الفصل السادس

تصميم واجهة التعامل User Imier Face

٦ـ٠ تمهيد

واجهة التعامل هى الشاشة/الشاشات التى يراها المستخدم، ويتعامل معها أثناء تشغيل البرنامج. وفى العادة تتكون واجهة التعامل من نافذة رئيسة، ومجموعة من أدوات التحكم؛ مثل أزرار الأوامر، صناديق النصوص، وهكذا. تعرف برامج Visual Basic التى تعمل على نظام التشغيل ويندوز باسم Windows Forms Applications أى تطبيقات نماذج النوافذ. وفيها يتم إنشاء واجهة التعامل باستخدام أدوات التحكم controls.

فى هذا الفصل، سوف نتعرف معا ـ عزيزى القارئ ـ على أشهر أدوات التحكم، وأكثرها استخداما.

الأهداف التعليمية

بنهاية هذا الفصل، يجب أن يكون الدارس قادرا على:

- معرفة أساسيات تصميم واجهة التعامل للبرنامج.
- استخدام أزرار الأوامر Buttons
- استخدام أداة العنوان وصندوق النص.

- كتابة كود الأحداث Even Handler
- استخدام صناديق وأزرار الاختيار.
- عرض الصور فى البرنامج.
- إنشاء القوائم Menus
- استخدام المؤقت Timer

٦ـ١ أساسيات واجهة التعامل

فى هذا الجزء، سوف نتعرف على ماهية واجهة التعامل، وأدوات التحكم، وكيفية إضافة أدوات التحكم إلى واجهة التعامل.

فى بدايات استخدام الحاسبات الشخصية، كان مستخدمو البرامج يتعاملون بشكل أساسى معها من خلال سطر الأوامر command line. حيث يبدأ البرنامج، ثم ينتظر حتى يتلقى مدخلات المستخدم، ثم يستكمل التنفيذ. أما أغلب البرامج التى نستخدمها هذه الأيام فهى تعمل فى بيئة النوافذ؛ حيث يتواصل ـ يتفاعل ـ المستخدم معها من خلال أزرار الأوامر وأشرطة القوائم، وهكذا. وسوف نتعرف معا فى هذا الجزء، وبقية هذا الفصل، على كيفية بناء واجهة تعامل فى بيئة النوافذ.

٦ـ١ـ١ استخدام النماذج Using Forms

تعتبر النماذج الجزء الأساسى فى واجهة التعامل. والنموذج عبارة عن نافذة تظهر لمستخدم البرنامج. وعلى هذا النموذج يقوم المبرمج بتصميم واجهة التعامل.

يتم إضافة أدوات التحكم إلى النماذج أثناء عملية تصميم واجهة التعامل. وأداة التحكم بمثابة كائن object ذى شكل معرف مسبقا، وله وظيفة أو غرض محدد. على سبل المثال: يستخدم زر الأمر Button لتنفيذ تعليمات معينة عند الضغط عليه بالفأرة، كما يستخدم صندوق النص TextBox لتلقى مدخلات المستخدم، فى حين يستخدم صندوق الصور PictureBox لعرض صورة معينة.

هناك أكثر من (٥٠) خمسين أداة تحكم فى لغة VB، كما تستطيع بنفسك إنشاء أدوات تحكم تعرف باسم User Controls. وسوف نتعرف أكثر على أدوات التحكم هذه فى الأجزاء التالية.

عند تصميم واجهة التعامل، يتم سحب drag أدوات التحكم من صندوق الأدوات Toolbox وإلقاؤها drop على سطح النموذج، ثم ضبط مكانها وحجمها حسب الرغبة. كما يمكن ضبط خصائص النموذج وأدوات التحكم عليه من خلال نافذة الخصائص Properties Window. على سبيل المثال: ضبط خاصية اللون الخلفى BackColor للنموذج.

تدريب عملي

١. من قائمة File اختر New Project

٢. اختر Windows Application واكتب اسم البرنامج FirstForm

٣. اضغط على النموذج مرة واحدة لاختياره.

٤. فى نافذة الخصائص، غير خاصية Text إلى My First Form واضغط Enter. تلاحظ تغير النص الموجود فى شريط عنوان النموذج.

٥. فى نافذة الخصائص أيضا، اذهب إلى خاصية BackColor لتغيير اللون الخلفى للنموذج حسبما تريد. يمكنك كذلك كتابة اللون المرغوب مباشرة، على سبيل المثال Red.

حاول التعرف بنفسك على بقية الخصائص الخاصة بالنموذج.

٦-١-٢ إضافة أدوات التحكم إلى النموذج

فى هذا الجزء، سوف نقوم بإضافة مجموعة من أدوات التحكم إلى النموذج، عن طريق اختيار الأداة من صندوق الأدوات Toolbox(عادة ما يوجد على يسار الشاشة)، ثم سحبها إلى سطح النموذج، وبعد ذلك نقوم بعملية ضبط خصائصها.

١. من صندوق الأدوات، اسحب كلا من الأدوات التالية، وضعهم على سطح النموذج: زر أمر Button، صندوق نص TextBox، أداة عنوان Label، صندوق اختيار CheckBox.

٢. اختر زر الأمر، واسحبه داخل النموذج لتغيير موقعه. لاحظ ـ عزيزى القارى ـ ظهور خطوط إرشادية تبين لك موضع زر الأمر مع الأدوات الأخرى الموجودة على سطح النموذج.

٣. كرر الخطوة السابقة مع بقية الأدوات؛ حتى تصل إلى الشكل المرغوب لواجهة التعامل.

٤. اختر زر الأمر، ثم اضغط على المربع الأبيض الصغير الموجود فى الناحية اليمنى أسفل الأداة، واسحبه لتغيير حجم زر الأمر.

٥. حاول ضبط بعض خصائص هذه الأدوات، مثل الخط Font، اللون الخلفى BackColor، اللون الأمامى ForeColor، والنص Text.

٦. اضغط F5 لتشغيل البرنامج. لاحظ أنه يمكنك الضغط على زر الأمر، وضع علامة وإزالتها داخل صندوق الاختيار CheckBox، وكتابة نص معين فى صندوق النص TextBox.

٦ـ٢ أزرار الأوامر Buttons

فى هذا الجزء، سوف نتعرف على كيفية إضافة أزرار الأوامر إلى النموذج، وكذلك كيفية تغيير شكل الزر، وكيفية كتابة الكود الذى يعمل عند الضغط على الزر.

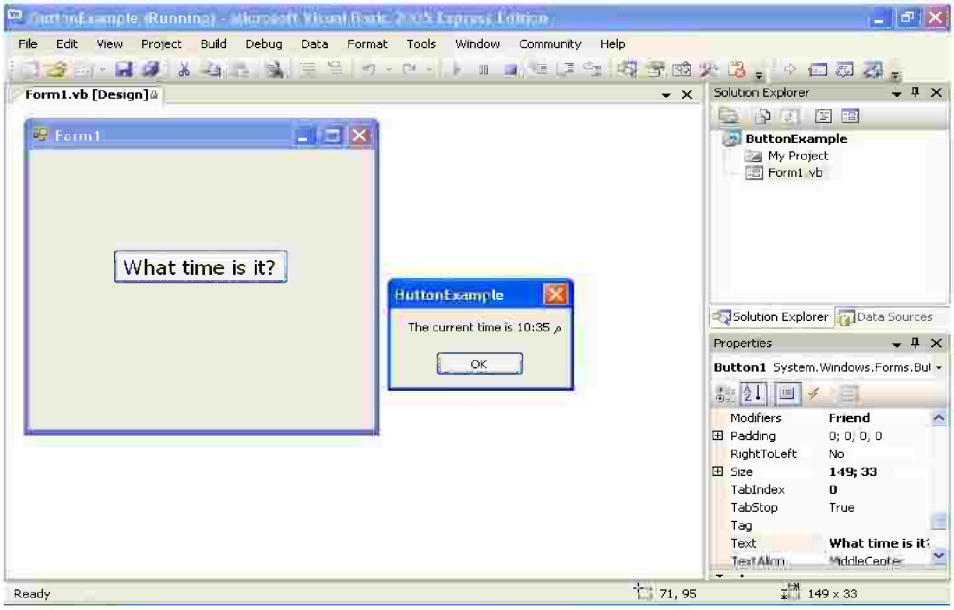
تعتبر أزرار الأوامر من أسهل الطرق التى يتعامل من خلالها المستخدم مع البرنامج. على سبيل المثال، يحتوى الكثير من البرامج على زر Exit للخروج من البرنامج.

هناك الكثير من الخصائص التى يمكن ضبطها بالنسبة إلى زر الأمر: أهمها خاصية النص Text؛ الذى يظهر على سطح الزر، وكذلك خاصية Font التى تحدد نوع الخط. هذا بالإضافة إلى خاصية BackColor بالنسبة إلى اللون الخلفى لزر الأمر، وخاصية ForeColor بالنسبة إلى لون النص.

عندما يضغط المستخدم على زر الأمر أثناء تشغيل البرنامج، يقوم الزر بإطلاق الحدث Click. وعندما يحدث ذلك، يتم تنفيذ الكود الخاص بهذا الحدث (والذى يحدده المبرمج).

تدريب عملي

١. من قائمة File اختر New Project
٢. اختر Windows Application واكتب اسم البرنامج ButtonExample
٣. من صندوق الأدوات Toolbox ضع زر أمر على سطح النموذج.
٤. من نافذة الخصائص، غير الخاصية Text إلى What time is it? واضغط Enter تلاحظ أن النص لا يناسب حجم الزر.



شكل (٦-١) استخدام زر الأمر Button فى واجهة التعامل

٥. من نافذة الخصائص، اضبط الخاصية AutoSize على True. الآن يتناسب النص مع حجم الزر.
٦. اضغط مرتين على زر الأمر لفتح نافذة محرر الكود Code Editor
٧. فى إجراء الحدث Button1_Click اكتب الكود التالى:

```
MsgBox("The current time is " & Now.ToShortTimeString)
```

٨. اضغط F5 لتشغيل البرنامج. اضغط على زر الأمر، تلاحظ ظهور رسالة تعرض الوقت الحالى.

٦ـ٣ أداة العنوان وصندوق النص

فى هذا الجزء، نتعرف على أداة العنوان Label التى تستخدم لعرض نص، كما نتعرف على أداة صندوق النص TextBox التى تستخدم للحصول على مدخلات المستخدم.

تعتبر النصوص text أحد أسهل الطرق لنقل المعلومات إلى، والحصول على بيانات من المستخدم. على سبيل المثال، يمكنك عرض معلومات عن وظيفة البرنامج، كما يمكنك الحصول على بيانات نصية من المستخدم، واستخدامها فى البرنامج.

٦ـ٣ـ١ عرض النصوص باستخدام Label

تعتبر الأداة Label هى الأداة الرئيسة لعرض النصوص فى برامج Visual Basic. من أهم الخصائص properties المرتبطة بهذه الأداة، الخاصية Text والخاصية Font والخاصية BackColor والخاصية ForeColor. وقد تعرفنا على وظيفة هذه الخصائص فى الجزء السابق: أزرار الأوامر Buttons.

٦ـ٣ـ٢ الحصول على بيانات باستخدام TextBox

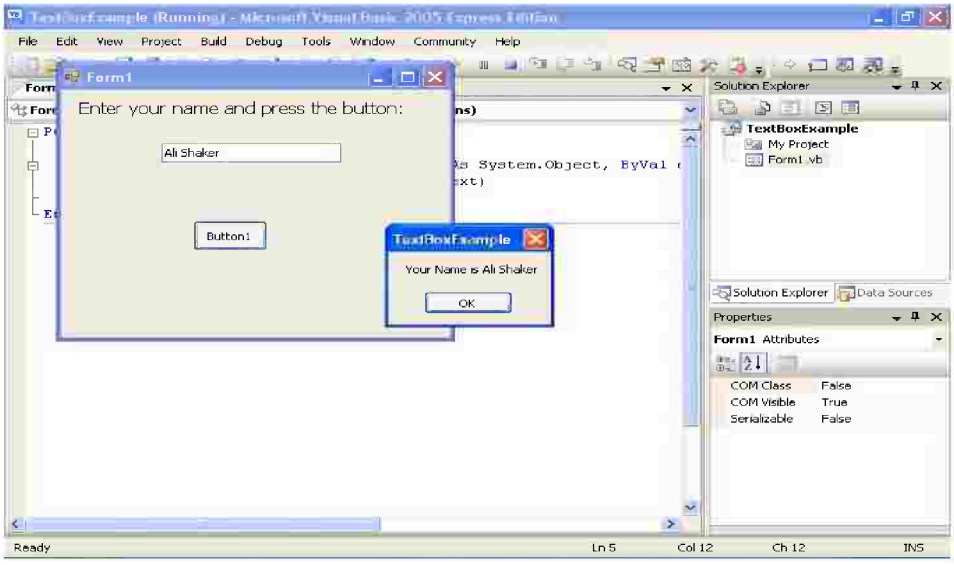
تسمح الأداة TextBox بالحصول على بيانات من المستخدم أثناء تشغيل البرنامج. ومن أهم الخصائص المرتبطة بها: الخاصية Text التى تمثل النص الذى يظهر فى الصندوق عند تشغيل البرنامج. وفى أغلب الأحيان، يترك المبرمج قيمة

هذه الخاصية فارغة؛ حتى يتمكن المستخدم من كتابة البيانات مباشرة. وكذلك هناك الخاصية Multiline والتى تسمح بوجود أسطر متعددة داخل صندوق النص. فإذا أردت الإفادة من هذه الخاصية، فعليك ضبط قيمتها على True.

تدريب عملى

١. من قائمة File اختر New Project
٢. اختر Windows Application واكتب اسم البرنامج TextBoxExample
٣. من صندوق الأدوات Toolbox ضع صندوق نص، وأداة عنوان، وزر أمر على سطح النموذج.
٤. اختر أداة العنوان، واسحبها حتى تصبح فوق صندوق النص.
٥. من نافذة الخصائص، اضبط خاصية Text لأداة العنوان، بحيث تصبح: Enter your name and press the button
٦. اضغط مرتين على زر الأمر لفتح نافذة محرر الكود Code Editor
٧. فى إجراء الحدث Button1_Click اكتب الكود التالى:

```
MsgBox("Your Name is " & Textbox1.Text)
```



شكل (٦-٢) استخدام صندوق النص وأداة العنوان فى واجهة التعامل

٨. اضغط F5 لتشغيل البرنامج.

٩. اكتب اسمك داخل صندوق النص، ثم اضغط زر الأمر، تلاحظ ظهور رسالة تعرض الاسم الذى كتبته فى صندوق النص.

٦ـ٤ كود الأحداث Even Handler

فى هذا الجزء، سوف نتعرف معا ـ عزيزى القارئ ـ على كيفية كتابة الكود الخاص بالاستجابة للأحداث فى أدوات التحكم.

لقد رأينا فى الأجزاء السابقة، أن أدوات التحكم تحتوى على خصائص properties، وطرق methods، وأحداث events، تستخدم جميعها أثناء واجهة التعامل وبناء البرنامج. والأحداث هى أشياء تحدث لأدوات التحكم. على سبيل المثال: الضغط Click على الأداة، أو كتابة نص داخل الأداة، أو حتى تمرير الفأرة MouseOver فوق الأداة، وهكذا.

وعندما تحدث هذه الأشياء، تقوم الأداة بإطلاق الحدث event؛ فيقوم البرنامج بالتحقق مما إذا كانت لديها الطرق methods الخاصة بالتعامل مع الحدث.

يمكنك كتابة كود أحداث للكثير من أدوات التحكم. سوف نتعرف الآن على الأحداث MouseEnter و MouseLeave الخاصة بزر الأمر Button، وهى أحداث تتعلق بحركة الفأرة فوق زر الأمر.

تدريب عملى

للتحكم فى الحدث MouseEnter:

١. من قائمة File اختر New Project

٢. اختر Windows Application واكتب اسم البرنامج EventHandler

٣. ضع زر أمر على سطح النموذج.

٤. من نافذة الخصائص، اضبط AutoSize على True

٥. من نافذة View اختر Code لفتح محرر الكود. يوجد صندوقان منسدلان أعلى نافذة الكود. يحتوى الصندوق الأيسر على قائمة بالأدوات controls الموجودة على النموذج، بالإضافة إلى Form1 وهو النموذج، و General وهو جزء يتعلق بالبرنامج ككل، و Form1 Events وهى أحداث النموذج. أما الصندوق الأيمن فيحتوى على قائمة بالأحداث المتاحة للأداة الحالية الموجودة فى الصندوق الأيسر.

٦. من الصندوق الأيسر، اختر Button1

٧. ومن الصندوق الأيمن، اختر MouseEnter

٨. اكتب الكود التالى فى إجراء الحدث Button1_MouseEnter:

```
Button1.Text = "The Mouse has entered"
```

٩. اضغط F5 لتشغيل البرنامج. حرك مؤشر الفأرة فوق زر الأمر. ولاحظ كيف يتغير النص الموجود على زر الأمر.

وللتحكم فى الحدث MouseLeave:

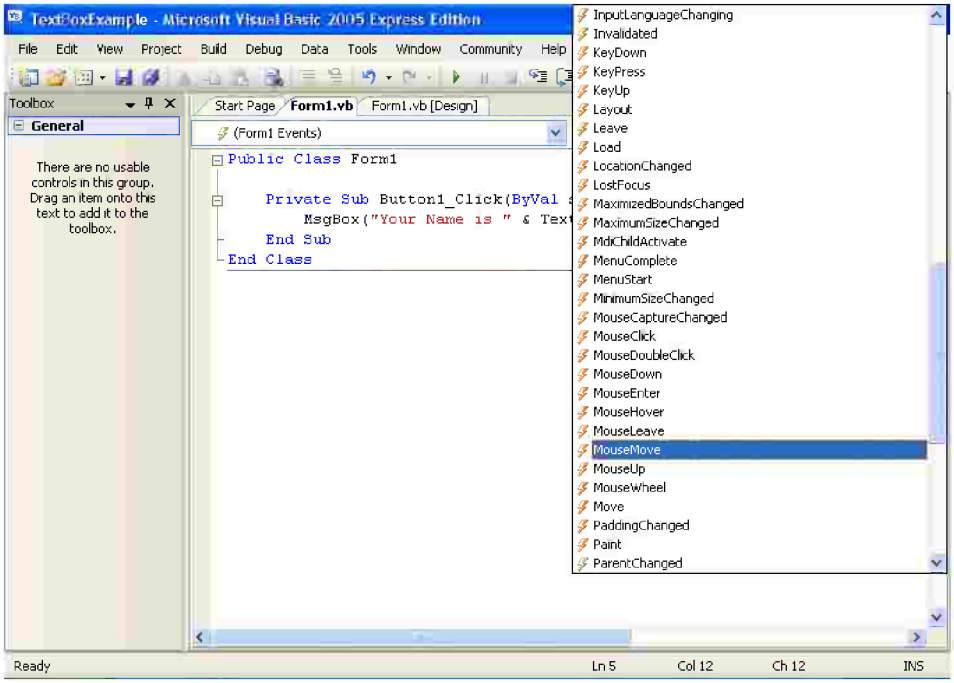
١. فى محرر الكود، تأكد أن Button1 هو العنصر الحالى داخل الصندوق الأيسر.، ثم اختر MouseLeave من الصندوق الأيمن.

٢. يظهر إجراء حدث جديد، هو Button1_MouseLeave، فى نافذة محرر الكود.

٣. اكتب الكود التالى داخل الحدث الجديد:

```
Button1.Text = "The mouse has left"
```

٤. اضغط F5 لتشغيل البرنامج. الآن عندما يتحرك مؤشر الفأرة فوق زر الأمر، يتغير النص إلى The mouse has entered، وعندما يتحرك المؤشر بعيدا، يتغير النص إلى The mouse has left.



شكل (٦-٣) قائمة الأحداث (الصندوق الأيمن) فى نافذة الكود

٦-٥ صناديق وأزرار الاختيار

فى هذا الجزء، سوف نتعرف معا على كيفية استخدام كل من صناديق الاختيار Check Boxes وأزرار الاختيار Radio Buttons ، وهما يستخدمان لتقديم مجموعة من الخيارات للمستخدم؛ للاختيار من بينها.

عند تصميم واجهة التعامل لبرنامج ما، غالبا ما يحتاج المبرمج إلى وسيلة يقدم بها مجموعة خيارات لكى يختار منها المستخدم. على سبيل المثال، افترض أنك تعد تطبيقا لتلقى الطلبات فى مطعم للبيتزا. فأنت تريد حتما وسيلة تمكن المستخدم من اختيار نوع البيتزا (أو مكوناتها). فى هذه الحالة يمكن استخدام صناديق الاختيار Check Boxes.

تتم عملية الاختيار عن طريق وضع علامة فى الصندوق المجاور لصندوق الاختيار. ولإزالة الاختيار، فقط نضغط على الصندوق مرة أخرى. ويمكن التعرف

على حالة الاختيار عن طريق الخاصية CheckBox.Checked. إذا كانت هناك علامة فى صندوق الاختيار، فإن قيمة هذه الخاصية تكون True، وبخلاف ذلك تكون القيمة False.

تدريب عملى

١. من قائمة File اختر New Project

٢. اختر Windows Application واكتب اسم البرنامج UserChoices

٣. ضع زر أمر على سطح النموذج، وكذلك ثلاثة صناديق اختيار.

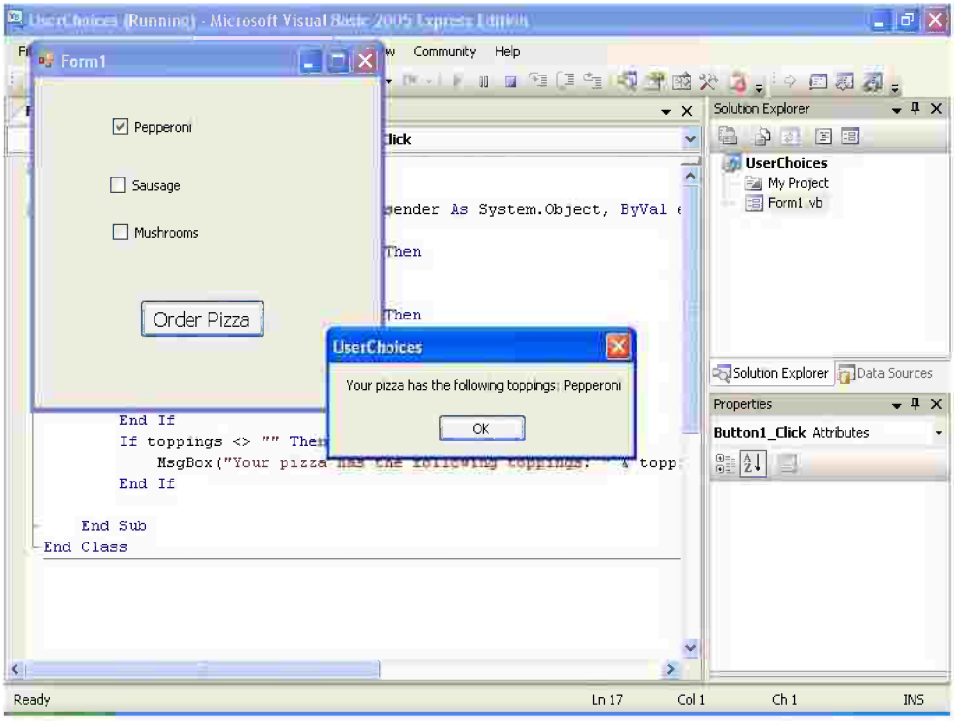
٤. من نافذة الخصائص، اضبط خاصية Text لكل صندوق من صناديق الاختيار على النحو التالى: Pepperoni, Sausage, and Mushrooms

٥. من نافذة الخصائص أيضا، غير الخاصية Text بالنسبة إلى زر الأمر إلى Order Pizza

٦. اكتب الكود التالى فى إجراء الحدث Button1_Click:

```
Dim toppings As String = ""
If CheckBox1.Checked = True Then
   toppings &= "Pepperoni "
End If
If CheckBox2.Checked = True Then
   toppings &= "Sausage "
End If
If CheckBox3.Checked = True Then
   toppings &= "Mushrooms"
End If
If toppings <> "" Then
   MsgBox("Your pizza has the following toppings: " & toppings)
End If
```

٧. اضغط F5 لتشغيل البرنامج. اختر أحد مكونات البيتزا من بين الاختيارات المتاحة، ثم اضغط زر الأمر. يظهر صندوق رسالة بها الاختيارات التى حددتها.



شكل (٦ـ٤) استخدام صناديق الاختيار CheckBox فى واجهة التعامل

٦ـ٥ـ١ أزرار الاختيار Radio Buttons

رأينا فى الفقرة السابقة، كيف أن صناديق الاختيار تسمح باختيار واحد أو كل الخيارات المتاحة. لكن ماذا لو أردت أن تسمح للمستخدم فقط باختيار أحد الخيارات المتاحة؟ فى هذه الحالة يجب استخدام أزرار الاختيار.

على العكس من صناديق الاختيار، تعمل أزرار الاختيار دائما فى نطاق مجموعة. وعندما يختار المستخدم أحد الخيارات، تلغى أية اختيارات أخرى. على سبيل المثال: قد تريد أن تسمح للمستخدم أن يختار إما صوصا عاديا وإماصوصا حارا فى البيتزا.

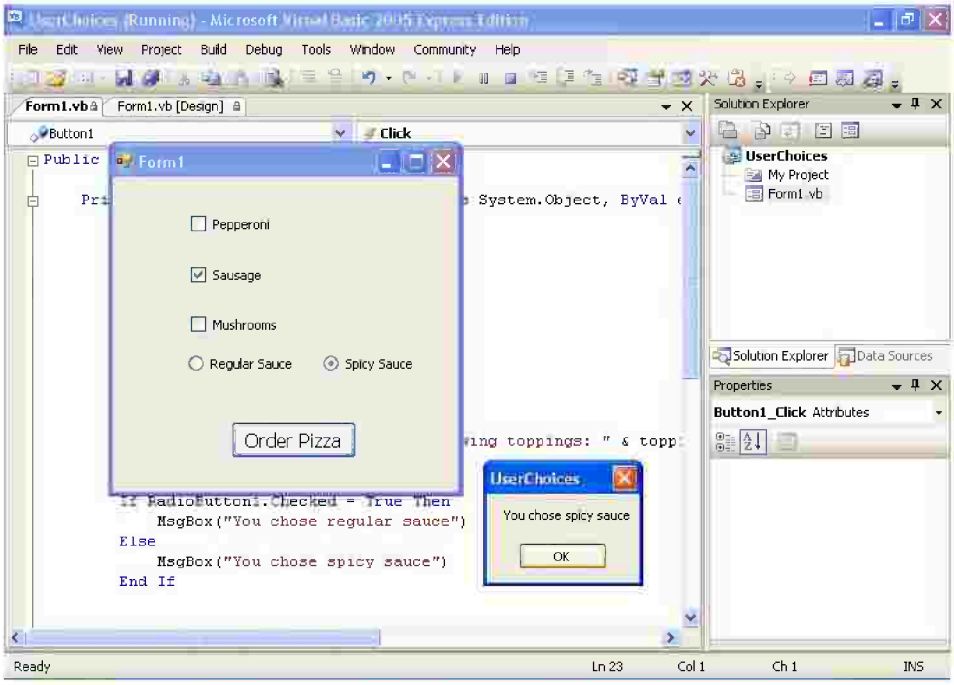
وكما كانت الحال فى صناديق الاختيار، يمكنك التعرف على حالة الاختيار عن طريق الخاصية RadioButton.Checked.

تدريب عملي

١. من صندوق الأدوات Toolbox، اسحب زرى اختيار على سطح النموذج.

٢. من نافذة الخصائص، غير خاصية Text بالنسبة إلى زر الاختيار الأول إلى Regular Sauce.

٣. اضبط خاصية Checked لزر الأمر الأول أيضا إلى True.



شكل (٦-٥) استخدام أزرار الاختيار RadioButton فى واجهة التعامل

٤. من نافذة الخصائص، غير خاصية Text بالنسبة إلى زر الاختيار الثانى إلى Spicy Sauce.

٥. اضغط مرتين على زر الأمر لفتح نافذة الكود.

٦. اكتب الكود التالى فى إجراء الحدث Button1_Click:

```
If RadioButton1.Checked = True Then
   MsgBox("You chose regular sauce")
Else
   MsgBox("You chose spicy sauce")
End If
```

٧. اضغط F5 لتشغيل البرنامج. اختر أحد أزرار الاختيار، ثم اضغط زر Order Pizza. تظهر رسالة توضح الاختيار الذى قمت به.

٦ـ٦ عرض الصور

فى هذا الجزء، سوف نتعرف على أداة التحكم PictureBox أو صندوق الصور، والذى يستخدم لعرض الصور؛ وذلك لعرض الصورة كخلفية background للنموذج.

من المعروف أن هناك قولا مأثورا إن الصورة تساوى ألف كلمة؛ وبالفعل هناك الكثير من البرامج التى تعتمد على الصور فى توصيل المعلومات للمستخدم. توجد طرق متعددة لعرض الصور فى لغة Visual Basic، لكن أشهرها هو الأداة PictureBox.

تعتبر الأداة PictureBox بمثابة حاوية للصور، ويتم اختيار الصورة التى تريد عرضها عن طريق ضبط الخاصية Image. يمكن ضبط الخاصية Image من خلال نافذة Properties، وكذلك يمكن تحديد الصورة من خلال الكود.

ومن بين الخصائص الأخرى المهمة بالنسبة إلى صندوق الصور، الأداة AutoSize التى تحدد ما إذا كان صندوق الصور سوف يتمدد لكى يناسب حجم الصورة. وهناك أيضا خاصية SizeMode التى يمكن أن تستخدم فى مد الصورة داخل جميع أجزاء الصندوق stretch أو جعلها فى وسط الصندوق center أو تقريبها Zoom. وقبل إضافة الصورة إلى الأداة PictureBox، يجب أولا إضافة الصورة إلى المشروع كمصدر resource. وبمجرد إضافة مصدر للمشروع، يمكنك إعادة استخدامها أى عدد من المرات.

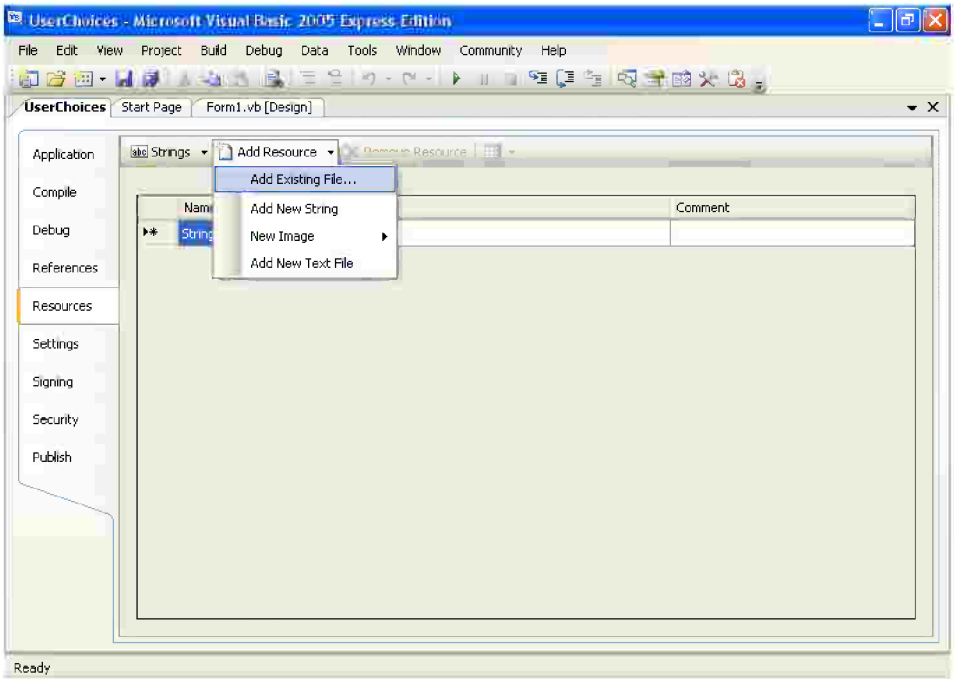
تدريب عملي

٦-٦-١ إضافة الصورة كمصدر فى المشروع

١. من قائمة File اختر New Project

٢. اختر Windows Application واكتب اسم المشروع Pictures

٣. فى نافذة مستكشف الحلول Solution Explorer اضغط مرتين على My Project وذلك لفتح مصمم المشروع Project Designer



شكل (٦-٦) مصمم المشروع Project Designer

٤. فى مصمم المشروع، افتح التبويب مصادر Resources

٥. اضغط على Add Resource لإضافة مصدر، واختر Add Existing File أى إضافة ملف موجود، وذلك من القائمة المنسدلة إلى أسفل.

٦. اختر ملف الصورة المطلوبة (الملفات ذات الامتداد bmp. أو gif. أو jpg.) ثم

اضغط على Open. وبذلك تكون قد أضفت الصورة إلى المشروع، وسوف تظهر فى نافذة Resource Manager.

٧. كرر الخطوات السابقة لإضافة صورة أخرى إلى المشروع.

٨. من قائمة File اختر Close واحفظ التغييرات بالضغط على Yes.

٦-٦-٢ عرض الصورة من خلال الأداة PictureBox

١. من مستكشف الحلول Solution Explorer اختر Form1.vb ثم اختر Designer من قائمة View.

٢. من صندوق الأدوات Toolbox اسحب الأداة PictureBox وضعها على سطح النموذج.

٣. من نافذة الخصائص، اضغط على زر ... الموجود بالخاصية Image؛ وذلك لفتح صندوق حوار Select Resource.

٤. من قائمة Entry اختر أيا من الصور التى أضفتها فى التدريب السابق، ثم اضغط على Ok.

٥. اختر الخاصية SizeMode واضبطها على AutoSize. لاحظ أن الصندوق يتمدد ليناسب حجم الصورة.

٦. من على النموذج، اضغط مرتين على صندوق الصورة لفتح نافذة الكود.

٧. اكتب الكود التالى فى إجراء الحدث PictureBox1_Click:

ملحوظة

يجب عليك استبدال MyPictureName2 باسم الصورة الثانية التى أضفتها فى التدريب السابق.

```
PictureBox1.Image = My.Resources.MyPictureName2
```

٨. اضغط F5 لتشغيل البرنامج. وعندما يظهر النموذج، اضغط على الصورة لعرض الصورة الثانية.

٦-٦-٣ عرض الصورة كخلفية النموذج

بالإضافة إلى عرض الصورة فى الأداة PictureBox، يمكنك، أيضا عرض الصورة كخلفية Background للنموذج. وهنا تستخدم الخاصية BackgroundImag الخاصة بالنموذج؛ لعرض الصورة بحيث تظهر خلف جميع أدوات التحكم على سطح النموذج، تماما مثل ورقة الحائط wallpaper بالنسبة إلى سطح المكتب windows desktop.

وكما هى الحال فى نظام تشغيل Windows حين يمكنك اختيار ما إذا كانت ورقة الحائط فى الوسط centered أو متكررة فى جميع أجزاء سطح المكتب tiled، أو ممتدة stretched بحيث تملأ الشاشة، تستخدم الخاصية BackgoundImageLayout لذات الغرض بالنسبة إلى النموذج.

ملحوظة

الكثير من أدوات التحكم الأخرى – مثل Panel، GroupBox، بل حتى زر الأمر Button – لها أيضا الخاصية BackgroundImage.

تدريب عملي

١. فى مستكشف الحلول Solution Explorer اختر Form1.vb ثم اختر Designer من قائمة View.

٢. اختر النموذج، عن طريق الضغط على أى مكان خارج صندوق الصورة.

٣. من نافذة الخصائص، اضغط على الزر ... الموجود بجوار الخاصية BackgroundImage؛ وذلك لفتح صندوق حوار Select Resource.

٤. من قائمة Entry اختر صورة من الصور التى أضفتها فى التدريب السابق، ثم اضغط Ok، تلاحظ أن الصورة تعرض على النموذج خلف صندوق الصور، وبشكل tiled أى متكررة، وهذا هو الوضع الافتراضى.

٥. اضبط الخاصية BackgroundImageLayout على Stretch أى الوضع الممتد، تلاحظ أن الصورة تتمدد لتملأ كامل مساحة النموذج.

٦. اضغط مرتين على النموذج لفتح نافذة الكود.

٧. تأكد من أن Form1 Events هو الموجود فى الصندوق الأيسر أعلى نافذة الكود، ثم اختر Click من الصندوق الأيمن.

٨. اكتب الكود التالى فى إجراء الحدث Form1_Click:

```
If Me.BackgroundImageLayout = ImageLayout.Stretch Then
  Me.BackgroundImageLayout = ImageLayout.Center
Else
  Me.BackgroundImageLayout = ImageLayout.Stretch
End If
```

٩. اضغط F5 لتشغيل البرنامج. وعندما يظهر النموذج، اضغط عليه لتغيير وضعية الصورة.

٦ـ٧ القوائم Menus

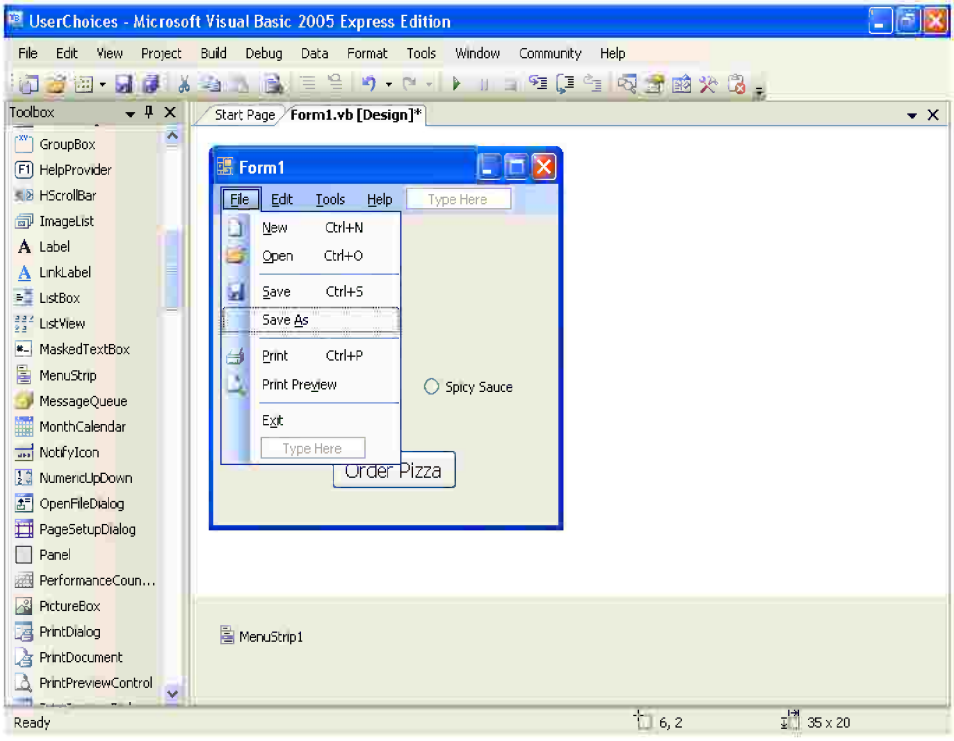
فى هذا الجزء، سوف نتعرف معا ـ عزيزى القارئ ـ على كيفية إنشاء القوائم، وكذلك كتابة الكود الخاص بتنفيذ الأوامر المختلفة الموجودة بالقائمة.

تعتبر القوائم من الطرق السهلة والشائعة بالنسبة إلى المستخدم عند التعامل مع وظائف البرنامج المختلفة. تشتمل الخيارات الشائعة للقوائم على ضبط خيارات البرنامج، والمهام التحريرية مثل القص Cut والنسخ Paste، وكذلك فتح وحفظ الملفات.

تسهل لغة Visual Basic عملية إنشاء القوائم؛ حيث يمكنك استخدام الأداة MenuStrip أو شريط القوائم، فى تحرير القائمة بسهولة ويسر. وعند سحبها على سطح النموذج، تظهر الأداة MenuStrip كصندوق يحتوى على العبارة " Type

Here" وتوجد فى الجزء العلوى من النموذج. يمكنك الضغط click على هذا الصندوق وكتابة عناوين عناصر القائمة المختلفة بداخله.

عند ضبط عنوان أحد عناصر القائمة، يمكنك إعداد العناصر الفرعية التى توجد تظهر عند فتح كل قائمة. وعند الانتهاء من القائمة، يمكنك كتابة الكود الذى يتم تنفيذه؛ عندما يضغط المستخدم على كل عنصر من عناصر القائمة.



شكل (٦-٧) استخدام شريط القائمة MenuStrip فى واجهة التعامل

تدريب عملى

١. من قائمة File اختر New Project

٢. اختر Windows Application واكتب اسم البرنامج Menus

٣. من صندوق الأدوات Toolbox اسحب MenuStrip على سطح النموذج.

بصرف النظر عن المكان الذى وضعتها فيه، سوف تظهر الأداة فى أعلى النموذج. وربما تكون قد لاحظت ـ عزيزى القارئ ـ أنه تمت إضافة MenuStrip1 إلى المنطقة الرمادية أسفل النموذج، والتى يطلق عليها Component tray.

٤. اضغط على MenuStrip الموجود على سطح النموذج، واكتب File ثم اضغط Enter. تظهر صناديق إضافية أسفل لكتابة عناصر القائمة الفرعية. يمكنك الاستمرار فى كتابة بقية العناصر حتى يكتمل شريط القائمة.

٥. فى الصندوق الموجود تحت العنصر الأول (File) اكتب Exit ثم اضغط Enter

٦. اضغط مرتين على العنصر Exit لفتح نافذة الكود.

٧. اكتب الكود التالى فى إجراء الحدث ExitToolStripMenuItem_Click:

```
Application.Exit( )
```

٨. اضغط F5 لتشغيل البرنامج. عن طريق الفأرة، افتح قائمة File، ثم اضغط على Exit، فسوف يغلق البرنامج.

٨ـ٦ المؤقت Timer

فى هذا الجزء، سوف نتعرف على كيفية استخدام المؤقت Timer لتنفيذ تعليمات برمجية معينة دون التدخل من المستخدم. ففى بعض الأحيان، قد ترى أنه من الضرورى تكرار تنفيذ بعض المهام فى البرنامج، على سبيل المثال: حفظ الملف كل ثلاث دقائق، أو تحديث واجهة التعامل.

يختلف المؤقت Timer عن غيره من أدوات التحكم controls التى استخدمناها حتى الآن؛ حيث لا يوجد له كيان مرئى على سطح النموذج أثناء تشغيل البرنامج. يطلق على أدوات التحكم controls التى ليس لها كيان مرئى اسم components أى مكونات. ولا توجد طريقة يتعامل بها المستخدم مباشرة مع المؤقت؛ لأنها تعمل فى الخلفية عند تشغيل البرنامج.

يحتوى المؤقت على خاصيتين properties 2 وحدث واحد One event هما الأكثر استخداما. فالخاصية Enabled تحدد ما إذا كان المؤقت يعمل. فإذا كانت قيمتها True كان المؤقت نشطا، أما إذا كانت قيمتها False، فإن المؤقت يكون غير نشط.

أما الخاصية Interval فتحدد عدد الميلى ثانية (١٠٠٠ ميلى ثانية تساوى ثانية واحدة) قبل انطلاق الحدث event المحدد داخل المؤقت.

ينطلق الحدث Tick عن طريق المؤقت على فترات intervals متساوية، على حسب القيمة الموجودة فى الخاصية Interval. يمكنك إضافة الكود إلى الحدث Timer.Tick وسوف يتم تنفيذه عند انطلاق الحدث Tick.

تدريب عملى

١. من قائمة File اختر New Project

٢. اختر Windows Application واكتب اسم البرنامج Timer

٣. من صندوق الأدوات Toolbox ضع أداة عنوان Label ومؤقت Timer على سطح النموذج. لن يظهر المؤقت على سطح النموذج، ولكن فى المنطقة الرمادية Components tray أسفل النموذج؛ لأنه لا يحتوى على كيان مرئى.

٤. اختر المؤقت، ومن نافذة الخصائص، اضبط الخاصية Enabled على True. وكذلك اضبط الخاصية Interval على ١٠٠٠.

٥. اضغط مرتين على المؤقت لفتح نافذة الكود.

٦. اكتب الكود التالى فى إجراء الحدث Timer1_Tick:

```
Label1.Text= My.Computer.Clock.LocalTime.ToLongTimeString
```

٧. اضغط F5 لتشغيل البرنامج. تلاحظ أن النص على أداة العنوان يتغير كل ثانية؛ ليعرض التوقيت الحالى.

إلى هنا نكون قد وصلنا معك ـ عزيزى القارئ ـ إلى نهاية الفصل السادس، والذى تعرفنا فيه على أساسيات تصميم واجهة التعامل User interface. فى الفصل التالى، نستكشف معا معالم أحد أهم تطبيقات البرمجة، إن لم يكن أهمها على الإطلاق، ألا وهو برمجة قواعد البيانات.

ملخص

فى هذا الفصل، ألقينا الضوء معا ـ عزيزى القارئ ـ على الخطوة الأولى فى بناء برامج لغة Visual Basic وهى تصميم واجهة التعامل User Interface.

وكان التركيز على أداوت التحكم الأساسية الأكثر استخداما فى بناء تطبيقات الويندوز: زر الأمر Button الذى يستخدم لتنفيذ تعليمات برمجية معينة، من خلال الضغط Click عليه أثناء تشغيل البرنامج. كما تعرفنا على استخدام صناديق النصوص TextBox وأداة العنوان Label؛ حيث تستخدم الأولى فى الحصول على مدخلات من المستخدم، أما الثانية فتستخدم بشكل أساسى لعرض معلومات نصية على واجهة التعامل.

ولقد تعرفنا كذلك على استخدام صناديق الاختيار CheckBox وأزرار الاختيار RadioButton. أما صناديق الاختيار فتستخدم لتمكين المستخدم من عمل اختيار واحد أو اختيارات متعددة فى الوقت ذاته، فى حين تمكن أزرار الاختيار من عمل اختيار واحد فقط بين عدة بدائل.

بالإضافة إلى ما سبق، رأينا ـ عزيزى القارئ ـ كيفية إدراج شريط القوائم MenuStrip، وكذا استخدام صندوق الصور PictureBox لعرض صورة داخل واجهة التعامل، أو استخدامها كخلفية Background للنموذج. وأخيرا تعرفنا على استخدام أداة المؤقت Timer لتنفيذ تعليمات معينة على فترات Intervals زمنية متساوية.

تمارين

١. ما أهم الخصائص Properties المرتبطة بزر الأمر Button؟.

٢. وضح أهمية استخدام صندوق النص TextBox وأداة العنوان Label فى واجهة التعامل؟.

٣. قارن بين الأداة CheckBox والأداة RadioButton عند تقديم اختيارات معينة للمستخدم.

٤. يطلق على الأداة الخاصة بالقوائم Menus اسم (اختر واحدة):

a. MenuEditor

b. MenuBar

c. MenuStrip

٥. تستخدم أداة PictureBox فى:

a. ---------- و

b. ----------.