الفصل الثانى
الجديد فى
Visual Basic 2008

يشرح هذا الفصل أهم السمات الجديدة فى **Visual Basic 2008** والتى لم تكن موجودة بالإصدارات السابقة من اللغة وبخاصةٍ الإصدارات التى تسبق **Visual Basic.NET**. لذا فإن هذا الفصل يخاطب من لهم خبرة مسبقة بالتعامل مع أيٍ من إصدارات **Visual Basic** التى تسبق **Visual Basic 2008**.

فإذا كنت تتعلم هذه اللغة للمرة الأولى، فننصحك بتخطى هذا الفصل والذهاب مباشرةً إلى الفصل التالى.

بانتهاء هذا الفصل، ستتعرف على السمات الجديدة والتغييرات الموجودة داخل **Visual Basic 2008** والتى تتضمن ما يلى:

- الجديد فى **Visual Basic 2008**.
- الجديد فى بيئة التطوير.
- تغييرات على مستوى الكود وأنواع البيانات
- تغييرات على مستوى الإجراءات
- تغييرات على ملفات المشروع
- تغييرات على مستوى أدوات التحكم

إذا كنت ممن استخدموا Visual Basic 6.0 أو أحد الإصدارات السابقة من Visual Basic، فبلا شك ستلاحظ أثناء عملك مع بيئة التطوير المتكاملة Visual Studio 2008 التطوير الكبير الذى حدث لها مقارنةً بالإصدارات السابقة. وستفاجأ بأن مهارات تطوير برامج Visual Basic التى تعلمتها واعتدت عليها على مدار سنوات، قد فنيت تماماً وتم استبدالها بمهاراتٍ أخرى. لكن على رسلك، لم أقصد بتلك المقدمة إضعاف عزيمتك، لكن ما أردت أن أنقله إليك أن هناك تغييراً كبيراً فى البنية التحتية للغة، إلا أن المبدأ الأساسى الذى اعتدنا أن نرى Visual Basic عليه وهو السهولة والسرعة فى الوصول إلى ما تريد لم يزل موجوداً فى Visual Basic 2008.

سنقوم فى هذا الفصل بالتعرف فى عجالة على التغييرات التى طرأت على بيئة التطوير Visual Basic بما فى ذلك التغييرات التى طرأت على اللغة نفسها، فكن معنا.

لماذا تغيرت لغة Visual Basic ؟

لعلك تعلم أن الإصدارات السابقة من Visual Basic كانت تستخدم أساساً فى إنشاء وتطوير تطبيقات النوافذ الصالحة للعمل تحت نظام التشغيل Windows. ومع التقدم الكبير فى عالم الاتصالات والإنترنت، كان لابد من إعطاء أهمية أكبر لتطبيقات الويب على أساس أنها بالفعل المستقبل المنشود. لذا قامت مايكروسوفت كإحدى الشركات الرائدة فى إنتاج البرمجيات فى العالم بتطوير Visual Basic.NET ضمن سلسلة أو مجموعة تسمى Visual Studio.NET تحوى بين طياتها مجموعة من لغات البرمجة الأخرى بالإضافة إلى Visual Basic.NET. ثم أعقبت ذلك بتطوير آخر وهو Visual Basic 2005 ضمن مجموعة Visual Studio 2005 ثم Visual Basic 2008 ضمن مجموعة Visual Studio 2008، وجميع هذه اللغات قادرة على إنشاء تطبيقات النوافذ وتطبيقات الويب بالإضافة إلى خدمات الويب التى تكون متاحة لمرتادى الإنترنت.

نتناول فى هذا الفصل الجديد من ثلاث جهات مختلفة:

الأولى: نتحدث فيها عن الجديد فى Visual Basic 2008 فقط والذى لم يكن موجوداً فى Visual Basic 2005.

الثانية: نتحدث عن الجديد فى Visual Basic 2008 وVisual Basic 2005 والذى لم يكن موجوداً فى Visual Basic.NET

الثالثة: نتحدث فيها عن الجديد فى كلٍ من Visual Basic 2008 و Visual Basic 2005 وVisual Basic.NET مقارنة بـ Visual Basic 6.0 وما قبله.

أولاً: الجديد فى Visual Basic 2008

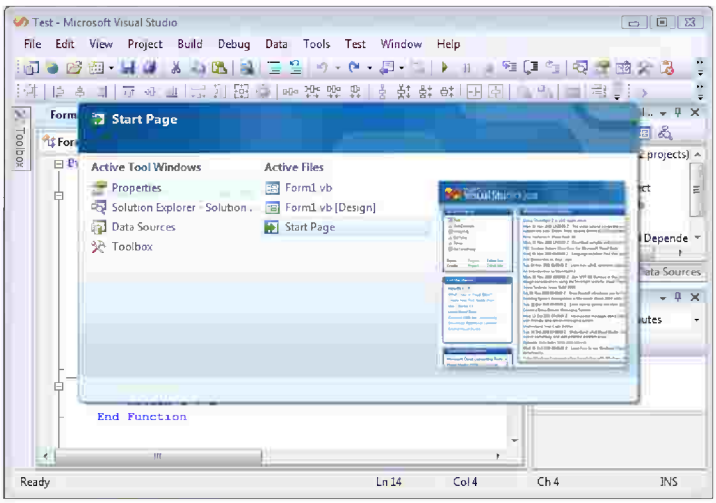
تكمن معظم التحديثات والسمات الجديدة التى ظهرت مع Visual Basic 2008 ولم تكن موجودة بالإصدارات السابقة فى سرعة إنشاء وتنفيذ التطبيقات من جهة، وتقليل الكود المستخدم فى إنشائها ومن ثمَّ سهولة عملية التطوير برمتها من جهة أخرى. أما بيئة التطوير المتكاملة، فلم تنل تحديثاً ملفتاً للنظر كما اعتدنا بالإصدارات السابقة.

ويمكن إجمال أهم التحديثات والسمات الملموسة التى ظهرت مع Visual Basic 2008 فيما يلى:

- تم تبسيط الكود المصدرى للمشروعات ومن ثم يمكنك التعامل مع هذا الكود وتعديله بسهولة مقارنة بالإصدار السابق. كما تستطيع الآن المكونات التى تستخدم سمات متقدمة التفاعل فيما بينها بسهولة تامة.
- تم استحداث تقنية جديدة للوصول إلى البيانات تسمى Language Integrated Query (LINQ) والتى يمكنك استخدامها فى الوصول إلى البيانات بقواعد البيانات العلائقية.
- يدعم Visual Basic 2008 ثلاثة إصدارات من NET Framework. وهى NET Framework 2.0. الذى يتم تثبيته مع Visual Studio 2005 و NET. Framework 3.0 الذى يتم تثبيته مع Windows Vista، وأخيراً NET. Framework 3.5 الذى يتم تثبيته مع Visual Studio 2008، ومن ثمَّ يمكنك

عند إنشاء مشروعاتك المختلفة اختيار الإصدار المناسب الذى يتناسب مع الكود الذى ستقوم بكتابته ومعارفك الشخصية. ويحتوى الإصدار الجديد .NET Framework على الدعم الكامل للتقنيات الجديدة التى ظهرت مع Visual Basic 2008 مثل Language Integrated Query (LINQ) المستخدمة فى الاستعلام عن مختلف أنواع البيانات و Windows Presentation Foundation(WPF) المستخدمة فى إنشاء الرسوم المركبة شديدة التعقيد، و Windows Communication Foundation (WCF) المستخدمة فى إنشاء برامج تعمل مع خدمات الويب، و Windows Workflow Foundation (WF) المستخدمة فى إنشاء برامج خطط سير الأعمال.

- سهولة التحويل من البرامج المنشأة بالإصدارات القديمة من Visual Basic مثل Visual Basic 6.0 على سبيل المثال إلى تنسيق Visual Basic 2008 من خلال معالج سهل الاستخدام.
- متصفح بيئة التطوير IDE Navigator والذى يمكنك استخدامه فى التنقل بين الملفات المفتوحة المختلفة الموجودة بالحل الحالى (انظر شكل ٢-١). لإظهار متصفح بيئة التطوير، اضغط الاختصار Ctrl+Tab من لوحة المفاتيح. كرر الضغط على مفتاح Tab حتى تصل إلى الملف المطلوب ثم قم بتحرير مفتاح Ctrl لتنشيط الملف الذى قمت باختياره داخل بيئة التطوير.



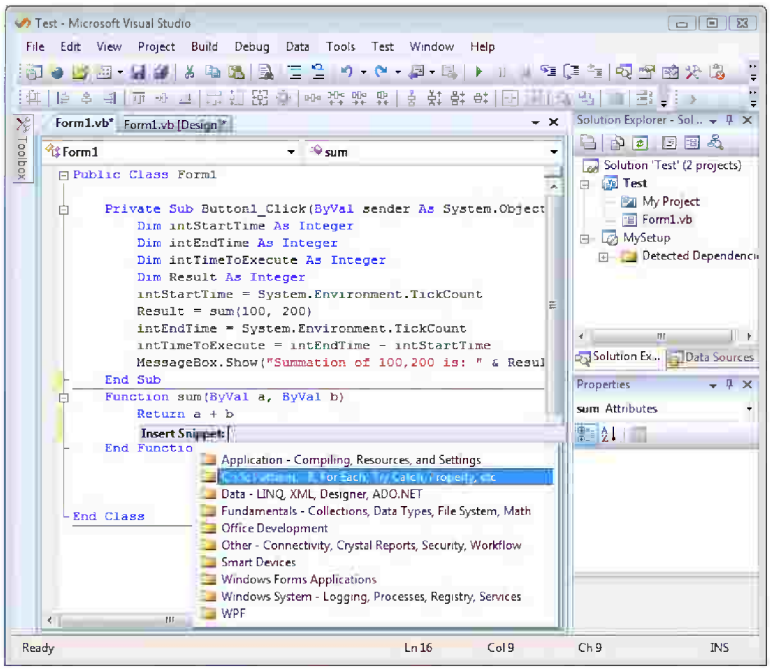
شكل ٢-١ استخدام متصفح بيئة التطوير فى التنقل بين الملفات المفتوحة

- مستعرض ويب بيئة التطوير، حيث تحتوى بيئة تطـــوير Visual Studio 2008 على مستعرض ويب شبيه إلى حد ما بمستعرض الويـــب Internet Explorer وبذلك يمكنك فتح صفحات ومواقع الويب المختلفة من داخل بيئة التطوير بدلاً من استخدام مستعرض الويب الرئيسى (انظر شكل ٢-٢). لإظهار مستعرض ويـــب بيئة التطوير، افتح قائمة View بشريط القـــوائم ثم اختـــر Other Windows وWeb Browser من القوائم المنسدلة.

شكل ٢-٢ استخدام مستعرض ويب بيئة التطوير

- الصيغ المختصرة للعمليات الحسابية، ففى حالة وجود متغير واحد مشترك بطرفى العملية الحسابية، يمكنك استخدام الصيغة المختصرة. فعلى سبيل المثال، يمكنك استبدال العبارة **X=X+1** بصيغتها المختصرة المكافأة التالية **X+=1**.
- المعاملين المنطقيين **AndAlso** و**OrElse** الذين يتم استخدامهما داخل العبارات المنطقية كعبارة **If** على سبيل المثال، ومن خلالهما يتم اختبار أكثر من شرط فى العبارة الواحدة. ففى حالة استخدام **AndAlso**، يجب أن يكون الشرطان صحيحان، فإذا كان الأول غير صحيح، يتم على الفور تخطى الشرط الثانى. أما فى حالة استخدام **OrElse**، يجب أن يكون أحد الشرطين صحيحاً، فإذا كان الأول صحيحاً، لا يتم اختبار الشرط الثانى لعدم جدواه.
- إدراج أجزاء جاهزة من الكود، حيث يمكنك استخدام الأمر **Insert Snippet** لإدراج جزء من الكود المعرف مسبقاً داخل **Visual Basic** أو إدراج هياكل العبارات كثيرة الاستخدام كعبارة **If** على سبيل المثال. لأداء ذلك، انقر داخل المكان الذى ترغب فى إضافة الكود إليه ثم افتح قائمة **Edit** بشريط القوائم واختر

IntelliSense ثم Insert Snippet من القوائم المنسدلة الناتجة (أو اضغط الاختصار Ctrl+K ثم Ctrl+X من لوحة المفاتيح). انقر المجموعة التى تحتوى على الكود الذى ترغب فى استخدامه نقراً مزدوجاً ثم اختر هذا الكود لإضافته (انظر شكل ٢-٣).



شكل ٢-٣ يمكنك إدراج أجزاء جاهزة من الكود

- استخدام الكائن Err أثناء العمل مع الاستثناءات. فمن خلال الخاصية Err.Number يمكنك التعرف على رقم الخطأ والذى يكون له بالطبع مدلول معين عن سبب حدوث المشكلة، كما يمكنك من خلال الخاصية Err.Description الحصول على الوصف (الشرح) الخاص بالاستثناء. وعادةً ما يتم استخدام هذا الكائن مع التركيب Catch When مثل Catch When Err.Number = 53 والذى يدل على عدم القدرة على قراءة الملف المحدد، أى أن الملف غير موجود بالمكان المحدد.

ثانياً : الجديد بدءً من Visual Basic 2005

تضمنت Visual Basic 2005 العديد من السمات الجديدة والتى سنقوم فيما يلى بإلقاء نظرة خاطفة عليها.

استخدام العبارة Continue

تحتوى لغة Visual Basic على العبارة Continue والتى يتم من خلالها الخروج من الدورة الحالية بالدوارة Do وFor وWhile. فإذا أردت فى أى وقت الخروج من الدورة الحالية داخل الدوارة المكونة عادةً من عدد من الدورات، استخدم العبارة Continue.

الوصول إلى النماذج

يمكنك داخل Visual Basic 2008 الوصول إلى النماذج بنفس الطريقة المستخدمة فى Visual Basic 6.0 وذلك بذكر اسم النموذج مباشرةً بدلاً من التقيد بإنشاء حالة من النموذج أولاً كما هو متبع فى Visual Basic.Net. فعلى سبيل المثال، لإظهار النموذج Form1، استخدم الصيغة التالية:

```
Form1.Show()
```

أما إذا أردت إنشاء حالة جديدة من النموذج، فيجب فى هذه الحالة استخدام المعامل New فى تعريف الحالة الجديدة ثم استدعاء الوظائف والخصائص مع هذه الحالة هكذا مثلاً:

```
Dim newForm1 As New Form1
newForm1.Show()
```

استخدام المعامل IsNot

يحتوى Visual Basic 2008 على المعامل IsNot والذى يمكنك استخدامه لتجنب استخدام المعاملين Not وIs، فالمعامل الجديد أسهل فى الفهم من المعاملين المستخدمين بالإصدارات القديمة من Visual Basic.

استخدام العبارة Using

يحتوى Visual Basic 2008 على العبارة Using ... End Using والتى تمكنك من استدعاء موارد محددة بجزء معين من الكود بدلاً من استخدامها داخل الكود بأكمله.

استخدام التعبير .. 0 To فى تحديد عناصر المصفوفة

يمكنك داخل Visual Basic 2008 استخدام التعبير ... 0 To فى تحديد المعامل الأدنى للمصفوفة ليبدأ من القيمة صفر مما يجعل الكود المستخدم سهل القراءة وسهل الفهم. ففى جميع الأحوال تبدأ عناصر المصفوفة من المعامل صفر، ولكن الصيغة الجديدة أكثر وضوحاً من الصيغة المعتادة. فللعبارتين التاليتين نفس المدلول:

```
Dim monthtotal(0 To 11) As Double
Dim monthtotal(11) As Double
```

فكلا العبارتين يقوم بتعريف مصفوفة باسم monthtotal تحتوى على ١٢ عنصراً تبدأ من الدليل صفر وتنتهى بالدليل ١١.

الخصائص ذات مستويات الوصول المتعددة

يمكنك داخل Visual Basic 2008 تعريف خصائص بمستويات وصول متعددة داخل الإجراءات Set وGet الموجودة بهذه الخصائص.

أنواع البيانات عديمة الإشارة

يحتوى Visual Basic 2008 على أنواع بيانات عديمة الإشارة، أى تحتوى على قيم موجبة فقط، وهى نوع البيانات UShort ونوع البيانات UInteger ونوع البيانات ULong بالإضافة إلى نوع البيانات SByte، وهى الأنواع المصاحبة للأنواع Short وInteger وLong وByte على الترتيب، إلا أن الأولى تحتوى على قيم موجبة فقط، بينما تحتوى الأخيرة على قيم موجبة وسالبة.

استخدام المتغيرات عديمة القيمة

يمكنك داخل Visual Basic 2008 تعريف متغيرات عديمة القيمة وبالتالى لا يقوم المترجم بتخصيصها بقيم افتراضية فى حالة عدم تخصيصها بقيمة أثناء عملية التعريف وذلك

باستخدام كلمة **Nullable** أثناء تعريف المتغير هكذا مثلاً:

```
Dim noOfStudents As Nullable(Of Integer)
```

التحميل الزائد للمعاملات

يمكنك داخل **Visual Basic 2008** تعريف وظائف مختلفة للمعاملات المعتادة كمعامل الجمع + ومعامل الضرب * وغيرها من المعاملات الأخرى، وهو ما يسمى بالتحميل الزائد للمعاملات **Operator Overloading**.

الفصل بين الأكواد

يمكنك داخل **Visual Basic 2008** فصل الكود الذى يتم توليده تلقائياً نيابةً عنك من خلال بيئة التطوير والكود الذى تقوم بتعريفه بنفسك وذلك بوضع كلاهمـــا فى ملفـــات مستقلة، وبالتالى يمكنك العمل مع الكود الذى قمت بكتابته بسهولة شديدة نظراً لإخفاء الكود التلقائى عن عينيك. كما يمكنك أيضاً تقسيم الكود على أكثر من ملف باستخدام الكلمة الأساسية **Partial**.

استخدام أنواع البيانات العامة

يدعم **Visual Basic 2008** أنواع البيانات العامة والتى يمكنها التعامل مع أكثر من نوع من أنواع البيانات.

تعريف الأحداث المخصصة

يمكنك داخل **Visual Basic 2008** تعريف أحداث مخصصة جديدة للكائن من خـــلال الكلمة الأساسية **Custom** بالعبارة **Event**.

استخدام السمة ***My***

من السمات الجديدة التى هلت مع **Visual Basic 2005** السمة **My** والتى يمكنك من خلالها الوصول إلى الكائنات والخصائص والوظائف المختلفة شائعة الاستخدام داخل نطاق **Net.** المعروف باسم **Net Framework.** إلى جانب العناصر المختلفة التى قمت بإنشائها داخل التطبيق. كما يمكنك من خلال **My** أيضاً الوصول إلى الموارد المختلفة الموجودة على

حاسبك كالحافظة Clipboard (من خلال الكائن My.clipboard) والمؤقت ولوحة المفاتيح والفأرة وغيرها. كما يمكنك كذلك من خلال My التعامل مع الملفات والعناصر المختلفة للشبكة. يوضح جدول ٢-١ التالى التصنيفات الأساسية داخل الكائن My واستخدام كلٍ منها.

جدول ٢-١ التصنيفات الأساسية داخل الكائن My

اسم التصنيف	الوظيفة
My.Application	يحتوى على بيانات عن التطبيق الحالى وخدماته المختلفة
My.Computer	يحتوى على ارتباطات مصادر وأدوات جهاز الكمبيوتر
My.Forms	يحتوى على ارتباطات لجميع النماذج الموجودة بالتطبيق
My.Log	الوصول إلى أدوات تسجيل سجل تطبيقات ASP.NET التى تستطيع التعامل مع عمليات الدخول بوسائل مختلفة
My.Request	الوصول إلى طلب الويب الحالى لتطبيقات ASP.NET
My.Resources	يحتوى على مصادر التطبيقات العامة
My.Response	الوصول إلى استجابة الويب الحالية مع تطبيقات ASP.NET
My.Settings	القدرة على تخزين إعدادات التطبيق الحالى واسترجاعها
My.User	الحصول على البيانات المختلفة لاسم المستخدم الحالى للتطبيق
My.WebServices	الوصول إلى خدمات الويب التى قام التطبيق بإنشاء مرجع لها

تعديل البرنامج أثناء التشغيل

يمكنك داخل Visual Basic 2008 إدخال التعديلات على البرنامج أثناء تشغيله فى طور التصحيح Debug Mode فى حالة الراحة Break ثم الاستمرار فى عمل البرنامج مع تطبيق التعديلات التى قمت بها.

تعديل سمات التصنيفات من خلال نافذة الخصائص

يمكنك الآن داخل **Visual Basic 2008** تعديل السمات التى يتم تطبيقها على الوظائف والتصنيفات المعرفة داخل التطبيق من نافذة الخصائص بنفس طريقة تعيين خصائص أدوات التحكم بدلاً من التقيد بتعيينها من خلال الكود.

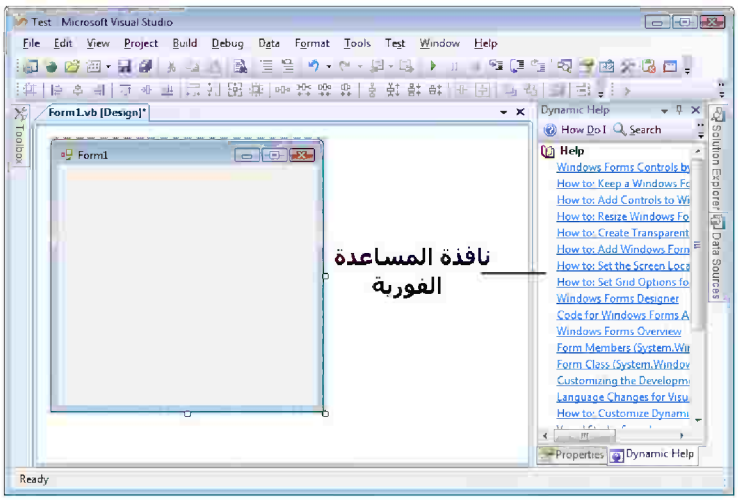
تنقية العناصر المعتادة

عند استخدام معامل النقطة داخل نافذة الكود، يتم إظهار العناصر المناسبة للكائن الـــذى يسبق النقطة فى تبويبين منفصلين، يحتوى على الأول **Common** على العناصر الشـــهيرة المستخدمة بكثرة داخل الكود، بينما يحتوى الآخر **All** على جميع العناصر الموجودة بالكائن من وظائف وخصائص.

الجديد فى بيئة تطوير Visual Studio 2008

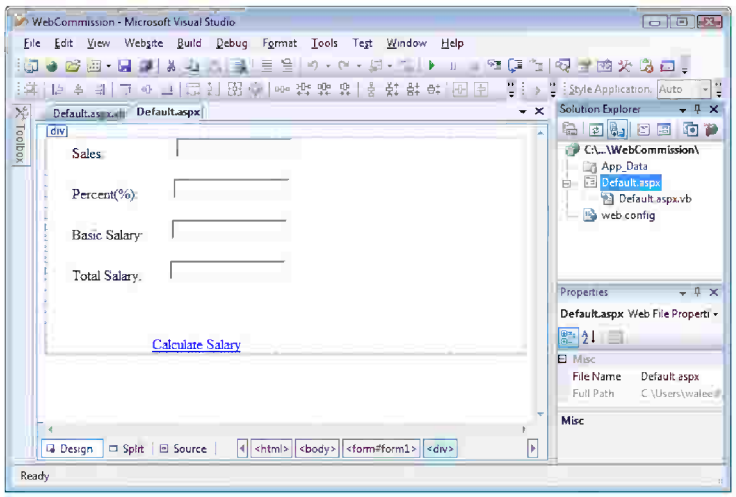
تبنى بيئة التطوير **Visual Studio 2008** أساساً على السمات الموجودة بالإصدار السابق بالإضافة إلى بعض السمات الجديدة التى تم إضافتها والتى يمكن إجمالها فيما يلى:

- إظهار المعلومات الصحيحة فى الوقت المناسب. فحينما تقوم على سبيل المثال بفتح بيئة التطوير دون أن تفتح أىٍ من التطبيقات الموجودة على حاسبك، فإن بيئة التطوير تقوم بإظهار معلوماتٍ عن كيفية إنشاء التطبيقات الجديدة كما تقوم بإمدادك بمعلومات عن القوالب والمعالجات المتاحة التى يمكنك استخدامها فى إنشاء تطبيقاتك. وبمجرد البدء فى إنشاء التطبيق، يتم إظهار المعلومات المناسبة التى تعتمد على موقعك داخل التطبيق وعلى الأداة التى تقوم باستخدامها. كل هذه المعلومات تظهر داخل نافذة المساعدة الفعالة **Dynamic Help Window** (انظر شكل ٢-٤).

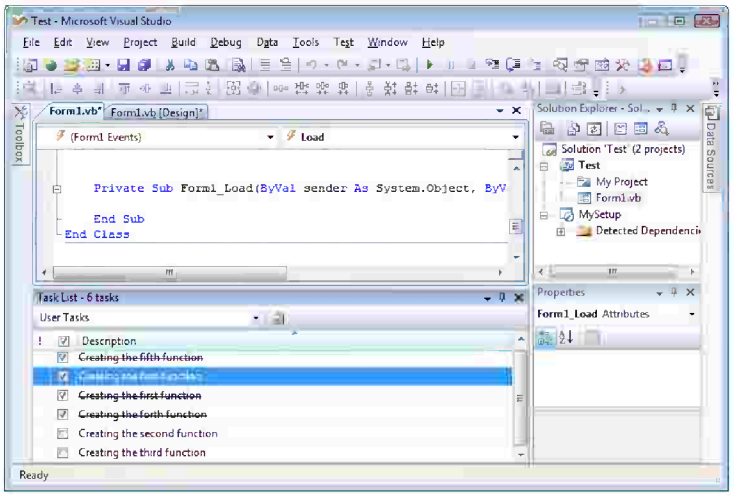


شكل ٢-٤ الحصول على المعلومات من خلال نافذة المساعدة الفعالة.

- محرر صفحات الويب المرئى وهو محرر يعمل بتقنية "ما تراه هو ما تحصل عليه" أو **WYSIWYG** حيث يمكنك من خلال هذا المحرر استخدام الواجهة الرسومية لإنشاء صفحات الويب دون الحاجة إلى معرفة كود **HTML** أو كود المستند البرمجى **Script**. وحقيقةً فإن هذا المحرر يحتوى على العديد من السمات التى تمكنك من إنشاء صفحات الويب بسهولة مثل الإكمال التلقائى للعبارات واختبار صيغة **XML** أثناء التشغيل والتحكم فى وضع العناصر على الصفحة (انظر شكل ٢-٥).
- نافذة قوائم المهام **Task Lists** يمكنك من خلالها إضافة المهام التى ترغب فى تنفيذها داخل الكود. فأثناء كتابتك للكود قد تواجهك بعض المهام التى ترغب فى تنفيذها فى أوقاتٍ أخرى. وهذه القوائم تعمل فى الواقع كمعلومات مفيدة لمطور التطبيق والأعضاء الجدد الذين يستخدمون هذا التطبيق حيث تحتوى على معلومات مفيدة عن حالة الكود الموجود داخل التطبيق. فإذا أردت التعرف على الكود المصاحب لأحد التلميحات الموجودة فى قائمة المهام، انقر هذا التلميح نقراً مزدوجاً (انظر شكل ٢-٦).



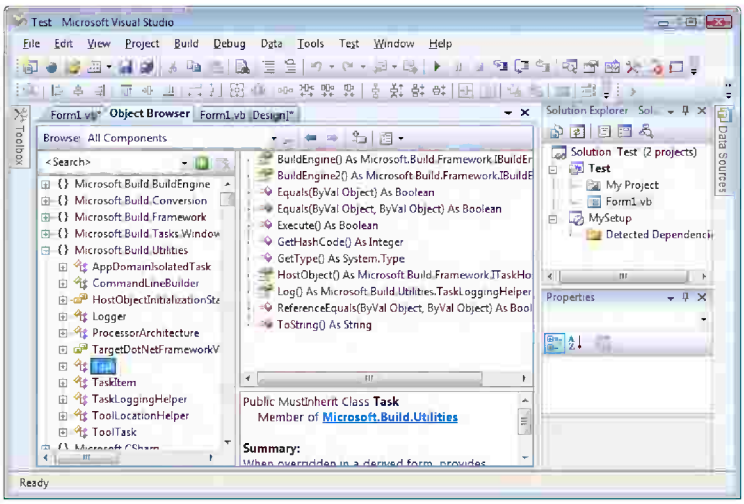
شكل ٢–٥ استخدام محرر صفحات الويب المرئي.



شكل ٢–٦ استخدام قوائم المهام لتعيين تلميحات لأجزاء معينة من الكود.

- مستعرض الكائنات **Object Browser** يقوم بإظهار خريطة بجميع الكائنات الموجودة بالنظام. وعلى الرغم من وجود هذا المستعرض بالإصدار السابق من بيئة التطوير، إلا أن مايكروسوفت قد أضافت إليه بعض التحسينات الجديدة حيث

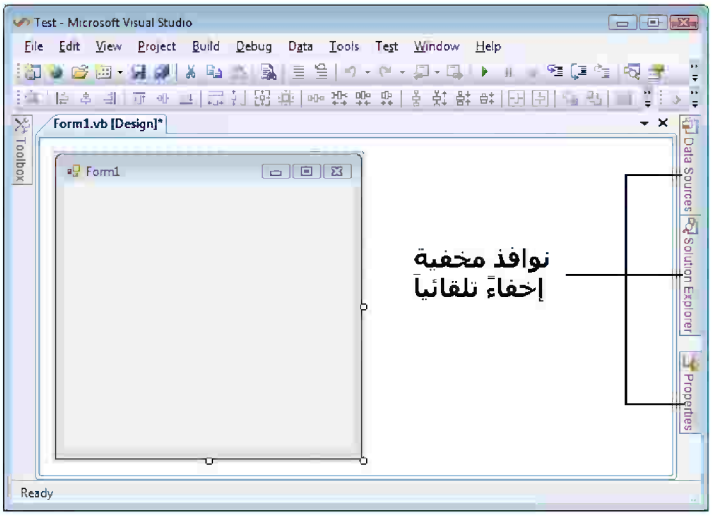
يمكنك الآن من داخل المستعرض البحث عن الكائنات وترتيبها أو حتى تجميعها (انظر شكل ٢-٧).



شكل ٢-٧ يقوم مستعرض الكائنات بإظهار معلومات عن جميع الكائنات الموجودة بالتطبيق بغض النظر عن اللغة المستخدمة فى إنشائها.

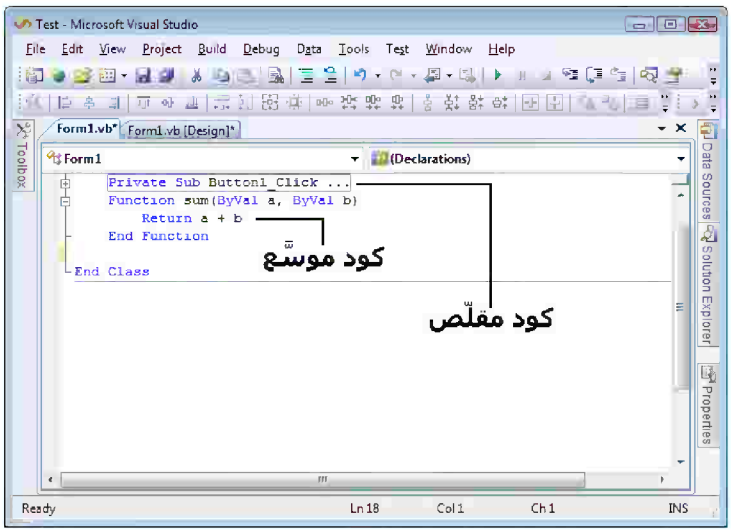
- النوافذ المبوبة **Tabbed Windows** والتى تتيح للنوافذ المفتوحة الرص فوق بعضها البعض عن طريق شريط من التبويبات يوجد أعلى النوافذ أو أسفلها. يمكنك استخدام هذا الشريط لتنشيط النافذة التى ترغب فى إظهارها (راجع شكل ٢-٧).
- سمة الإخفاء التلقائى للنوافذ **Auto-hide** ويمكنك من خلالها الاحتفاظ بأكبر مساحة للشاشة. وهذه السمة مفيدة جداً خاصةً مع زيادة كمية النوافذ والسمات الموجودة داخل بيئة التطوير الجديدة، حيث يمكنك من خلالها إظهار النوافذ حسب الحاجة. فإذا انتهيت على سبيل المثال من استخدام مربع الأدوات، يتم تقليص هذه النافذة إلى إحدى حواف الشاشة. فإذا أردت إظهارها مرةً أخرى، قم بتوجيه المؤشر إلى النافذة المقلصة، تلاحظ تمدد النافذة وظهورها بحجمها السابق (انظر شكل ٢-٨). لتطبيق سمة الإخفاء التلقائى على النافذة، انقر الزر المجاور لزر إغلاق النافذة حيث يتحول الزر إلى الرمز . أما إذا أردت تطبيق سمة الإخفاء

التلقائى على جميع النوافذ مرةً واحدة، افتح قائمة **Window** من شريط القوائم ثم اختر **Auto Hide All** من القائمة المنسدلة.



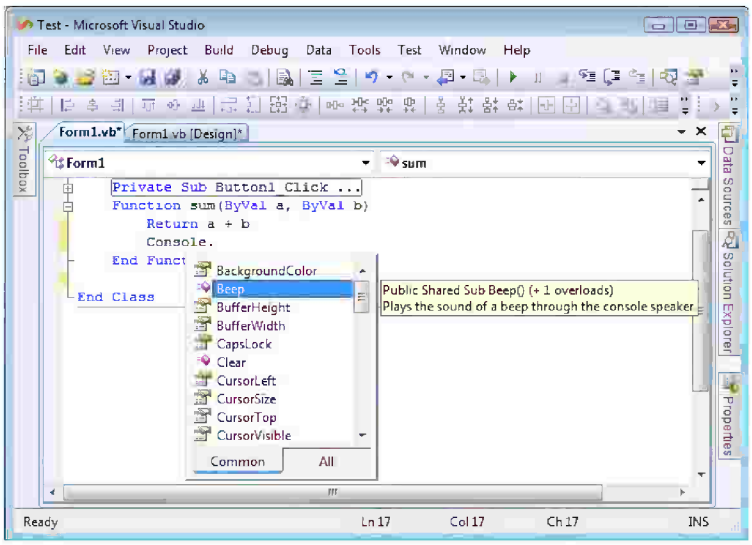
شكل ٢-٨ الإخفاء التلقائى للنوافذ.

- تقليص الكود **Code Collapsing** وهذه السمة تتيح لك تقليص أو توسيع جزء معين من الكود بحيث يمكنك التركيز على الجزء الذى أنت بصدده دون مشاهدة الجزء المتبقى من الكود. وهذه السمة مشابهة تماماً لتوسيع وتقليص المجلدات داخل مستكشف **Windows** (انظر شكل ٢-٩). لتعطيل هذه السمة، افتح قائمة **Edit** من شريط القوائم ثم اختر **Outlining** ثم **Stop Outlining** من القوائم المنسدلة. فإذا أردت تنشيطها مرةً أخرى، قم بتكرار نفس الطريقة ولكن مع اختيار **Collapse to Definitions**.

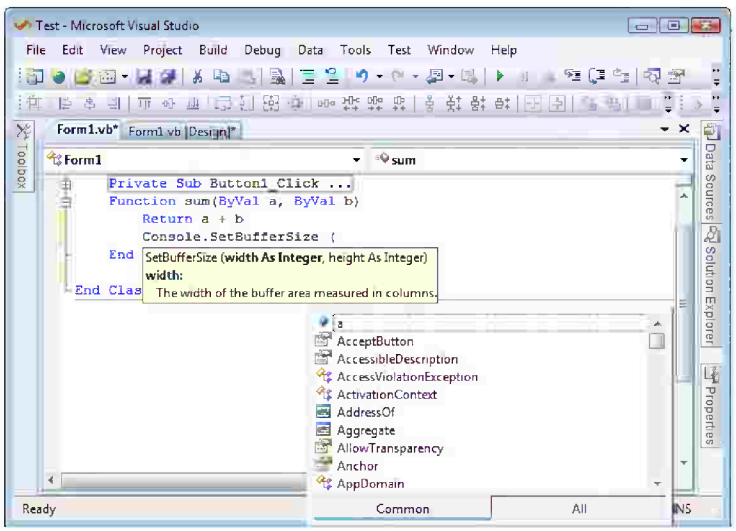


شكل ٢-٩ تقليص الكود وتوسيعه باستخدام سمة تقليص الكود.

- الحاسة الذكية **IntelliSense** تساعدك على تذكر الصيغة الصحيحة للدوال (الوظائف). فحينما تقوم بكتابة اسم أحد الكائنات، تقوم هذه السمة تلقائياً بإظهار قائمة بجميع الدوال (الوظائف) التى تخص هذا الكائن (انظر شكل ٢-١٠). وبمجرد اختيارك لإحدى هذه الدوال (الوظائف) ثم فتح قوس بداية الدالة (الوظيفة)، تظهر قائمة بالمعاملات المستخدمة مع هذه الدالة أو الوظيفة (انظر شكل ٢-١١).
- تقوم بيئة التطوير بالتأكد من صحة الكود الذى تقوم بكتابته لحظة بلحظة كما تقوم بإظهار شريط مضاء فوق الكود الغير صحيح كى تتيح لك فرصة تصحيح هذا الكود قبل عملية الترجمة.



شكل ٢-١٠ تظهر قائمة بالدوال (الوظائف) المستخدمة مع الكائن بمجرد كتابته.



شكل ٢-١١ تظهر قائمة بالمعاملات المستخدمة مع الدالة (الوظيفة) بمجرد كتابة قوس بداية الدالة.

ثالثاً: الجديد فى Visual Basic.NET/ 2005/2008

جميع السمات الجديدة التى شرحناها هنا ظهرت أساساً مع Visual Basic.Net وموجودة داخل Visual Basic 2005 وكذلك Visual Basic 2008، وقد ذكرناها فى هذا الفصل حتى يتعرف مستخدمو Visual Basic 6.0 وما قبله على الطفرة الهائلة التى تمت على اللغة بمجرد ظهور Visual Basic.Net.

تغييرات على مستوى الكود وأنواع البيانات

تم إجراء العديد من التغييرات على مستوى كتابة الكود وتعريف أنواع البيانات والتى يمكن إيجازها فيما يلى:

حدود المصفوفات

حينما تقوم بتعريف مصفوفة داخل Visual Basic 6.0، فإن السلوك الافتراضى للمصفوفة يتم داخل أول عنصر بالمصفوفة (العنصر 0). وبمعنى آخر فإن التعريف التالى:

```
Dim MyArray (3) As Integer
```

يقوم بتعريف مصفوفة تحتوى على أربعة عناصر هى MyArray(0) وMyArray(1) وMyArray(2) وMyArray(3). ولكن يمكنك استخدام العبارة التالية:

```
Option Base
```

كى تبدأ المصفوفة بالعنصر MyArray(1) بدلاً من MyArray(0). وبالتالى تحتوى المصفوفة MyArray(3) على ثلاثة عناصر فقط وليس أربعة.

أما فى Visual Basic.NET/2005/2008 فلم يعد هناك وجود للعبارة Option Base وبالتالى تبدأ جميع المصفوفات من الفهرس 0.

تغيير حجم المصفوفة

يمكنك فى Visual Basic 6.0 تعيين أعلى وأقل مدى للمصفوفة من خلال العبارة التالية:

```
Dim Sales(0 to 4) As Single
```

وحينئذٍ لا يمكنك فى أى وقت تغيير حجم المصفوفة باستخدام العبارة ReDim. أما Visual Basic.NET/2005/2008 فلا يدعم استخدام المصفوفات ثابتة الحجم حيث يمكنك تغيير حجم أى مصفوفة من خلال العبارة ReDim. فإذا قمت على سبيل المثال بتعريف مصفوفة باسم Sales تحتوى على أربعة عناصر وأردت زيادة حجمها لتحتوى على عشرة عناصر، يمكنك استخدام العبارة ReDim كما يلى:

```
ReDim Sales(10)
```

استخدام العبارة ReDim

بعكس Visual Basic 6.0، لا يتيح Visual Basic.NET/2005/2008 استخدام العبارة ReDim أثناء الإعلان عن المصفوفة، حيث يجب تعريف المصفوفة أولاً باستخدام Dim أو ما يكافئها مثل Public قبل استخدام العبارة ReDim.

سلاسل البيانات الثابتة

يمكنك فى Visual Basic 6.0 تعريف سلسلة بيانات ثابتة كما يلى:

```
Dim sLastName As String * 20
```

وذلك لتعريف سلسلة بيانات تحتوى على ٢٠ حرفاً. أما فى Visual Basic.NET/2005/2008، فيتم تعريف جميع سلاسل البيانات لتكون متغيرة الطول. ففى العبارة التالية:

```
Dim sLastName As String
```

يتم تعريف سلسلة البيانات sLastName التى يمكنها احتواء أى عدد من الحروف.

أنواع البيانات الصحيحة

تم إدخال بعض التعديلات على أنواع البيانات الصحيحة داخل Visual Basic.NET/2005/2008، يوضح الجدول التالى الفرق بين الأنواع الموجودة داخل Visual Basic 6.0 وVisual Basic.NET/2005/2008.

Visual Basic 6.0	Visual Basic.NET/ 2005/2008	المدى

Visual Basic 6.0	Visual Basic.NET/ 2005/2008	المدى
Integer	Short	٣٢.٧٦٨- إلى ٣٢.٧٦٧
Long	Integer	٢.١٤٧.٤٨٣.٦٤٨- إلى ٢.١٤٧.٤٨٣.٦٤٧
—	Long	٩.٢٢٣.٣٧٢.٠٣٦.٨٥٤.٧٧٥.٨٠٨- إلى ٩.٢٢٣.٣٧٢.٠٣٦.٨٥٤.٧٧٥.٨٠٧

نوع البيانات العام

فى Visual Basic 6.0 يتم استخدام المتغير العام Variant لتخزين أى نوع من أنواع البيانات، أما فى Visual Basic.NET/2005/2008 فيتم استخدام المتغير Object كمتغير عام.

نوع بيانات العملة

لا يدعم Visual Basic.NET/2005/2008 استخدام نوع بيانات العملة Currency، حيث تم استبداله بنوع البيانات Decimal الذى يحبذ استخدامه فى المعاملات المالية وذلك لاحتوائه على دقة عالية على جانبى النقطة العائمة (العشرية) حيث يدعم ما يقرب من ٢٨ مكان.

نوع بيانات التاريخ Date

فى Visual Basic 6.0 يقوم نوع البيانات Date بتخزين التاريخ باستخدام نوع البيانات Double الذى يحتوى على ٤ بايت. أما فى Visual

Basic.NET/2005/2008 فيتم استخدام نوع البيانات DateTime لتخزين التواريخ فى ٨ بايت.

إنشاء أنواع البيانات المعرفة

فى Visual Basic 6.0 يتم إنشاء أنواع البيانات المعرفة من قِبَل المستخدم باستخدام التركيب الآتى:

```
Type
.....
End Type
```

وقد تم استبدال هذا التركيب فى Visual Basic.NET/2005/2008 ليصبح هكذا:

```
Structure
.....
End Structure
```

تعريف أكثر من متغير

فى Visual Basic 6.0 إذا أردت تعريف أكثر من متغير فى نفس السطر، يجب أن تقوم صراحةً بتعريف نوع بيانات كل متغير على حده حتى وإن كان لجميع المتغيرات نفس نوع البيانات وإلا يكون نوع البيانات هو النوع العام Variant. فالعبارة التالية على سبيل المثال:

```
Dim a,b,c As Integer
```

تقوم بتعريف المتغيرين a وb من النوع Variant والمتغير c من النوع Integer.

أما فى Visual Basic.NET/2005/2008 فيمكنك تعريف أكثر من متغير من نفس النوع فى نفس السطر دون الحاجة إلى تكرار نوع البيانات. فالعبارة السابقة على سبيل المثال، تقوم بتعريف ثلاثة متغيرات (حقول) من النوع Integer وهى a وb وc.

كما يمكنك أيضاً تعريف أكثر من متغير بأنواع بيانات مختلفة فى نفس السطر. فالعبارة التالية:

```
Dim a,b As Integer, str As String
```

تقوم بتعريف المتغيرين a وb من النوع Integer والمتغير str من النوع String.

مدى المتغيرات

أضافت Visual Basic.NET/2005/2008 مدىً جديداً إلى مجالات رؤية المتغير، ألا وهو مدى المتغير على مستوى جزء من الكود يسمى Block. فإذا أردت أن يكون المتغير متاحاً فقط داخل جزء من الكود مثل الدوارة For ... Next أو عبارة If ... End If، يمكنك بسهولة تامة تعريف هذا المتغير داخل هذا الجزء. ففى الكود التالى على سبيل المثال:

```
If x > 120 Then
Dim sMsg As String
sMsg = "Please enter a smaller value."
MessageBox.Show(sMsg)
End If
```

تم تعريف المتغير sMsg داخل عبارة If ... End If، لذا يكون متاحاً فقط داخل هذه العبارة. لذا فإذا قمت بتعديل العبارة السابقة كما يلى:

```
If x > 120 Then
Dim sMsg As String
sMsg = "Please enter a smaller value."
End If
MessageBox.Show(sMsg)
```

يقوم المترجم بالاعتراض على الفور لأن المتغير sMsg غير متاح خارج نطاق العبارة If.

استهلال المتغيرات بالقيم الابتدائية

يمكنك فى Visual Basic.NET/2005/2008 تعيين قيم ابتدائية للمتغيرات (استهلال المتغيرات) أثناء الإعلان عن هذه المتغيرات. ففى العبارة التالية على سبيل المثال:

```
Dim sFirstName As String = "Waleed"
```

يتم تعريف المتغير sFirstName مع تخصيصه بسلسلة البيانات "Waleed" وهذا لم يكن متاحاً فى Visual Basic 6.0.

العبارة While ... Wend

تستخدم العبارة While ... Wend فى Visual Basic 6.0 لتكرار تنفيذ مجموعة من العبارات فى حالة تحقق شرط معين. وهذه العبارة لم تزل موجودة داخل Visual

Basic.NET/2005/2008 ولكن مع استبدال كلمة Wend بعبارة End While فلم يعد هناك تدعيم لكلمة Wend.

تغييرات على مستوى الإجراءات

تم إدخال التغييرات الآتية على أسلوب عمل الإجراءات وذلك كما يلى:

إجراءات الخصائص

حينما تقوم بتعريف إجراء خاصية Property Procedure داخل Visual Basic 6.0، يتم إنشاء ثلاثة إجراءات منفصلة، الإجراء Get والإجراء Let والإجراء Set. أما فى Visual Basic.NET/2005/2008، فيتم إنشاء إجراء واحد فقط يحتوى على الخيارين Get وSet التى تقوم بنفس وظيفة مثيلاتها فى Visual Basic 6.0.

تمرير المعاملات إلى الإجراءات

فى Visual Basic 6.0 يمكنك تمرير المعاملات إلى الإجراءات الفرعية بدون استخدام الأقواس إذا لم تقم باستخدام كلمة Call فى عملية الاستدعاء هكذا:

```
ProcEmployee "Waleed Abdelrazek",98
```

أما فى Visual Basic.NET/2005/2008 فمن الضرورى تضمين الأقواس فى عملية الاستدعاء.

إرجاع القيم من الدوال

فى Visual Basic 6.0، يتم إرجاع القيم من الدوال عن طريق تخصيص القيمة المرجعة إلى اسم الدالة كما فى الكود التالى:

```
Function AddNums(ByVal a As Single, ByVal b As Single)
Dim x As Single
x = a + b
AddNums = x
End Function
```

وعلى الرغم من أن هذه التقنية لم تزل تعمل بشكل صحيح إلا أنه من الأفضل استخدام العبارة Return لتعيين القيمة المرجعة من الدالة كما يلى:

```
Function AddNums(ByVal a As Single, ByVal b As Single)
```

```
        Dim x As Single
        x = a + b
        Return x
End Function
```

التمرير الافتراضى للمعاملات

يمكنك تمرير المعاملات إلى الدالة بقيمتها **ByValue** كما يمكنك تمريرها كمرجع **ByRef**. وفى **Visual basic 6.0**، إذا قمت بتمرير معامل دون تعين نوع هذا التمرير، هل هو **ByVal** أم **ByRef**، يتم اعتباره **ByRef**. أما فى **Visual Basic.NET/2005/2008** فيتم اعتباره **ByVal** وهذا لحماية المعاملات من التغيير داخل الإجراء الذى يتم تمريرها إليه. فإذا أردت إتاحة تغييرها بالفعل، قم صراحةً بتعريف المعامل ليكون **ByRef**.

تغييرات على ملفات المشروع

حينما تقوم بحفظ أحد المشروعات فى **Visual Basic 6.0**، يتم تخزين كل نموذج فى ملف مستقل بالامتداد **frm**. كما يتم تخزين كل تصنيف فى ملف مستقل بالامتداد **cls**. ويتم تخزين المشروع بالكامل داخل ملف بالامتداد **vbp**. ويمكنك تجميع أكثر من مشروع فى ملف واحد بالامتداد **grp**.. أما فى **Visual Basic.NET/2005/2008**، فيتم تخزين النماذج والتصنيفات داخل ملفات بالامتداد **vb**. وتخزين هذه الملفات داخل مشروع بالامتداد **vbproj**.، كما يتم تخزين مشروع أو أكثر داخل "حل" **Solution** بالامتداد **sln**.

تغييرات على مستوى أدوات التحكم

تم تغيير الطريقة التى يعمل بها بعض الأدوات فى **Visual Basic.NET/2005/2008** عما كانت تعمل به فى **Visual Basic 6.0**. وفيما يلى نقوم بإلقاء الضوء على هذه التغييرات.

الخصائص الافتراضية

تدعم معظم الأدوات الموجودة فى Visual Basic 6.0 استخدام الخصائص الافتراضية. بمعنى أنك إذا قمت بكتابة اسم الأداة فقط دون كتابةٍ أى من خصائصها، يتم افتراض أنك استخدمت القيمة الافتراضية. وخير مثال على ذلك مربع النص TextBox الذى يحتوى على الخاصية الافتراضية Text ، حيث يمكنك فى Visual Basic 6.0 تخصيص القيمة "Waleed" للخاصية Text بمربع النص TextBox1 كما يلى:

```
TextBox1 = "Waleed"
```

أى بدون كتابة اسم الخاصية. أما فى Visual Basic.NET/2005/2008 فلم يعد هناك تدعيم للخاصية الافتراضية، ويجب عليك كتابة العبارة كاملة كما يلى:

```
TextBox1.Text = "Waleed"
```

استبدال الخاصية Caption بالخاصية Text

تستخدم الخاصية Caption فى Visual Basic 6.0 لتعيين النص الذى يظهر على أداة العنوان أو شريط عنوان النموذج أو على واجهة زر الأمر. أما فى Visual Basic.NET/2005/2008، فتم استبدال الخاصية Caption بالخاصية Text.

تعديلات زر الأمر

تم استبدال اسم زر الأمر Command Button الموجود بالإصدارات السابقة من Visual Basic بالزر Button والذى يحتوى على جميع الوظائف الموجودة بالزر عدا الخاصية Caption التى تم استبدالها كما ذكرنا بالخاصية Text.

استبدال أداة الإطار Frame Control

تم استبدال أداة الإطار Frame Control الموجودة فى Visual Basic 6.0 والمستخدمة كحاوية لمجموعة من الأدوات، بمربع المجموعة GroupBox واللوحة Panel. والفرق بين الأداتين أن أداة مربع المجموعة GroupBox تحتوى على إطار خارجى واضح وتحتوى على عنوان، أما أداة اللوحة Panel، فلا تظهر للمستخدم وقت التشغيل ولا يمكن أن تحتوى على عنوان وإنما تحتوى على وظائف تجميع الأدوات.

أداة المؤقت Timer

فى Visual Basic 6.0، يتم تنفيذ كود مرور الوقت المحدد داخل المؤقت داخل الحدث Timer، أما فى Visual Basic.NET/2005/2008 فتم استبدال هذا الحدث بالحدث Tick، الذى يتم تنفيذه على فترات منتظمة محددة داخل الخاصية Interval.

أداة القائمة MenuStrip Control

فى الإصدارات السابقة من Visual Basic، كنا نستخدم محرر القوائم Menu Editor فى إنشاء القوائم وتضمينها داخل نماذجك. أما Visual Basic.NET/2005/2008 فيحتوى على أداة القائمة MenuStrip التى يمكنك إضافتها إلى النموذج من مربع الأدوات مثلها كبقية الأدوات الأخرى وإنشاء عناصرها بسهولة متناهية.

الخاصيتان Anchor و Dock

تم إضافة خاصيتين إلى أدوات التحكم المختلفة وهما الخاصية Anchor والخاصية Dock التى تتيح لك وضع الأدوات على النماذج بالطريقة التى تضمن بقائها كما هى حتى وإن تم تغيير حجم النموذج.