاً للتعديل من قبل المستخدم. تأكد أن مربع السرد الواقع في الزاوية العليا اليسارية، من إطار البرنامج يحوي البند cmdExit، وأن المربع المجاور له يحوي البند Click.

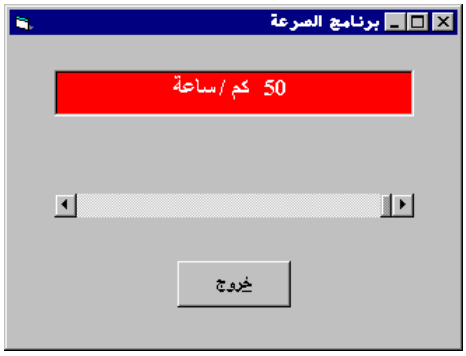
❑ احفظ المشروع باختيار البند **Save Project** من قائمة **File** التابعة لفيجول بيسك.

تنفيذ برنامج السرعة

لم ننته بعد من كتابة نصوص برنامج السرعة، لكن رغم ذلك، سننفذ برنامج السرعة لرؤية نتائج ما أنجزناه حتى الآن.

❑ نفّذ برنامج السرعة (ضغط مفتاح F5 أو اختيار البند **Start** من قائمة **Run**).

يظهر إطار برنامج السرعة كما هو مبين في الشكل ٢-٤.



الشكل ٢-٤

إطار برنامج السرعة.

❑ غيّر موضع مؤشّر شريط التمرير بواسطة الفأرة.

كما تلاحظ، لا يظهر أي شيء ضمن مربع النص، والسبب في ذلك طبعاً، هو أننا لم نكتب نص البرنامج اللازم لإظهار القيم الموافقة لتغيير مؤشّر شريط التمرير.

❑ انقر الزر **خروج** لإنهاء البرنامج.

الخصائص Min و Max و Value لشريط التمرير

تشرح الفقرات التالية بعض خصائص شريط التمرير:

الخاصيتان Min و Max

يمثّل شريط التمرير مجموعة من القيم. تحدد الخاصية Min القيمة الدنيا، وتحدد الخاصية Max القيمة العليا. مثلاً، تأخذ الخاصية Min في الجدول ٢-١ القيمة صفر، وتأخذ الخاصية Max القيمة ١٠٠، وهذا يعني أن شريط التمرير يمكن أن يعطي أية قيمة بين الصفر و ١٠٠.

الخاصية Value

تمثل الخاصية Value التابعة لشريط التمرير، القيمة الراهنة لهذا الشريط، وبالتالي فقد تكون أية قيمة صحيحة بين الرقم صفر والرقم ١٠٠ حسب مثالنا هذا. لم نعط الخاصية Value قيمة معينة أثناء مرحلة التصميم، وبالتالي ستستخدم القيمة الافتراضية

(القيمة صفر) لهذه الخاصية، وعند تنفيذ البرنامج، يوضع مؤشّر شريط التمرير عند الموضع المرافق للخاصية Value (أي عند أقصى يمين شريط التمرير، وهو الموضع المرافق للقيمة صفر للخاصية Value).

الآن، وباعتبار أن السرعة الافتراضية يجب أن تكون ٥٠، لهذا يجب إسناد القيمة ٥٠ للخاصية Value التابعة لشريط التمرير:

- ❑ أسند القيمة ٥٠ للخاصية Value.

الآن، ستجد عند تنفيذ البرنامج، أن الموضع الافتراضي لمؤشّر شريط التمرير سيكون في وسطه (أي منتصف المسافة بين صفر و ١٠٠).

لاحظ أن الجدول ٢-١ يطالبك بإسناد القيمة "٥٠ كم / الساعة" للخاصية Text التابعة لمربع النص. فعند تشغيل البرنامج، ستجد أن مربع النص يُظهر القيمة الابتدائية "٥٠ كم / الساعة" والمشابهة للموضع الراهن لمؤشّر شريط التمرير (Value = 50).

تركيز Focus لوحة المفاتيح

تستطيع ضغط المفتاح Tab من على لوحة المفاتيح، أثناء عمل البرنامج، لنقل التركيز من عنصر تحكم إلى آخر. وتستطيع تمييز عنصر التحكم الذي يمتلك التركيز بسهولة، لأن ويندوز يعطي دلالة على ذلك، (توضّع الإضاءة عنده، أو يظهر حول عنوانه مستطيل منقط .. الخ).

فمثلاً، يظهر مؤشّر وامض في مربع النص، إذا كان التركيز موضوعاً عنده، بينما يومض مؤشّر شريط التمرير عندما يكون التركيز موضوعاً عنده. كما يظهر مستطيل منقط حول عنوان الزر **خروج** (مثلاً) إذا كان هذا الزر يمتلك التركيز، وهكذا.

ما المقصود بأن عنصر تحكم ما، يمتلك التركيز ؟!. المقصود من ذلك، هو أنك تستطيع استخدام لوحة المفاتيح للتحكم به عند امتلاكه للتركيز. جرب مثلاً ما يلي لرؤية ظاهرة تركيز لوحة المفاتيح على أرض الواقع:

- ❑ نفّذ برنامج السرعة. (لاحظ أن مؤشّر شريط التمرير يتوضع في الوسط وهذا طبعاً بسبب إسناد القيمة ٥٠ للخاصية Value التابعة لشريط التمرير).

- اضغط المفتاح Tab في لوحة المفاتيح، إلى أن يصل التركيز إلى شريط التمرير، (ستجد أن مؤشّر شريط التمرير يُومض).

يمتلك شريط التمرير الآن التركيز، استخدم مفتاحي الأسهم اليميني واليساري على لوحة المفاتيح لتحريك مؤشّر شريط التمرير. باعتبار أن شريط التمرير يمتلك تركيز لوحة المفاتيح فإن الضغط على مفتاحي الأسهم على لوحة المفاتيح يكافئ نقر زري السهمين اليساري واليميني لشريط التمرير. جرب ضغط المفاتيح Home و End و PgUp و PgDn وراقب النتائج.

- اضغط المفتاح Tab حتى يصل التركيز إلى الزر **خروج**، ثم اضغط مفتاح Space أو مفتاح Enter. وهذا يكافئ نقر الزر **خروج** لإنهاء البرنامج.

لاحظ كم من العمليات تستطيع إنجازها بواسطة برنامج السرعة!. تستطيع تبديل موضع مؤشّر شريط التمرير، وتكبير أو تصغير إطار البرنامج (بسحب حواف الإطار)، ونقل إطار البرنامج بسحب شريط عنوانه، وإنجاز الكثير من مهام ويندوز القياسية الأخرى. الجميل في الموضوع، أنه لا يلزم كتابة أي نص برمجي لإنجاز ذلك. بل لعل ذلك من أهم محاسن كتابة برامج تحت بنية ويندوز.

فالمعالم القياسية لويندوز تكون مبرمجة أصلاً في برامجك، ولست بحاجة كمستخدم ويندوز (أو مستخدم لبرامجك)، الإلمام بكل المظاهر القياسية لويندوز حتى تتمكن من العمل بكفاءة وفق هذه البنية.

تحسين برنامج السرعة

سنحسّن برنامج السرعة الآن:

- انقر نقرة مزدوجة على عنصر تحكم شريط التمرير ضمن النموذج لإظهار الإجراء hsbSpeed_Change() تحقق بأنّ مربع السرد في الزاوية اليسرى العليا من إطار نص البرنامج يحمل العبارة النصية hsbSpeed، وأنّ مربع السرد المجاور له، يحمل العبارة النصية Change، وبالتالي، عند قراءتهما سوياً تنتج العبارة

.hsbSpeed_Change()

❑ أدخل النص التالي ضمن الإجراء hsbSpeed_Change():

```
Private Sub hsbSpeed_Change()
    txtSpeed.Text = Str(hsbSpeed.Value) + " كم / ساعة "
End Sub
```

ينفذ الإجراء hsbSpeed_Change() (حسب ما يتبدى من اسمه)، عند تغيير موضع مؤشّر شريط التمرير. وبالتالي تتغير الخاصية Value تلقائياً تبعاً لذلك التغيير. فمثلاً، تُصبح قيمة الخاصية Value مساوية الصفر، عند وضع مؤشّر شريط التمرير عند أقصى يمين الشريط، وذلك بسبب إسناد القيمة صفر إلى الخاصية Min.

ملاحظة

تنتقل بداية شريط التمرير من الجهة اليسرى إلى الجهة اليمنى، عند إسناد القيمة True للخاصية RightToLeft. لمزيد من المعلومات عن الخاصية RightToLeft اقرأ الفصل الثاني والعشرين، (إنشاء تطبيقات عربية السمة مع فيجول بيسك).

يتوجب على مربع النص إظهار قيمة الموضع الجديد لمؤشّر شريط التمرير، عند تغيير موقعه. أي بكلمة أخرى، يلزمنا إسناد قيمة الخاصية Value لشريط التمرير، إلى الخاصية Text لمربع النص، وهذا هو دور العبارة التالية:

```
txtSpeed.Text = Str(hsbSpeed.Value) + " كم / ساعة "
```

فمثلاً، إذا كانت قيمة الخاصية Value لشريط التمرير تساوي ٢٠، فقيمة الخاصية Text لمربع النص ستساوي ٢٠ كم / ساعة.

تتوقع الخاصية Text أن يُسند لها قيمة نصية (سلسلة كتابية)، أما الخاصية Value فهي عبارة عن قيمة عددية. مما يعني أنّه يجب استخدام التابع الوظيفي Str() لتحويل القيمة العددية للخاصية Value إلى سلسلة كتابية. يُكتب ضمن قوسي التابع الوظيفي Str()، القيمة العددية المطلوب تحويلها إلى سلسلة كتابية، فمثلاً، يُستخدم التابع Str(11) لتحويل العدد ١١ إلى السلسلة الكتابية "١١". كما يُستخدم التابع Str(12345) لتحويل العدد ١٢٣٤٥ إلى السلسلة الكتابية "١٢٣٤٥".

ملاحظة

عند تحويل القيمة الرقمية إلى قيمة نصية، فإنها تفقد قيمتها الرقمية، وتصبح كأي حرف آخر. طبعاً، يوجد تابع وظيفي معاكس للوظيفة Str() وهو التابع Val()، الذي يحول القيمة النصية ("١٢٣٤") إلى قيمة عددية (١٢٣٤).

في حالتنا هذه، يُطلب من الإجراء تحويل القيمة العددية hsbSpeed.Value إلى سلسلة كتابية، ولهذا استخدمنا العبارة التالية:

```
Str(hsbSpeed.Value)
```

وبالتالي، فإذا فرضنا مثلاً، أنّ الموضع الحالي لمؤشّر شريط التمرير يساوي ٣٢ (hsbSpeed.Value = 32)، والعبارة التالية:

```
txtSpeed.Text = Str(hsbSpeed.Value) + " كم / ساعة"
```

تُسند للخاصية Text التابعة لمربع النص، القيمة التالية: ٣٢ كم / ساعة

والآن لنشاهد نتائج ما كتبناه على أرض الواقع:

- ❑ احفظ المشروع باختيار **Save Project** من القائمة **File**.
- ❑ نفّذ برنامج السرعة.
- ❑ انقل مؤشّر شريط التمرير، وستجد أنّ محتويات مربع النص تتغير تبعاً لموضع مؤشّر الشريط.
- ❑ أَنهِ برنامج السرعة بنقر الزر **خروج**.

تغيير محتويات مربع النص مع سحب مؤشّر شريط التمرير

أنهينا برنامج السرعة تقريباً، ولكن بقيت مشكلة واحدة ينبغي حلها. وللتعرف على هذه المشكلة، اتبع الخطوات التالية:

- ❑ نفّذ برنامج السرعة.
- ❑ جرّب سحب مؤشّر شريط التمرير (دون إفلاته)، وستجد أنّ محتوى مربع النص لا يتغير أثناء عملية السحب! وإنما يتغير فقط بعد تحرير المؤشّر.

كم سيكون جميلاً لو يترافق تغير محتوى مربع النص مع حركة مؤشّر شريط التمرير.

يُنفذ الإجراء hsbSpeed_Scroll() آلياً، عند سحب مؤشّر شريط التمرير، لهذا:

- انقر نقرة مزدوجة على عنصر تحكم شريط التمرير ضمن النموذج، لإظهار الإجراء hsbSpeed_Scroll() في مربع السرد الموجود في الزاوية اليسرى والعليا من إطار نص البرنامج ووجود عبارة النص Scroll في مربع السرد المجاور له.
- أدخل النص التالي في الإجراء hsbSpeed_Scroll():

```
Private Sub hsbSpeed_Scroll()
    hsbSpeed_Change
End Sub
```

- احفظ المشروع باختيار **Save Project** من القائمة **File** لفيجول بيسك.

النص الذي أدخلناه في الإجراء hsbSpeed_Scroll() هو التالي:

```
hsbSpeed.Change
```

يتسبب نص الإجراء هذا، بتنفيذ الإجراء hsbSpeed_Change()، الذي كتبناه مسبقاً، مما يعني أنّ نص الإجراء hsbSpeed_Change() سينفذ عند سحب مؤشّر شريط التمرير، الذي يعمل على إسناد الموقع الحالي لشريط التمرير إلى الخاصية Text لمربع النص، دعنا نشاهد بأنفسنا أثر ما كتبناه:

- نفّذ برنامج السرعة.
- اسحب مؤشّر شريط التمرير، وتحقق أن محتويات مربع النص، تتغير تبعاً لسحب مؤشّر شريط التمرير.
- انقر الزر **خروج** لإنهاء برنامج السرعة.

ملاحظة
ينفذ الإجراء hsbSpeed_Change() من ضمن الإجراء hsbSpeed_Scroll() كما يلي: لاحظ، عدم استخدام الأقواس بعد الكلمة hsbSpeed_Change، أي عند استدعاء الإجراء hsbSpeed_Change(). سيؤدي استخدام القوسين بعد اسم الإجراء في مثل هذه الحالة إلى ظهور رسالة خطأ. وبالواقع يُظهر فيجول بيسك جزء البرنامج الذي يحمل الخطأ بلون أحمر، وهي دلالة مرئية على وجود خطأ ضمن نص البرنامج.

كلمة أخيرة حول برنامج السرعة

يوضح برنامج السرعة، كيف تتمكن من تشكيل واجهة مستخدم محكمة، لإدخال الأعداد.

فبدلاً من إجبار المستخدم على إدخال الأعداد بين صفر و ١٠٠ يدوياً، قدمنا للمستخدم شريط تمرير، يمكّن المستخدم من إعطاء أي قيمة صحيحة ضمن المجال المسموح باستخدام شريط التمرير هذا، والحصول على تغذية عكسية (أو تغذية راجعة Feedback وهي الاستجابة الناتجة عن مربع النص) عن مجال الأرقام المسموح بإدخالها.

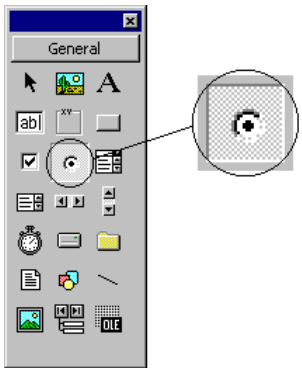
برنامج الخيارات

يوضّح هذا البرنامج كيف يمكنك كتابة برامج تسمح للمستخدم بانتقاء خيار ما Options.

التمثيل المرئي لبرنامج الخيارات

يستخدم برنامج الخيارات عنصر تحكم زر الخيار Options Button. انظر الشكل ٢ – ٥ لرؤية شكل وموقع عنصر التحكم هذا ضمن مربع الأدوات.

علماً بأن الموقع يتغير حسب الإصدار المستخدم، يؤدي وضع مؤشّر الفأرة فوق أي عنصر تحكم ضمن مربع الأدوات إلى ظهور مستطيل أصغر يحمل بداخله اسم ذلك العنصر (مثلاً Options Button في حالتنا هذه).



الشكل ٢–٥

رمز زر الخيار Options ضمن إطار مربع الأدوات.

- أنشئ مشروعاً جديداً باختيار **New Project** من القائمة **File** لفيجول بيسك، ثم اختر الرمز Standard EXE وانقر الزر **فتح** ضمن الإطار New Project.

- تحقق بأن إطار النموذج Form1 هو الإطار الراهن (أي أنه تم اختياره، أو بكلمة أخرى، الإضاءة متوضعة لديه). ثم اختر **Save Form1 As** من القائمة **File** لفيجول بيسك. استخدم الآن مربع الحوار Save File As لحفظ الملف بالاسم Options.Frm في الدليل C:\VB5Prg\Ch02.
- اختر الآن **Save Project As** من القائمة **File** لفيجول بيسك، واستخدم مربع الحوار Save Project لحفظ المشروع بالاسم Options.Vbp في الدليل C:\VB5Prg\Ch02.
- أنشئ النموذج frmOptions طبقاً للجدول ٢-٢.

يفترض أن يبدو النموذج لدى اكتماله كذاك المبين في الشكل ٢-٦.

الشكل ٢-٦

النموذج frmOptions

في طور التصميم.

الجدول ٢-٢. جدول خصائص النموذج frmOptions.

الكائن	الخاصية	القيمة
Form	**Name**	**frmOptions**
	BackColor	Red
	Caption	برنامج الخيارات
	RightToLeft	True
CommandButton	**Name**	**cmdExit**
	Caption	&خروج
	RightToLeft	True
Check Box	**Name**	**chkSound**
	BackColor	Red

	Caption	أ&صوات
	Font	(اختر ما شئت)
	ForeColor	White
	RightToLeft	True
Check Box	**Name**	**chkMouse**
	BackColor	Red
	Caption	ال&فأرة
	Font	(اختر ما شئت)
	ForeColor	White
	RightToLeft	True

الكائن	الخاصية	القيمة
Check Box	**Name**	**chkColors**
	BackColor	Red
	Caption	ال&ألوان
	Font	(اختر ما شئت)
	ForeColor	White
	RightToLeft	True
Option Button	**Name**	**optLevel1**
	BackColor	Red
	Caption	المستوى &١
	Font	(اختر ما شئت)
	ForeColor	White
	RightToLeft	True
Option Button	**Name**	**optLevel2**
	BackColor	Red
	Caption	المستوى &٢
	Font	(اختر ما شئت)
	ForeColor	White
	RightToLeft	True
Option Button	**Name**	**optLevel3**
	BackColor	Red
	Caption	المستوى &٣
	Font	(اختر ما شئت)
	ForeColor	White
	RightToLeft	True

Label	Name	lblChoice
	Alignment	2-Center
	BorderStyle	1-Fixed Single
	Font	(اختر ما شئت)
	RightToLeft	True

ملاحظة

ستضطر غالباً إلى زيادة ارتفاع النموذج frmOptions عند بنائه طبقاً للجدول ٢-٢ وذلك حتى تتسع كل العناصر فيه. ولزيادة الارتفاع اسحب الحافة السفلى للإطار باتجاه الأسفل.

قسم التصاريح العامة General Declarations للنموذج

سنُدخل في هذا القسم جزءاً من برنامج، علماً بأنّ هذا القسم عبارة عن منطقة ضمن إطار نص البرنامج يكتب فيها شتى العبارات العامة.

تعتبر العبارة Option Explicit مثالاً على عبارة عامة. سنتناول المعنى الدقيق لهذه العبارة لاحقاً في هذا الفصل. أما الآن فيكفينا تعلم كيفية تناول قسم التصاريح العامة، وكيفية كتابة نص برمجي داخله.

اتبع الخطوات التالية:

- انقر نقراً مزدوجاً على أي منطقة خالية من النموذج frmOptions لإظهار إطار نص البرنامج (Code Window).

يستجيب فيجول بيسك بإظهار إطار نص البرنامج.

- انقر على رمز السهم النازل لمربع السرد الموجود في الزاوية اليسرى العليا من إطار نص البرنامج، ثم اختر البند (General) من القائمة.
- انقر رمز السهم النازل لمربع السرد المجاور للمربع السرد السابق، والمتوضع عند الزاوية العليا اليمنى من إطار نص البرنامج، ثم اختر البند (Declarations) منه.

يُظهر إطار نص البرنامج الآن قسم التصاريح العامة General Declarations،

حسب ما يوضحه الشكل ٢-٧.

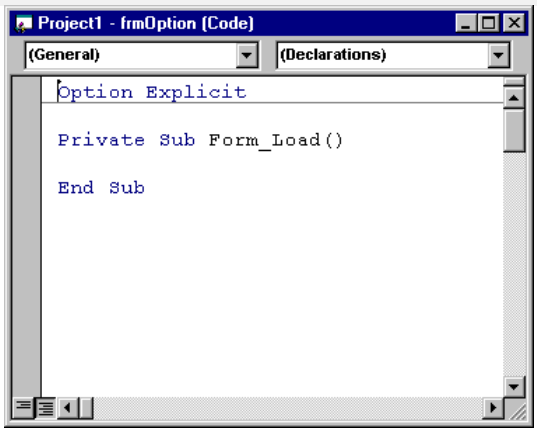
ستُلاحظ النص التالي في الشكل ٢-٧:

```
Private Sub Form_Load()

End Sub
```

حسب ما هو واضح في الشكل ٢-٧، يقع قسم التصاريح العامة فوق الإجراء Form_Load()، وقد تجد عبارات مكتوبة مسبقاً في هذا القسم، مثلاً:

```
Option Explicit
```



الشكل ٢-٧

إظهار قسم التصاريح العامة General Declarations لإطار نص البرنامج Code Window.

❑ تستطيع الآن نقر قسم التصاريح العامة، وكتابة أي نص برنامج إضافي تريد. فمثلاً إذا لم تشاهد العبارة Option Explicit في هذا القسم، فاكتب العبارة التالية:

```
Option Explicit
```

ربط حادثة Click للزر خروج بنص البرنامج المناسب

❑ أدخل النص التالي ضمن الإجراء cmdExit_Click() للنموذج frmOptions:

```
Private Sub cmdExit_Click()

    End
End Sub
```

يُنفذ جزء البرنامج الذي أدخلته، آلياً عند نقر الزر **خروج**. علماً بأنّ جزء البرنامج هذا يُنهي عمل برنامج الخيارات.

تنفيذ برنامج الخيارات

رغم أننا لم ننته بعد من تصميم برنامج الخيارات لكن دعنا ننفذه:

- ❑ نَفّذ برنامج الخيارات.
- ❑ انقر زر الخيار **المستوى١**.

*يستجيب البرنامج، باختيار زر الخيار **المستوى١**، (تظهر دائرة مصمتة ضمن زر الخيار **المستوى١**).*

- ❑ انقر زر الخيار **المستوى٢**.

*يستجيب البرنامج بإلغاء اختيار زر الخيار **المستوى١** (إزالة الدائرة المصمتة من زر الخيار **المستوى١**)، واختيار زر الخيار **المستوى٢** بدلاً عنه (وضع دائرة مصمتة داخله).*

- ❑ انقر زر الخيار **المستوى٣**.

*يستجيب البرنامج بإلغاء اختيار الزر **المستوى٢** ويختار بدلاً منه الزر **المستوى٣**.*

إذاً يسمح فقط باختيار زر خيار واحد في نفس الوقت. تستخدم أزرار الخيارات ضمن البرامج عندما يرغب المستخدم باختيار خيار واحد فقط من أجل عدة خيارات. (لاحظ أنّ بعض كتب ويندوز، تُطلق على هذا الزر اسم الزر الراديوي Radio Button، تشبيهاً له بأزرار الراديو التي لا يسمح بضغط أكثر من زر واحد في نفس الوقت).

- ❑ انقر خانة الاختيار **أصوات**.

*يستجيب البرنامج بوضع علامة اختيار في خانة الاختيار **أصوات**.*

- ❑ انقر خانات الاختيار الأخرى.

كما تشاهد، يُسمح باختيار أكثر من خانة اختيار واحدة في نفس الوقت. استخدم خانات الاختيار، عندما ترغب في أن يتمكن المستخدم من اختيار عدة وضعيات في نفس الوقت. فمثلاً. قد يختار المستخدم إذا كان البرنامج عبارة عن لعبة!، اللعب مع صوت أو بدونه، وبفأرة أو بدونها، وبألوان أو بدون ألوان.

كما يستطيع المستخدم اللعب وفق مستوى أول أو ثاني أو ثالث، لكن لا حكمة من وراء اللعب بثلاثة مستويات مثلاً دفعة واحدة (عملياً، لا يمكن اللعب بثلاث مستويات في نفس الوقت، حسب المثال).

لإلغاء اختيار خانة الاختيار، انقر مجدداً عليه، فتزول علامة الاختيار من داخله.

❑ أَنهِ البرنامج بنقر الزر **خروج**.

التحقق من أزرار الاختيار وخانات الاختيار التي يتم اختيارها

سنكتب الآن نصاً يتحقق من أزرار الخيار وخانات الاختيار التي يتم اختيارها.

❑ أدخل النص التالي ضمن الإجراء chkColors_Click() للنموذج frmOptions:

```
Private Sub chkColors_Click()

    UpdateLabel
End Sub
```

❑ أدخل النص التالي ضمن الإجراء chkMouse_Click():

```
Private Sub chkMouse_Click()

    UpdateLabel
End Sub
```

❑ أدخل النص التالي ضمن الإجراء chkSound_Click():

```
Private Sub chkSound_Click()

    UpdateLabel
End Sub
```

❑ أدخل النص التالي ضمن الإجراء optLevel1_Click():

```
Private Sub optLevel1_Click()

    UpdateLabel
End Sub
```

❑ أدخل النص التالي ضمن الإجراء optLevel2_Click():

```
Private Sub optLevel2_Click()

    UpdateLabel
End Sub
```

❑ أدخل النص التالي ضمن الإجراء optLevel3_Click():

```
Private Sub OptLevel3_Click()

    UpdateLabel
```

```
End Sub
```

ما الذي فعلناه خلال الخطوات السابقة هذه ؟! لقد كتبنا العبارة التالية:

```
UpdateLabel
```

في جميع الإجراءات السابقة. يقصد بالعبارة UpdateLabel اسم إجراء جديد، وسنباشر بعد قليل بكتابته. يُنفذ هذا الإجراء آلياً فور اختيار أي من أزرار الخيار أو خانات الاختيار في هذا المثال:

اتبع الآن الخطوات التالية لإضافة الإجراء UpdateLabel إلى النموذج:

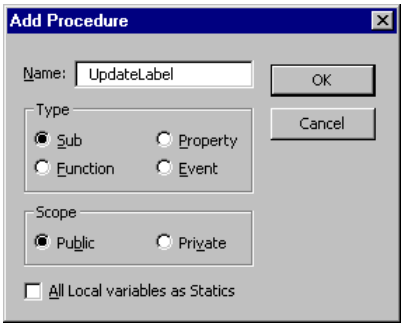
❑ انقر نقراً مزدوجاً في أي منطقة خالية من النموذج.

يستجيب فيجول بيسك بإظهار إطار نص البرنامج (Code Window).

❑ اختر Add Procedure من القائمة Tool لفيجول بيسك.

يستجيب فيجول بيسك بإظهار مربع الحوار Add Procedure.

❑ أدخل UpdateLabel في الحقل Name من مربع الحوار Add Procedure. (لأن UpdateLabel هو اسم الإجراء الجديد الذي سنضيفه). تحقق الآن بأن أزرار الخيارات في مربع الحوار Add Procedure مطابقة للشكل ٢-٨.



الشكل ٢-٨

إضافة الإجراء الجديد UpdateLabel.

❑ انقر الزر **OK** في مربع الحوار Add Procedure.

يستجيب فيجول بيسك بإظهار إطار نص البرنامج عند الإجراء UpdateLabel جاهزاً للتعديل.

ملاحظة
لقد أضاف فيجول بيسك الإجراء UpdateLabel في المنطقة العامة General Area من النموذج frmOptions. ونستطيع التحقق من ذلك كما يلي:

- ضع قراءة مربع السرد الواقع أعلى يسار إطار نص البرنامج على General.
- ضع قراءة مربع السرد الواقع أعلى يمين إطار نص البرنامج علىDeclarations.

إذا تفقدت محتوى مربع السرد اليميني فستجد بندين فيه: وهما Declarations وUpdateLabel. يوجد لدينا في قسم التصاريح العامة General Declarations العبارة Option Explicit.

الإجراءات التي تضاف إلى النموذج (مثالنا الإجراء UpdateLabel)، تضاف إلى المنطقة العامة General Area.

من الهام طبعاً معرفة أين تضاف الإجراءات للرجوع إليها لاحقاً بغية قراءتها أو تعديلها.

إذاً، تضاف الإجراءات في المنطقة العامة من النموذج. وللوصول إلى إجراء ما، ضع قراءة مربع السرد الواقع أعلى يسار إطار نص البرنامج على General ثم ضع قراءة مربع السرد اليميني المجاور على اسم الإجراء الذي تريد تعديله أو قراءته.

يكتب فيجول بيسك نيابة عنك، السطرين الأول والأخير من الإجراء، ويبقى عليك كتابة نص الإجراء المناسب:

❑ أدخل النص التالي ضمن الإجراء UpdateLabel:

```
Public Sub UpdateLabel()
   Dim Info
   Dim LFCR
   LFCR = Chr(13) + Chr(10)
   ' الصوت
   If chkSound.Value = 1 Then
      Info = "الصوت : تشغيل"
   Else
      Info = "الصوت : إيقاف"
   End If
   ' الفأرة
   If chkMouse.Value = 1 Then
      Info = Info + LFCR + "الفأرة : تشغيل"
```

```
    Else
       Info = Info + LFCR + "الفأرة : إيقاف"
         f' الألوان
    If chkColors.Value = 1 Then
       Info = Info + LFCR + "الألوان : تشغيل"
    Else
       Info = Info + LFCR + "الألوان : إيقاف"
    End If
    ' المستوى ١
    If optLevel1.Value = True Then
       Info = Info + LFCR + "المستوى : ١"
    End If
    '
    If optLevel2.Value = True Then
       Info = Info + LFCR + "المستوى : ٢"
    End If
    If optLevel3.Value = True Then
       Info = Info + LFCR + "المستوى : ٣"
    End If

    lblChoice.Caption = Info
End Sub
```

- احفظ المشروع باختيار **Save Project** من القائمة **File**.

تنفيذ برنامج الخيارات

دعنا نُنفّذ البرنامج قبل المضي في دراسة نص الإجراء UpdateLabel:

- نفّذ برنامج الخيارات.
- انقر خانات الاختيار وأزرار الخيار المختلفة.

يستجيب برنامج الخيارات، بإظهار حالة خانات الاختيار وأزرار الخيارات ضمن اللافتة lblCHoice (انظر الشكل ٢-٩).

- أَنهِ برنامج الخيارات بنقر الزر **خروج**.

الشكل ٢-٩
إطار برنامج الخيارات
مع حالة التوضعات الحالية
لكل من أزرار الخيارات
وخانات الاختيار.

كيف يعمل برنامج الخيارات

يُنفّذ برنامج الخيارات، الإجراء UpdateLabel عند نقر أحد أزرار الخيارات أو خانات الاختيار.

نص برنامج الإجراء chkColors_Click() للنموذج frmOptions

يُنفّذ هذا الإجراء آلياً، عند نقر خانة الاختيار chkColors:

```
Private Sub chkColors_Click()
    UpdateLabel
End Sub
```

ينفذ بعد ذلك نص الإجراء UpdateLabel والذي سنشرحه بعد قليل.

بطريقة مشابهة، يؤدي نقر أي زر من أزرار الخيارات أو خانات الاختيار الأخرى إلى تنفيذ الإجراء المرافق لذلك العنصر (زر خيار أو خانة اختيار)، وبالتالي تنفيذ الإجراء UpdateLabel.

نص برنامج الإجراء UpdateLabel

إذاً، ينفذ هذا الإجراء حسب ما ذكرنا، عند اختيار أي من أزرار الخيارات أو خانات الاختيار في هذا المثال.

هذا الإجراء ليس إجراء حادثة خاصة بأحد كائنات فيجول بيسك، أي لا ينفذ تلقائياً عند وقوع حادثة ما، وإنما هو إجراء أنشأناه بأنفسنا باستخدام مربع الحوار Add Procedure كما مرّ معنا سابقاً.

من الهام التمييز بين إجراءات مثل cmdExit_Click() (إجراءات مرتبطة بحادثة ما)، وبين إجراءات مثل UpdateLabel، فالإجراء cmdExit_Click() ينفذ آلياً عند وقوع حادثته، ولا حاجة لكتابة أي نص برمجي للتسبب بعملية تنفيذه. ما هو السبب؟!. السبب أنّ فيجول بيسك يعمل!، ففي اللحظة التي يتم فيها نقر الزر **خروج**. تقع حادثة النقر Click ، وبالتالي ينفذ الإجراء cmdExit_Click().

لا ينفذ الإجراء UpdateLabel آلياً، بل يجب على برنامجك أن يستدعيه للتنفيذ، ويتم ذلك بذكر اسمه، وهو السبب الذي دفعنا إلى كتابة العبارة pdateLabel في ستة مواقع، فالعبارة:

```
UpdateLabel
```

تتسبب بتنفيذ الإجراء UpdateLabel. مما يعني أنّ هذا الإجراء سينفذ عند نقر أي من أزرار الخيار أو خانات الاختيار في هذا المثال.

التصريح عن المتحول Info

تصرح أول عبارة كتبناها في الإجراء UpdateLabel عن المتحول Info بالشكل التالي:

```
Dim Info
```

تعتبر Dim تعليمة فيجول بيسك، وتدل أن الكلمة التي تليها (Info في مثالنا هذا)، هي اسم لمتحول سوف نستخدمه لاحقاً في الإجراء. يُستخدم المتحول Info كمتحول نصي، يقوم بتخزين سلسلة من الأحرف الكتابية أثناء تنفيذ الإجراء UpdateLabel.

تستطيع التصريح عن هذا المتحول بالطريقة التالية أيضاً:

```
Dim Info As String
```

لحسن الحظ، يعتبر فيجول بيسك متساهلاً من هذه الناحية، ولا يُجبرك على التصريح عن نوع المتحول، بل يفترض نوعه حسب طريقة استخدامه.

يعرف من لديه شيئاً من الخبرة في لغات البرمجة الأخرى، أنّ بعض لغات البرمجة لا تتطلب من المستخدم التصريح عن المتحولات، لكن تبقى عادةُ التصريح عن المتحولات عادة حسنة، ولمعرفة السبب افترض أنّ الإجراء يحوي على الحسابات التالية:

```
Time = 10
Velocity = 50
Distance = Velocity * Time
lblDistance.Caption = " = المسافة " + Str(Distance)
```

تُسند العبارات الأربعة السابقة، القيمة ١٠ إلى المتحول Time، والقيمة ٥٠ إلى المتحول Velocity، ثم تحسب حاصل ضرب هذين المتحولين وتظهر المسافة Distance، بإسناد قيمة المسافة Distance إلى الخاصية Caption للافتة lblDistance.

لنفترض الآن، أنك كتبت أحد المتحولات بشكل خاطئ، (نسيت مثلاً كتابة الحرف a بعد الحرف t في كلمة Distance) كالتالي:

```
lblDistance.Caption = " = المسافة " + Str(Distnce)
```

يَعتبر فيجول بيسك أنّ **Distnce** (المتحول المكتوب بشكل خاطئ)، هو متحول جديد مختلف عن المتحول الأساسي Distance، ويُسند له آلياً القيمة صفر، وبالتالي تُظهر اللافتة lblDistance الجملة التالية:

```
المسافة = ٠
```

وهذا بالطبع خطأ جسيم، قد تهدر كثيراً من الوقت لاكتشافه. تستطيع تلافي حدوث أمثال هذه الأخطاء، بأن تدعو فيجول بيسك إلى التذمر عند استخدام متحول في فيجول بيسك بدون التصريح عنه مسبقاً. وبالتالي، في مثالنا هذا يجب أن تؤول العبارات السابقة إلى ما يلي:

```
Dim Time
Dim Velocity
Dim Distance

Time = 10
Velocity = 50
Distance = Velocity * Time
lblDistance.Caption = " = المسافة " + Str(Distnce)
```

إذا تم إعداد فيجول بيسك، بحيث يتذمر لدى مصادفته متحول غير مصرح عنه ضمن نص البرنامج، فإنه سوف يعطيك رسالة خطأ أثناء تنفيذ البرنامج ويخبرك بأن

المتحول غير معروف لديه. فمثلاً يضيء فيجول بيسك المتحول Distnce في حالتنا هذه، ويخبرك بأن فيه شيئاً ما خاطئ.

أما كيف يتم إعداد فيجول بيسك للتذمر عند مصادفة متحولات غير مصرح عنها في نص برنامج ؟! فيتم ذلك بوضع العبارة التالية:

```
Option Explicit
```

ضمن قسم التصاريح العامة General Declarations، وهو السبب الذي دفعنا إلى وضع العبارة Option Explicit في قسم التصاريح العامة عبر الأمثلة السابقة.

ملاحظة

ضع العبارة Option Explicit دائماً ضمن قسم التصاريح العامة للنموذج. فبهذه الطريقة تخبر فيجول بيسك بعدم قبول المتحولات غير المصرح عنها وتوفر على نفسك ساعات طويلة من تنقيح الأخطاء. لنتكلم بعمومية أكثر، يؤدي تجاهل العبارة Option Explicit إلى إنفاق ساعات طويلة في تنقيح أخطاء قد تكون بسيطة ناتجة من أخطاء في التهجي، لهذا كن حكيماً ودع فيجول بيسك يجد عنك المتحولات التي كتبتها بشكل خاطئ.

التصريح عن المتحول LFCR

صرحنا أيضاً عن المتحول LFCR ضمن الإجراء UpdateLabel:

```
Dim LFCR
```

ثم أسندنا إلى المتحول LFCR ما يلي:

```
LFCR = Chr(13) + Chr(10)
```

يمثل الرمز Chr(13) رمز المفتاح Enter على لوحة المفاتيح، كما أنّ الرمز Chr(10) هو رمز التغذية السطرية (أي ينقل مؤشر الكتابة إلى سطر جديد). وكما سنرى، تُظهر اللافتة lblChoice سلسلة طويلة تنتشر على عدة أسطر، وذلك بالاستعانة بالمتحول LFCR.

التحقق من قيمة الخاصية Value

يأتي بعد التصريح عن المتحولات في الإجراء UpdateLabel كتلة الشرط If .Else.End:

```
'الصوت
If chkSound.Value = 1 Then
   Info = "الصوت : تشغيل"
Else
   Info = "الصوت : إيقاف"
End If
```

تستطيع في فيجول بيسك، إضافة تعليقات ضمن نص البرنامج، باستخدام رمز الفاصلة العلوية (') أو الكلمة Rem. فمثلاً السطر التالي:

```
'الصوت
```

يطابق السطر:

```
Rem الصوت
```

يَستخدم هذا الكتاب الفاصلة العلوية للدلالة على أسطر التعليقات.

كما يمكن إضافة التعليقات في أسطر البرنامج، كما يلي:

```
MyVariable = 1  ' تهيئة المتحول
```

تعتبر عادة وضع التعليقات ضمن نص البرنامج عادة حسنة، لأنها تسهل قراءة وتنقيح البرامج. تستطيع كتابة أي شيء تريده بعد رمز الفاصلة العلوية (')، يتجاهل فيجول بيسك كل الرموز التي تلي هذا الرمز أي (').

يتحقق الإجراء UpdateLabel من أن قيمة الخاصية Value لخانة الاختيار chkSound تساوي ١. فإذا كانت الخاصية Value تساوي واحد، فهذا يعني أنّ العبارات بين السطر If والسطر Else سوف تُنفذ، وفي هذه الحالة لدينا عبارة واحدة فقط بين If و Else وهي العبارة:

```
Info = "الصوت : تشغيل"
```

حيث تنفذ هذه العبارة عندما Value تساوي الواحد. تسند هذه العبارة السلسلة "الصوت : تشغيل" إلى المتحول Info. لاحظ أنه يجب ذكر كلمة Then ضمن العبارة If.

إذا كانت قيمة الخاصية Value لخانة الاختيار **الصوت** تساوي الواحد، فهذا يعني أنه توجد علامة اختيار في خانة الاختيار chkSound؛ وبالتالي سوف تساوي قيمة المتحول Info إلى:

```
"الصوت : تشغيل"
```

تنفذ العبارات الموجودة بين Else و End If، إذا كانت قيمة الخاصية Value لخانة الاختيار chkSound لا تساوي الواحد. فعند عدم وجود علامة اختيار في خانة الاختيار تكون قيمة الخاصية Value لخانة الاختيار مساوية الصفر، وبالتالي تنفذ العبارة الواقعة بين Else و End If، وتسند هذه العبارة إلى المتحول Info الجملة التالية:

```
"الصوت : إيقاف"
```

إذاً لنلخص ما سبق، يُسنَد إلى المتحول Info إما الجملة:

```
"الصوت : تشغيل"
```

أو:

```
"الصوت : إيقاف"
```

بطريقة مشابهة، تتحقق العبارة If.Else.End If بأن قيمة الخاصية Value لخانة الاختيار chkMouse تساوي الواحد:

```
' الفأرة
If chkMouse.Value = 1 Then
   Info = Info + LFCR + "الفأرة : تشغيل"
Else
   Info = Info + LFCR + "الفأرة : إيقاف"
End If
```

فمثلاً، إذا كانت خانة الاختيار chkSound تحوي علامة اختيار داخلها، وكانت خانة الاختيار chkMouse لا تحوي علامة اختيار، فسوف تتسبب عبارتا If.Else.End If الأوليتين من الإجراء UpdateLabel، بإسناد السلسلة التالية إلى المتحول Info:

```
"الفأرة : إيقاف" + LFCR + "الصوت : تشغيل"
```

وسوف تظهر هذه السلسلة لاحقاً على سطرين:

```
الصوت : تشغيل
الفأرة : إيقاف
```

وذلك بسبب إضافة LFCR بين السلسلتين.

تتحقق عبارة If.Else.End If التالية ضمن الإجراء UpdateLabel بأن قيمة الخاصية Value لخانة الاختيار chkColors تساوي الواحد، وتعدّل المتحول Info تبعاً لذلك:

```
' الألوان
If chkColors.Value = 1 Then
   Info = Info + LFCR + "الألوان : تشغيل"
Else
   Info = Info + LFCR + "الألوان : إيقاف"
End If
```

تتحقق عبارة If.End.If التالية في الإجراء UpdateLabel بأن الخاصية Value لزر الخيار optLevel1 تساوي قيمة الثابت True:

```
If optLevel1.Value = True Then
   Info = Info + LFCR + "المستوى : ١"
End If
```

تحدد الخاصية Value أيضاً حالة عنصر التحكم هذا، فإذا كانت قيمة الخاصية Value تساوي True، فهذا معناه أنه تم اختيار زر الخيار، ويتم إعداد المتحول Info تبعاً لذلك. أما إذا كانت الخاصية Value لزر الخيار optLevel1 لا تساوي True فهذا يعني أنه لم يتم انتقاء زر الخيار هذا.

ملاحظة

الخاصية Value لخانة الاختيار قد تساوي ٠ أو ١ أو ٢. فإذا كانت تساوي الواحد، فهذا معناه وجود علامة اختيار بداخله. أما إذا كانت تساوي صفر، فهذا يعني عدم وجود علامة اختيار، بينما إذا كانت Value لمربع اختيار تساوي ٢، فهذا يعني حالة بين الاثنين، ويظهر بشكل رمادي أو باهت.

ملاحظة

الخاصية Value لزر خيار ما، قد تساوي True أو False. فإذا كانت تساوي True، فهذا معناه وجود نقطة داخل زر الخيار. أما إذا كانت تساوي False فهذا يعني عدم وجود نقطة داخله. وبالتالي عدم اختياره.

تعدّل عبارتا If.End.If التاليتين المتحول Info تبعاً لقيمة الخاصية Value لكل من زري الخيار optLevel2 و optLevel3:

```
If optLevel2.Value = True Then

   Info = Info + LFCR + "المستوى : ٢"
End If
If optLevel3.Value = True Then
   Info = Info + LFCR + "المستوى : ٣"
End If
```

تسند آخر عبارة في الإجراء UpdateLabel محتوى المتحول Info إلى الخاصية Caption للافتة lblChoice:

```
lblChoice.Caption = Info
```

تظهر هذه العبارة محتوى المتحول Info داخل اللافتة lblChoice، كما يظهره الشكل٢-٩.

ماذا لدينا أيضاً ؟!

لا بد أنك أدركت حتى الآن، أنّ البرمجة بلغة فيجول بيسك ترتكز على فهم معنى كل عنصر من عناصر التحكم داخل مربع الأدوات، ومعنى ووظيفة ذلك العنصر. فنفس الخاصية Property، تحمل معاني مختلفة لعناصر التحكم المختلفة.

فمثلاً الخاصية Caption (العنوان) للنموذج، تحوي النص الذي يظهر في شريط عنوان النموذج، أما الخاصية Caption لعنصر التحكم Label (لافتة)، فتحوي النص الذي سيظهر في اللافتة. كذلك الخاصية Value لخانة اختيار، تُشير إلى وجود أو عدم وجود علامة اختيار فيه. والخاصية Value لزر الخيار، تشير إلى وجود أو عدم وجود دائرة مصمتة فيه. وأما الخاصية Value لشريط تمرير، فتشير إلى الموضع الحالي لمؤشّر شريط التمرير.

يحوي مربع الأدوات على رموز عناصر التحكم، ويمتلك كل عنصر تحكم، مجموعته الخاصة من الخصائص Properties. تعتبر بعض عناصر التحكم Controls، عناصر

تحكم ويندوز قياسية، مثال ذلك، شريطي التمرير الأفقي والعمودي، ومربعات النص، واللافتات، وخانات الاختيار، وأزرار الخيار، وأزرار الأوامر.

يمكن إضافة رموز أخرى إلى مربع الأدوات في فيجول بيسك، ثم وضع عناصر التحكم التي تمثلها هذه الرموز في النموذج. تدعى الرموز الإضافية لعناصر التحكم هذه بالاسم ActiveX Controls، وتعرف أيضاً باسم عناصر التحكم OCX. سنتعرف بشكل أوسع على هذه العناصر عبر فصول الكتاب.

لا تُعتبر لغة فيجول بيسك لغة برمجة صعبة التعلم، ولكن هنالك الكثير مما يجب تعلمه. ومفتاح التعلم الناجح هو التمرن والتجريب. والمفترض بعد كتابة برامج هذا الكتاب أن تصبح قادراً على تدريس اللغة. فحاول أن تكتب البرامج، وأن تفهم نصوصها، وكذلك جرّب تغيير خصائص عنصر التحكم أثناء طور التصميم، (أي أثناء مرحلة بناء النموذج – مرحلة التمثيل المرئي للبرنامج)، ومرحلة التنفيذ.

يعني تغيير الخصائص أثناء مرحلة التنفيذ، تغيير قيمة الخاصية داخل نص البرنامج. فمثلاً، تسند العبارة الأخيرة في الإجراء UpdateLabel المتحول Info أثناء مرحلة التنفيذ إلى الخاصية Caption للافتة lblCHoice:

```
lblChoice.Caption = Info
```

بينما بالمقابل، أسند العنوان **برنامج الخيارات** إلى الخاصية Caption للنموذج خلال مرحلة التصميم Design Time.

تتقبل بعض الخصائص تغيير قيمتها، خلال أي من الطورين، طور التصميم، أو طور التنفيذ. بينما تقبل بعض الخصائص تغيير قيمتها فقط أثناء مرحلة التنفيذ، (أي من خلال نص البرنامج فقط). فمثلاً تقبل الخاصية Caption لعنصر التحكم Label تغيير قيمتها خلال كلا الطورين: التصميم والتنفيذ.

ستتعرف في هذا الكتاب على خصائص لا يمكن إسنادها أو تغييرها، إلا من خلال طور زمن التنفيذ.

اصطلاحات التسمية المستخدمة في هذا الكتاب

تُسمى عناصر التحكم عبر هذا الكتاب، وفق الجدول ٢-٣. فتبدأ أسماء أزرار الأوامر مثلاً، بالرموز cmd (كما في cmdMyButton)، وتبدأ أسماء مربعات النص بالرموز txt (كما في txtMyTextBook).

الجدول ٢-٣. اصطلاحات التسمية لعناصر تحكم فيجول بيسك Controls.

نوع عنصر التحكم	البادئة	مثال
Check box	chk	chkReadOnly
Combo box	cbo	cboEnglish
Command button	cmd	cmdExit
Common dialog	dlg	dlgFileOpen
Communications	com	comFax
Data control	dat	datBiblio
Directory list box	dir	dirSource
Drive list box	drv	drvTarget
File list box	fil	filSource
Form	frm	frmEntry
Frame	fra	fraLanguage
Grid	grd	grdPrices
Horizontal scroll bar	hsb	hsbVolume
Image	img	imgIcon
Label	lbl	lblHelpMessage
Line	lin	linVertical
List box	lst	lstPolicyCodes
MCI	mci	mciVideo
MDI child form	mdi	mdiNote
Menu	mnu	mnuFileOpen
Picture	pic	picVGA
Shape	shp	shpCircle
Text box	txt	txtLastName
Timer	tmr	tmrAlarm
UpDown	upd	updDirection

Vertical scroll bar	vsb	vsbRate

لا تعتبر تسمية الخاصية Name لعناصر التحكم وفق الجدول ٢-٣ من متطلبات فيجول بيسك. فمثلاً، كنا قد أطلقنا التسمية lblChoice على عنصر التحكم Label الذي يظهر معلومات حول خيار المستخدم. نستطيع إسناد القيمة Choice للخاصية Name لعنصر التحكم بدلاً من lblChoice، وبالتالي ستصبح آخر عبارة في الإجراء UpdateLabel كما يلي:

```
Choice.Caption = Info
```

بدلاً من:

```
lblChoice.Caption = Info
```

لاحظ أنّ تسمية عناصر التحكم وفق الجدول ٢-٣ تسهل قراءة البرنامج. فإذا نظرت مثلاً إلى العبارة:

```
Choice.Caption = Info
```

لن تكون أنت أو غيرك قادراً على التنبؤ بأن Choice عبارة عن عنصر تحكم Label. بل قد يعتقد من يقرأ هذه العبارة أنها تعديل لعنوان النموذج المسمى Choice. بينما تأمل العبارة:

```
lblChoice.Caption = Info
```

سيدرك من يقرأ هذه العبارة أنّ lblChoice هو اسم لافتة، وهذا بسبب وضع الأحرف lbl. وهكذا يصبح معنى العبارة "أسند قيمة المتحول Info إلى الخاصية Caption للافتة".

العبارات التي قد لا تتسع على سطر واحد في هذا الكتاب

العبارة يمكن إدخالها بحيث تمتد على أكثر من سطر واحد، فمثلاً العبارة:

```
MyVariable = 1 + 2 + 3
```

يمكن كتابتها بهذه الطريقة:

```
MyVariable = 1 + _
             2 + 3
```

أو بهذه الطريقة:

```
MyVariable = 1 _
             + 2 _
```

```
                    + 3
```

إذاً، نستطيع في فيجول بيسك استئناف كتابة العبارة على السطر التالي وذلك بترك فراغ في نهاية السطر، يليه الرمز (_) Underscore.

ملاحظة

لا تستطيع كتابة العبارة الواحدة على أسطر وأنت في منتصف سلسلة، فمثلاً العبارة التالية:

```
lblMyLabel.Caption = "This is my string "
```

لا يمكن كتابتها كالتالي:

```
lblMyLabel.Caption = "This is _
                     my string "
```

إذا كان لا بُد، فاكتبها كالتالي:

```
lblMyLabel.Caption = "This is _
                   & "my string "
```

الخلاصة

قفزنا مع هذا الفصل إلى الماء!. أجل لقد بدأنا فعلياً ببناء برامج ويندوز حقيقية تتضمن شريط تمرير ومربع نص ولافته وزر أمر وخانات أختيار وأزرار خيارات. فتعلمنا كيف نضع عناصر التحكم هذه فى برامج لغة فيجول بيسك، وكيفية تحديد قيمة الخاصية Value لهذه العناصر وكيف نضيف نص برنامج إلى الإجراءات المرافقة لعناصر االتحكم هذه. كما تعلمنا من هذا الفصل أيضاً ككيف نضيف إجراء إلى النموذج (عندما أضفنا الإجراء Update Label إلى النموذج FrmOpptioon).