الفصل الثامن عشر
تحسين الواجهات بالرسوم

نشرح في هذا الفصل تحسين واجهة المستخدم (النماذج والنوافذ) بالخطوط والنقاط والدوائر والصور المختلفة سواءً باستخدام الوظائف التى تقوم برسم الخطوط والدوائر والمنحنيات أو بإظهار الصور من خلال الأداة **Picture Box** ونوضح فوائد ومساوئ كل طريقة.

بانتهاء هذا الفصل سنتعرف علي:

- تحسين واجهة المستخدم بالرسوم
- عرض الصور على الأدوات والنماذج
- استخدام أداة مربع الصورة **Picture Box**
- تعيين خلفية الأداة
- إنشاء الرسوم أثناء التشغيل
- استخدام الأقلام وفرش الرسم والألوان
- رسم الخطوط والأشكال

تحسين واجهة المستخدم بالرسوم

واجهة المستخدم النمطية عبارة عن عدة نوافذ بها قوائم ومربعات نصوص وعناوين وأزرار أوامر وما إلى ذلك. ولكن لو اكتفينا بذلك لصارت هذه الواجهة مملة وغير جذابة. إدخال الرسوم إلى الواجهة يحسن الواجهة من النواحي التالية:

- لفت الانتباه إلى معلومات معينة على الشاشة
- تستخدم لعرض المعلومات بطرق مختلفة (طريقة رسومية)
- تجعل فهم وظيفة العناصر في كثير من الأحيان بديهية وسهلة

هذه التحسينات تتم كما سنرى من خلال طريقتين مختلفتين هما:

- إضافة الصور إلى أدوات التحكم التى تدعم أساساً استخدام الصور مثل أداة مربع الصور **Picture Box** أو زر الأمر أو حتى النموذج نفسه.
- استخدام وظائف الرسم المختلفة الموجودة بتصنيف الرسوم **Graphics** داخل **.NET**.

وسوف نتعرف على كلتا الطريقتين من خلال هذا الفصل.

تحتوى الإصدارات التى تسبق **Visual Basic.Net** مثل **Visual Basic 6.0** على أداتى التحكم **Line** و**Shape** التى يمكن من خلالها رسم الخطوط والأشكال المختلفة كالمستطيلات والدوائر والمربعات ... الخ. أما **Visual Basic 2012** فلم يعد يدعم أياً من الأداتين. وإنما يتم رسم هذه الأشكال من خلال الوظائف فقط.

عرض الصور على الأدوات والنماذج

يدعم الكثير من أدوات التحكم الموجودة فى **Visual Basic 2012** بما فى ذلك النماذج تضمين الصور ذات التنسيقات المختلفة وذلك من خلال خاصيتين مشتركتين بين جميع هذه الأدوات وهما الخاصية **Image** والخاصية **Back ground Image** التى يمكنك تخصيص صورة لأيٍ منها، حيث يمكنك استخدام أحد تنسيقات الصور الموضحة فى جدول ١٨-١

التالى. ونقصد بالصور Pictures الرسوم النقطية Bitmaps التي تحتوى رسوم فنية أو مخططات توضيحية أو صور فوتوغرافية. يمكن لهذه الصور أيضاً أن تحمل بعض المعلومات غير تحسين المظهر. مثال لذلك الصور التي تقوم برامج الإعداد بعرضها لتسليتك أثناء الإعداد حيث تقوم بعرض خصائص البرنامج الذي يتم إعداده.

جدول ١٨-١ أنواع ملفات الرسوم التي يعرضها Visual Basic

امتداد الملف	نوعية الملف
.BMP	ملف صور نقطي Windows bitmap
.ICO	ملف رمز Icon
.WMF	ملف أوامر رسم Windows Metafile
.EMF	ملف أوامر رسم محسن Enhanced Metafile
.GIF	الاسم اختصار لعبارة Graphics Interchange Format و هو نوع من الملفات يستخدم بكثرة على الإنترنت و هو نقطي أيضاً
.JPG و.JPEG	الاسم اختصار لعبارة Joint Photographic Experts Group وهي المجموعة التي ابتكرت هذا النوع من الملفات . وهو نوع يستخدم مع الملفات ذات الألوان الحقيقية ، لكن به أسلوب للضغط يوفر في حجم الملف – وهو نوع منتشر أيضاً على الإنترنت
.PNG	وهو اختصار للعبارة Portable Network Graphics ويعد امتداد للتنسيق GIF ويجمع بين كفاءة الصورة وقلة حجم الملف كما أنه أحد التنسيقات الجديدة المستخدمة على الإنترنت

استخدام الخاصية Image

يمكنك من خلال الخاصية Image تعيين الصورة التى ترغب فى إضافتها للأداة حيث تُدعم هذه الخاصية من قِبَل أدوات تحكم مربع الصورة PictureBox وزر الأمر Button وأداة العنوان Label وعنوان الارتباط LinkLabel ومربع الاختيار CheckBox وأخيراً زر

الاختيار **Radio Button**. وهذه الخاصية كثيرة الاستخدام مع أداة مربع الصورة التى تم تصميمها خصيصاً لاحتواء الصور والرسوم. أما الأدوات الأخرى فتستخدم هذه الخاصية فى حالة الرغبة فى تدعيم نصوصها بالصور أو الاستغناء أساساً عن النصوص. فزر الأمر الذى يقوم بإجراء أمر الطباعة على الطابعة، من الممكن أن يحتوى على صورة للطابعة بجانب أو بدون كلمة "طباعة" أو **Print**. يوضح شكل ١٨-١ استخدام الخاصية **Image** مع كل أداة من أدوات التحكم التى تدعم هذه الخاصية.

شكل ١٨-١ استخدام الخاصية **Image** لتضمين الصور مع أدوات التحكم.

تعيين الخاصية Image أثناء التصميم

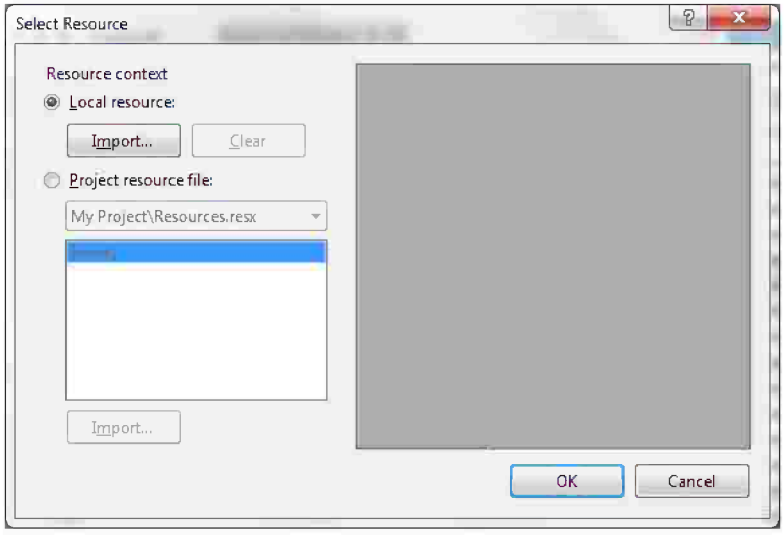
يمكنك تعيين قيمة الخاصية **Image** أثناء التصميم أو حتى أثناء التشغيل من خلال الكود. لتعيين قيمة الخاصية، أثناء التصميم، تابع معنا الخطوات الآتية:

١. اختر الأداة التى ترغب فى تعيين قيمة الخاصية لها داخل نافذة التصميم.

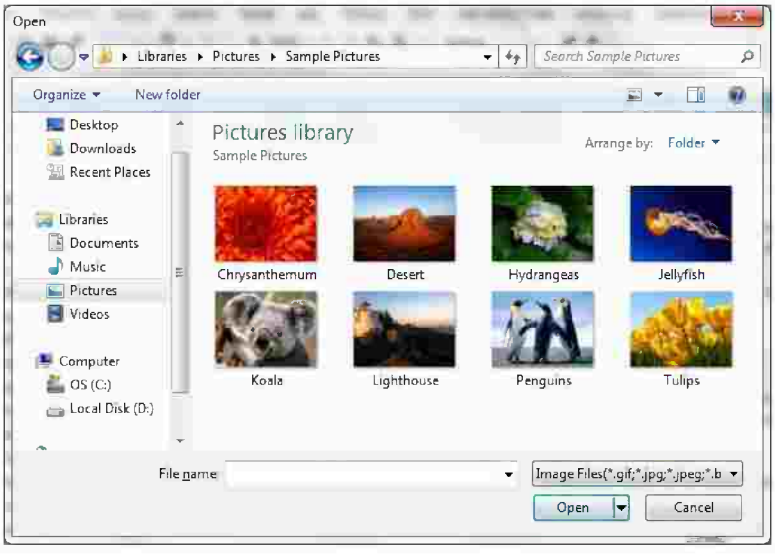
٢. انقر الزر ... المجاور للخاصية **Image** داخل المجموعة **Appearance** بمربع الخصائص، يظهر المربع الحوارى **Select Resource** (انظر شكل ١٨-٢).

٣. يحتوى المربع الحوارى **Select Resource** على جزأين، الأول **Local Resource** ويتم من خلاله تحديد إحدى الصور الموجودة على الحاسب أو المتصلة بالشبكة، والثانى **Project resource file** ويتم من خلاله اختيار إحدى الصور

التى تم بالفعل إضافتها إلى المشروع من قبل. نشط زر الاختيار المناسب ثم انقر زر **Import**، يظهر المربع الحوارى **Open** المعتاد (انظر شكل ١٨-٣).



شكل ١٨-٢ المربع الحوارى **Select Resource**.



شكل ١٨-٣ اختيار الصورة المناسبة من المربع الحوارى **Open**.

٤. اختر الصورة التى ترغب فى تضمينها ثم انقر زر **Open**، يتم إضافة الصورة إلى أداة التحكم.

حينما تستخدم الخاصية **Image** مع إحدى الأدوات، يمكنك أيضاً استخدام الخاصية **ImageAlign** التى تحدد محاذاة الصورة داخل الأداة حيث يوجد تسع قيم مختلفة وهى القيم الموضحة بجدول ١٨-٢ التالى.

جدول ١٨-٢ قيم الخاصية**ImageAlign**

القيمة	وضع الصورة
TopLeft	الركن الأيسر العلوى من الأداة
TopCenter	منتصف أعلى الأداة
TopRight	الركن الأيمن العلوى من الأداة
MiddleLeft	منتصف الحافة اليسرى من الأداة
MiddleCenter	منتصف الأداة (القيمة الافتراضية)
MiddleRight	منتصف الحافة اليمنى من الأداة
BottomLeft	الركن الأيسر السفلى من الأداة
BottomCenter	منتصف الحافة السفلية من الأداة
BottomRight	الركن الأيمن السفلى من الأداة

إذا قمت بإضافة صورة إلى أداة تحكم تحتوى أساساً على عنوان أو نص (مثل زر الأمر **Button**)، يمكنك استخدام الخاصية **TextAlign** لتحديد محاذاة العنوان على الأداة إلى جانب استخدام الخاصية **ImageAlign** لتحديد محاذاة الصورة على الأداة.

تعيين الخاصية Image وقت التشغيل

كما يمكنك تعيين قيمة الخاصية **Image** أثناء التصميم من خلال اختيار إحدى الصور من المربع الحوارى **Open** أو المربع الحوارى **Select Resource**، يمكنك أيضاً تعيين قيمتها أثناء التشغيل. فإذا أردت مثلاً تعيين قيمة الخاصية لأداة مربع الصور **PictureBox1**، قم باستخدام العبارة التالية:

```
PictureBox1.Image = Image.FromFile("c:\Mohamed.gif")
```

وإذا أردت فى أى وقت حذف هذه الصورة، قم بتخصيص القيمة **Nothing** للخاصية **Image** كما يلى:

```
PictureBox1.Image = Nothing
```

لكن ربما تتساءل الآن عن أى الطريقتين أفضل، تعيين الصور أثناء مرحلة التصميم أم تعيينها أثناء التشغيل ؟. فى الواقع هناك بعض الاعتبارات التى يجب أن تضعها فى خلدك قبل اتخاذ القرار. فحينما تقوم بتعيين الصورة فى مرحلة التصميم، يتم تضمين الصورة برمتها داخل تطبيقك ومن ثم يتم تضمينها أثناء ترجمة هذا التطبيق. وعلى هذا كلما زاد عدد الصور كلما زاد حجم التطبيق. أما إذا قمت بتحميل الصور وقت التشغيل من خلال الكود، فلا يتم ترجمة هذه الصور داخل التطبيق كما أنك تستطيع تغيير هذه الصور كيفما شئت. ولكن كن حذراً، فإذا لم يجد البرنامج الصورة المحددة أثناء التشغيل، يقوم على الفور بإظهار رسالة الخطأ المناسبة وعليك فى هذه الحالة احتواء هذا الخطأ حتى لا يؤدى إلى توقف التطبيق بالكامل.

استخدام أداة مربع الصور PictureBox

يستخدم مربع الصورة فى عرض الصور والرسوم المختلفة على النماذج، مثل الصورة الطبيعية الموجودة بشكل ١٨-١ السابق. ولأن هذا المربع هو حاوى الصورة، فيمكنك التحكم فى موقع الصورة على النموذج ببساطة شديدة من خلال تعيين قيمة الخاصية **Location** الخاصة بمربع الصورة. ومن الضرورى أيضاً تعيين قيمة الخاصية **SizeMode** التى تحدد كيفية تحجيم الصورة داخل المربع أو تحجيم المربع كى يظهر الصورة بالكامل، فغالباً

ما يختلف حجم الصورة عن حجم مربع الصورة المستخدم. يوضح جدول ١٨-٣ التالى القيم المتاحة للخاصية SizeMode وتأثير كل منها.

جدول ١٨-٣ قيم الخاصية SizeMode داخل مربع الصورة PictureBox

القيمة	التأثير
Normal	لا يتم إعادة تحجيم الصورة أو مربع الصورة. فإذا كانت الصورة أكبر من المربع، يتم قص الصورة. أما إذا كان المربع أكبر من الصورة، فيتم وضع الصورة بالركن الأيسر العلوى من المربع.
AutoSize	يتم إعادة تحجيم المربع كى يتوافق مع حجم الصورة.
StretchImage	يتم إعادة تحجيم الصورة كى تتوافق مع حجم المربع.
CenterImage	إذا كان المربع أكبر من حجم الصورة، يتم توسيط الصورة داخل المربع. أما إذا كانت الصورة أكبر من المربع، فيتم توسيط الصورة داخل المربع وقص الأطراف الزائدة.
Zoom	يتم زيادة حجم الصورة أو إنقاصه مع الحفاظ على النسبة الموجودة بحجم الصورة (نسبة الطول إلى العرض)

تحتوى الإصدارات التى تسبق Visual Basic.Net على الخاصية Picture الخاصة بالنماذج والتى تمكِّنك من وضع صورة بالركن الأيسر العلوى من النموذج دون إمكانية إعادة تحجيم الصورة. يمكنك تطبيق هذه الوظيفة داخل Visual Basic 2012 عن طريق وضع الصورة المطلوبة داخل مربع الصورة PictureBox ثم وضع المربع نفسه فى المكان المناسب.

يوضح شكل ١٨-٤التالى تأثير كل قيمة من القيم الخمسة على ظهور الصورة داخل مربع الصور.

شكل ١٨-٤ تأثير الخاصية **SizeMode** على ظهور الصورة داخل مربع الصورة.

تعيين خلفية الأداة

يحتوى العديد من أدوات التحكم بما فى ذلك النماذج نفسها على الخاصية **BackgroundImage** التى تستخدم لتعيين صورة كخلفية للأداة أو النموذج. وهذه الأدوات هى زر الأمر **Button** ومربع الاختيار **CheckBox** وزر الاختيار **RadioButton** ومربع المجموعة **GroupBox** ومربع الصورة **PictureBox** وأخيراً اللوحة **Panel**. وعادةً تستخدم صور الخلفية فى تزيين مظهر الأداة أو النموذج بعكس الصورة **Image** التى تستخدم دائماً لتوضيح وظيفة الأداة (انظر شكل ١٨-٥).

إذا كانت صورة الخاصية **BackgroundImage** أصغر من الأداة المستخدمة، تتمدد الصورة أفقياً ورأسياً كى تملأ الأداة بالكامل.

شكل ١٨-٥ إضافة خلفية إلى النموذج.

يمكنك تعيين قيمة الخاصية **BackgroundImage** فى مرحلة التصميم أو أثناء التشغيل كما يمكنك حذف قيمة هذه الخاصية أثناء التشغيل باتباع نفس الخطوات المستخدمة مع الخاصية **Image**.

إنشاء الرسوم أثناء التشغيل

ناقشنا في الأجزاء السابقة من الفصل إظهار الصور الجاهزة المعدة مسبقاً من برنامج للرسم ولكن **Visual Basic** لا تقف قدرته عند هذا الحد، بل من الممكن تعديل وإنشاء هذه الرسوم بالأوامر من خلال الكود. تحتاج ذلك مثلاً عند عرض مخططات بيانية لبيانات شركة مثلاً. ولأن البيانات تتغير فيصعب إنشاء صور جاهزة لها، ويلزم أن ينشئها البرنامج أثناء تشغيله.

يتم إنشاء الرسوم في **Visual Basic** أساساً بوظائف الرسم الخاصة بالنافذة ومربع الصور بالإضافة إلى استخدام الكائن **Printer** في الطباعة على الطابعة المتاحة في **Windows**.

هناك خمس وظائف أساسية لإنشاء الرسوم وهي:

- الوظيفة **DrawLine()** وتستخدم لرسم خط مستقيم بين نقطتين.
- الوظيفة **DrawRectangle()** وتستخدم لرسم مربع أو مستطيل.
- الوظيفة **DrawEllipse()** وتستخدم لرسم دائرة أو شكل بيضاوى.

- الوظيفة **DrawPolygon()** وتستخدم لرسم شكل مغلق يحتوى على عدد من الجوانب (مضلع).
- الوظيفة **Clear()** وتستخدم فى إزالة الرسوم الموجودة على أحد الكائنات.

وفيما يلى نوضح الصيغة العامة لكل وظيفة من هذه الوظائف.

الوظيفة DrawLine()

تقوم هذه الوظيفة برسم خط بين نقطتين وتحتوى على الصيغة العامة التالية:

```
control.CreateGraphics.DrawLine(Pen, x1, y1, x2, y2)
```

حيث:

- تعبر **control** عن الأداة أو النموذج الذى يتم رسم الخط بداخله
- يعبر **Pen** عن القلم المستخدم فى عملية الرسم والذى يتم تحديده مسبقاً
- تعبر **x1,y1** عن إحداثى نقطة بداية الخط
- تعبر **x2,y2** عن إحداثى نقطة نهاية الخط

الوظيفة DrawRectangle()

تقوم هذه الوظيفة برسم مستطيل محدد بزوج من الإحداثيات وعرض وطول وتحتوى على الصيغة التالية:

```
control.CreateGraphics.DrawRectangle(Pen, x, y, width, height)
```

حيث:

- تعبر **control** عن الأداة أو النموذج الذى يتم رسم المستطيل بداخله
- يعبر **Pen** عن القلم المستخدم فى عملية الرسم والذى يتم تحديده مسبقاً
- تعبر **x,y** عن إحداثى الركن الأيسر العلوى من المستطيل
- تعبر **Width** عن عرض المستطيل
- تعبر **Height** عن ارتفاع المستطيل

الوظيفة DrawEllipse()

تقوم هذه الوظيفة برسم قطع ناقص عن طريق تعريف المستطيل المحيط به والذى يحتوى على إحداثى الركن الأيسر العلوى والعرض والطول كما سبق، لذا فهى تحتوى على الصيغة العامة التالية:

control.CreateGraphics.DrawEllipse(Pen, x, y, width, height)

وكما ترى، تحتوى هذه الوظيفة على نفس معاملات الوظيفة السابقة.

الوظيفة DrawPolygon()

تقوم هذه الوظيفة برسم مضلع باستخدام مصفوفة من النقاط التى تحدد أبعاد هذا المضلع وعدد أضلاعه وتحتوى على الصيغة العامة التالية:

control.CreateGraphics.DrawPolygon(Pen, PointsArray)

حيث **PointsAray** عبارة عن مصفوفة من النقاط.

الوظيفة Clear()

تقوم هذه الوظيفة بمسح جميع الرسوم الموجودة على الأداة أو النموذج وإحلالها بلون الخلفية وتحتوى على الصيغة العامة التالية:

control.CreateGraphics.Clear(Color)

حيث يعبر **Color** عن لون الخلفية الذى ترغب فى استخدامه وهو اختيارى. فإذا أهملته يتم استخدام لون الخلفية الحالى للأداة.

استخدام الأقلام وفرش الرسم والألوان

يمكنك استخدام كائنات القلم والفرشاة لرسم الخطوط والأشكال المختلفة. فالقلم هو أحد حالات التصنيف **Pen** ويستخدم لرسم الخطوط والأشكال المجوفة (أى الخالية من الألوان). بينما تعتبر الفرشاة إحدى الحالات المشتقة من التصنيف **Brush** وتستخدم فى رسم الأشكال الملونة. وهناك أيضاً كائنات الألوان التى تشتق من التصنيف **Color** وتستخدم مع كلٍ من الأقلام وفرش الرسم لتوضيح اللون الذى يتم استخدامه فى عملية الرسم.

استخدام الأقلام Pens

يستخدم القلم Pen فى رسم الخطوط والمنحنيات والأشكال المجوفة ويتم تعريفه كما يلى:

- يمكنك تعريف قلم أحمر اللون بسمك نقطة واحدة باستخدام العبارة التالية:

```
Dim myPen As New Pen (Color.Black)
```

- يمكنك تعيين سمك أكبر للقلم وليكن ٥ نقاط كما يلى:

```
Dim myPen As New Pen (Color.Black,5)
```

- يمكنك استخدام لون إحدى فرش الرسم المعرفة من قبل كما يلى:

```
Dim myPen As New Pen (myBrush,5)
```

وبمجرد تعريف القلم، يمكنك استخدامه فى رسم خط أو منحنى أو حتى شكل مجوف كما فى المثال التالى الذى يقوم برسم قطع ناقص:

```
Dim myPen As New Pen (Color.Black)
Dim g As Graphics = Me.CreateGraphics
g.DrawEllipse(myPen,20,30,10,6)
```

وبمجرد تعريف القلم يمكنك التحكم فى خصائصه المختلفة والتى تحدد طريقة رسمه للخطوط. فهناك الخصائص Width وColor التى تحدد مظهر الخط، والخصائص StartCap وEndCap التى تتيح لك إضافة أشكال مخصصة لبداية الخط ونهايته، والخاصية DashStyle التى تتيح لك الاختيار بين الأشكال المختلفة للخط هل هو مجسم Solid أو منقط Doted أو مقطع Dashed وغيرها من الخيارات الأخرى.

حذف الأقلام

بعد الانتهاء من استخدام القلم، يكون من الأفضل حذفه لتوفير المساحة المستغلة بالذاكرة. لأداء ذلك، قم باستخدام الوظيفة Dispose كما يلى:

```
myPen.Dispose()
```

استخدام فرش الرسم

فرش الرسمBrushes عبارة عن كائنات تستخدم مع كائن الرسوم Graphics لإنشاء الأشكال المجسمة (ذات اللون الداخلى). يوضح جدول ١٨-٤ التالى الأنواع المختلفة من فرش الرسم وهى عبارة عن تصنيفات مشتقة أساساً من التصنيف المجرد Brush.

جدول ١٨-٤ الأنواع المختلفة من فرش الرسم

تصنيف الفرشاة	الوصف
SolidBrush	هذا النوع هو أبسط أنواع فرش الرسم ويستخدم فى رسم اللون المصمت
HatchBrush	تشبه النوع السابق إلى حد كبير إلا أنها تتيح لك الاختيار من بين عدد من الأشكال دون التقييد باللون المصمت
TextureBrush	تقوم بالرسم باستخدام مادة داخلية كالصور مثلاً
LinearGradientBrush	تقوم بالرسم بلونين مع استخدام تدرج الألوان من البداية إلى النهاية
PathGradientBrush	تقوم بالرسم باستخدام مجموعة من الألوان المركبة متدرجة اللون وذلك من خلال مسار تقوم أنت بتحديده

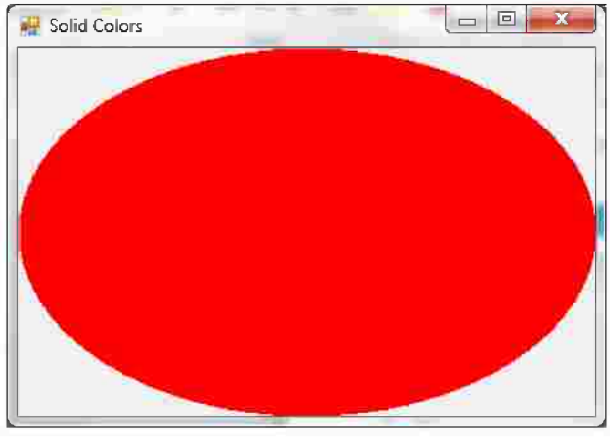
وفيما يلى نوضح بعض الأمثلة على استخدام هذه الأنواع.

رسم الأشكال المجسمة

لرسم قطع ناقص مصمت أحمر اللون على النموذج، قم بإدخال الكود التالى داخل الحدث Form_Paint:

```
Dim myBrush As New SolidBrush(Color.Red)
Me.CreateGraphics.FillEllipse(myBrush, ClientRectangle)
```

حيث قمنا باستخدام ClientRectangle لتحديد مساحة الأداة المستخدمة بالكامل (النموذج فى هذه الحالة)، لذا حينما تقوم بتنفيذ هذا الكود، يظهر لك قطع ناقص حمراء بحجم النموذج بالكامل (انظر شكل ١٨-٦).



شكل ١٨-٦ القطع الناقص بحجم النموذج.

يمتاز الحدث **Paint** بإعادة تنفيذ الكود الموجود بداخله كلما تم تغيير حجم النموذج أو تحديثه وهذا على عكس الحدث **Load** الذى يتم تنفيذه بمجرد تحميل النموذج داخل الذاكرة، لذا يجب دائماً وضع كود الرسوم الخاصة بالنموذج داخل الحدث **Form_Paint**.

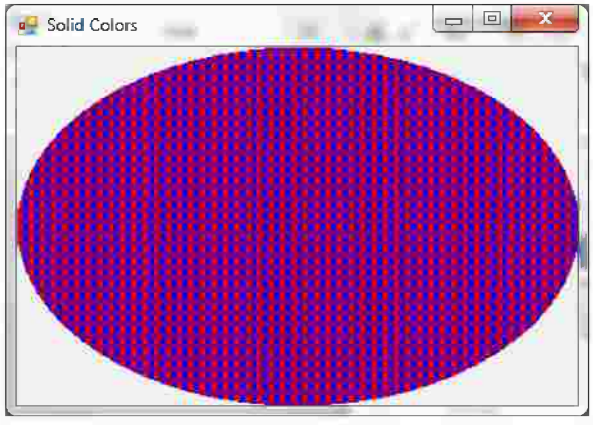
رسم الأشكال البسيطة

يمكنك استخدام الفرشاة **HatchBrush** فى الرسم باستخدام عدد كبير من الأشكال وليس اللون المصمت كما فى حالة الفرشاة السابقة. لرسم قطع ناقص نسيجى بمقدمة حمراء وخلفية زرقاء (انظر شكل ١٨-٧)، قم بكتابة السطر التالى أعلى نافذة الكود:

```
Imports System.Drawing.Drawing2D
```

ثم قم بإدخال الكود التالى داخل الحدث **Form_Paint**:

```
Dim myHatchBrush As New HatchBrush(HatchStyle.Plaid,
Color.Red, Color.Blue)
Me.CreateGraphics.FillEllipse(myHatchBrush, ClientRectangle)
```



شكل ١٨-٧ رسم قطع ناقص نسيجي من خلال الفرشاة HatchBrush.

رسم الأشكال المركبة

يمكنك رسم الأشكال المركبة التي تحتوى بداخلها على مادةٍ ما ولتكن صورة من الصور مثلاً باستخدام الفرشاة **TextureBrush**. لرسم القطع الناقص السابق باستخدام مادة داخلية عبارة عن صورة (انظر شكل ١٨-٨)، قم بإدخال الكود التالى داخل الحدث **Paint** الخاص بالنموذج:

```
Dim myBrush As New TextureBrush(New Bitmap("c:\logo.bmp"))
Me.CreateGraphics.FillEllipse(myBrush, ClientRectangle)
```

شكل ١٨-٨ الرسم باستخدام محتويات صورة.

رسم الظلال المركبة

يمكنك استخدام الفرشاة **LinearGradientBrush** لرسم الظلال المركبة التى تتكون من لونين، بينما يمكنك استخدام الفرشاة **PathGradientBrush** لرسم الظلال الأكثر تعقيداً. لترى تأثير استخدام هذا النوع من فرش الرسم، قم بإدخال الكود التالى داخل الحدث **Paint** الخاص بالنموذج (انظر شكل ١٨-٩):

```
Dim myBrush As New LinearGradientBrush(ClientRectangle,
Color.Red, Color.Yellow, LinearGradientMode.Vertical)
Me.CreateGraphics.FillEllipse(myBrush, ClientRectangle)
```

شكل ١٨-٩ تتدرج الألوان داخل القطع الناقص رأسياً من اللون الأحمر إلى اللون الأصفر.

حذف فرش الرسم

بعد الانتهاء من استخدام فرشاة الرسم، يكون من الأفضل حذف الفرشاة لتوفير المساحة المستغلة بالذاكرة. لأداء ذلك، قم باستخدام الوظيفة **()Dispose** كما يلى:

```
myBrush.Dispose()
```

استخدام الألوان

للألوان علاقة وطيدة بكلٍ من الأقلام وفرش الرسم، فهى التى تحدد اللون المستخدم فى رسم الخطوط أو الأشكال بالقلم كما أنها تحدد اللون الداخلى المستخدم مع الفرشاة. وهناك مجموعة من الألوان المعرفة مسبقاً داخل **Visual Basic 2012** والتى تزيد عن مائة لون مثل الألوان التالية:

```
Dim myColor As Color
myColor = Color.Red
myColor = Color.Aquamarine
myColor = Color.LightGoldenrodYellow
myColor = Color.PapayaWhip
myColor = Color.Tomato
```

فكل من العبارات السابقة تقوم بتخصيص لون معرف مسبقاً إلى اللون myColor الذى يمكنك استخدامه فيما بعد كلون جديد مع الأقلام وفرش الرسم.

إنشاء ألوان مخصصة

فضلاً عن الألوان المعرفة داخل النظام، يمكنك تعريف ألوان مخصصة بنفسك باستخدام الوظيفة Color.FromArgb() والتى تتيح لك تحديد دقة الألوان الثلاثة الأساسية الأحمر Red والأخضر Green والأزرق Blue كما فى الكود التالى:

```
Dim myColor As Color
myColor = Color.FromArgb(23,56,78)
```

وهذا الكود يقوم بتعريف لون جديد رمادى مائل إلى الزرقة. وكل رقم من الأرقام الثلاثة يكون من 0 إلى 255 حيث يعبر الرقم 0 عن غياب اللون تماماً بينما يعبر الرقم 255 عن اكتماله، لذا فإن Color.FromArgb(0,0,0) تعنى اللون الأسود، بينما Color.FromArgb(255,255,255) تعنى اللون الأبيض.

يمكنك أيضاً استخدام نفس الوظيفة FromArgb() لتعيين شفافية اللون أو ما نطلق عليه "ألفا" Alpha وذلك بإدخال رقم من 0 إلى 255 كمعامل أول قبل المعاملات الثلاثة السابقة كما يلى:

```
Dim myColor As Color
myColor = Color.FromArgb(127,23,56,78)
```

وهذا الكود يقوم بتعريف لون رمادى مائل للزرقة وله درجة شفافية بنسبة ٦% .

يمكنك بالطبع إضافة درجة الشفافية إلى لون موجود مسبقاً سواءً من الألوان المعرفة داخل NET. أو التى قمت بتعريفها بنفسك كما فى الكود التالى:

```
Dim myColor As Color
myColor = Color.FromArgb(127,Color.Red)
```

رسم الخطوط والأشكال

بعد أن تعرفنا على مفهوم الأقلام وفرش الرسم والألوان، دعنا نتعرف على كيفية رسم الخطوط والأشكال، حيث يحتوى التصنيف **Graphics** على الوظائف المستخدمة لرسم المجموعات المختلفة من الخطوط والأشكال سواءً البسيطة أو المركبة التى تتكون من ألوان مجسمة أو شفافة. كما عرفنا منذ قليل يمكنك إنشاء الخطوط والمنحنيات المفتوحة والأشكال باستخدام التصنيف **Pen**، كما يمكنك ملء الأشكال كالمستطيلات والمضلعات باستخدام التصنيفات المشتقة من التصنيف**Brush**.

الخطوات العامة لرسم الخطوط والأشكال المجوفة

لرسم خط أو شكل مجوف (أى لا يحتوى على ألوان داخلية)، قم بإجراء الخطوات العامة التالية:

١. قم بإنشاء مرجع للتصنيف (الكائن) **Graphics** الذى ستقوم باستخدامه فى عملية الرسم (وليكن زر **Button1** مثلاً) كما فى الكود التالى:

```
Dim g As Graphics = Button1.CreateGraphics
```

٢. قم بإنشاء حالة جديدة من التصنيف **Pen** لتحديد مواصفات القلم الذى ترغب فى استخدامه فى عملية الرسم كما فى الكود التالى الذى يقوم بتعريف قلم رسم باسم **myPen** يحتوى على اللون الأحمر وعرضه ٥ نقاط:

```
Dim myPen As New Pen(Color.Red,5)
```

٣. قم باستدعاء الوظيفة المناسبة للشكل الذى ترغب فى رسمه مستعيناً بجدول ١٨-٥ وباستخدام كود مشابه للكود التالى:

جدول ١٨-٥ وظائف رسم الخطوط والأشكال المجوفة

الوظيفة	الشكل الناتج
Graphics.DrawLine()	تقوم هذه الوظيفة برسم خط وتحتاج إلى تعيين إحداثيات نقطة البدء ونقطة الانتهاء وكذلك القلم المستخدم فى عملية الرسم.

الوظيفة	الشكل الناتج
Graphics.DrawPolygon()	تقوم هذه الوظيفة برسم الأشكال المعقدة وربما تحتاج إلى مصفوفة من الإحداثيات.
Graphics.DrawRectangle()	تقوم هذه الوظيفة برسم مستطيل وتحتاج إلى كائن أو أكثر كمعاملات.

```
g.DrawLine(myPen, 5, 5, 45, 65)
g.DrawBezier(myPen, 15, 15, 30, 30, 45, 30, 87, 20)
g.DrawEllipse(myPen, New Rectangle(33, 45, 40, 6))
g.DrawPolygon(myPen, New PointF() {New PointF(1, 1), New
PointF(20, 10), New PointF(5, 4), New PointF(100, 2), New
PointF(200, 6), New PointF(39, 45)})
```

الخطوات العامة لرسم الأشكال الملونة

لرسم شكل ملون (أى يحتوى على ألوان داخلية)، قم بإجراء الخطوات العامة التالية:

١. قم بإنشاء مرجع للتصنيف (الكائن) Graphics الذى ستقوم باستخدامه فى عملية الرسم (وليكن زر Button1 مثلاً) كما فى الكود التالى:

```
Dim g As Graphics = Button1.CreateGraphics
```

٢. قم بإنشاء حالة جديدة من تصنيف الفرشاة التى ترغب فى استخدامها لتحديد مواصفاتها (راجع جدول ١٨-٤) كما فى الكود التالى الذى يقوم بتعريف فرشاة مصمتة باسم myBrush تحتوى على اللون الأحمر:

```
Dim myBrush As New SolidBrush(Color.Red)
```

٣. قم باستدعاء الوظيفة المناسبة للشكل الذى ترغب فى رسمه مستعيناً بجدول ١٨-٦ وباستخدام كود مشابه للكود التالى:

جدول ١٨-٦ وظائف رسم الأشكال الملونة

الوظيفة	الشكل الناتج
Graphics.FillPolygon()	تقوم هذه الوظيفة برسم الأشكال المعقدة وربما

الوظيفة	الشكل الناتج
	تحتاج إلى مصفوفة من الإحداثيات.
Graphics.FillRectangle()	تقوم هذه الوظيفة برسم مستطيل وتحتاج إلى كائن أو أكثر كمعاملات.
Graphics.FillPie()	تقوم هذه الوظيفة برسم مخطط بياني.
Graphics.FillPath()	تقوم هذه الوظيفة برسم مسار محدد.

```
g.FillRectangle(myBrush, New Rectangle(33, 45, 40, 6))
g.FillPie(myBrush, New Rectangle(33, 45, 40, 6),0,90)
g.FillPolygon(myBrush, New PointF() {New PointF(1, 1), New
PointF(20, 10), New PointF(5, 4), New PointF(100, 2), New
PointF(200, 6), New PointF(39, 45)})
```