الفصل السابع عشر
مراقبة كفاءة التطبيق

تتوقف كفاءة التطبيق على سرعة تنفيذ أجزاءه والتى من أهمها عمليات الاستدعاء المختلفة للوظائف والدوال والإجراءات. يمكنك التأكد من كفاءة تطبيقك قبل إعداده للتوزيع ونشره إلى مستخدميك. سنقوم فى هذا الفصل بالتعرف على كيفية قياس كفاءة التطبيق ومراقبة عدادات الكفاءة الموجودة بنظام التشغيل والتحكم فيها من خلال كود Visual Basic 2008.

بانتهاء هذا الفصل، ستتعرف على:

- *قياس زمن التنفيذ.*
- *استخدام عدادات الكفاءة.*
- *قراءة البيانات من عدادات الكفاءة.*
- *الكتابة فى عدادات الكفاءة.*

على الرغم من عمل تطبيقك بشكل سليم والحصول منه على النتائج المرجوة، إلا أن هذا لا يعنى بالفعل نجاح التطبيق، فهناك عامل سرعة التنفيذ الذى يجعل أحد التطبيقات يتفوق على تطبيق آخر. ويمكنك كمبرمج أداة مهمة معينة من خلال عدة طرق مختلفة، وحينئذٍ يمكنك اختيار الطريقة المثلى من خلال قياس عاملين هامين هما سرعة التنفيذ ومساحة التخزين المستخدمة. وعلى الرغم من تعارض أحد العاملين غالباً مع العامل الآخر، إلا أن عامل السرعة يكون له نصيب الأسد، لأن المستخدم لا يشعر كثيراً بكبر مساحة التخزين المستخدمة أو صغرها بقدر ما يشعر بسرعة تنفيذ التطبيق والتى من خلالها يتم الحكم على كفاءة التطبيق ككل. من أجل ذلك، يجب أن تقوم باستخدام عملية ضبط الكفاءة **Performance Tuning** داخل تطبيقاتك وهو ما يعنى تسريع عمل هذه التطبيقات. ومن الأمثلة العملية على ضبط الكفاءة، تغيير استعلامات قاعدة البيانات وحذف عمليات الاستدعاء أو تخزين البيانات الغير ضرورية. وتظهر أهمية ضبط الكفاءة حقيقةً عند العمل مع تطبيقات الإنترنت متعددة الصفوف، وذلك لأن كل طلب لصفحة الويب ربما يقوم بإنشاء أحد كائنات **Visual Basic** والذى يقوم بدوره بإنشاء كائنات أخرى أو تحديث قاعدة بيانات، وفى مثل هذه الحالات يكون هناك أكثر من عامل يتحكم فى كفاءة عمل التطبيق مثل سرعة نقل البيانات من خلال الشبكة وزمن تنفيذ كود **Visual Basic** والإجراءات المخزنة أو الاستعلامات الموجهة لقاعدة البيانات. ولكن كيف تستطيع قياس سرعة تنفيذ كل جزء من أجزاء التطبيق ؟ سنقوم فى هذا الفصل بالإجابة على هذا السؤال من خلال وسيلة للتعرف على متوسط سرعة استدعاء الدوال والوظائف. كما سنتعرف أيضاً على عدادات الكفاءة **Performance Counters** التى يمكنك استخدامها لتتعرف على أجزاء التطبيق كثيرة الاستخدام، ومن ثَمَّ يمكنك استخدام حاستك كمبرمج للتعرف على الأجزاء التى يجب التركيز عليها للحصول على أعلى كفاءة للتطبيق. كانت هذه خلاصة الفصل، والآن إلى التفاصيل.

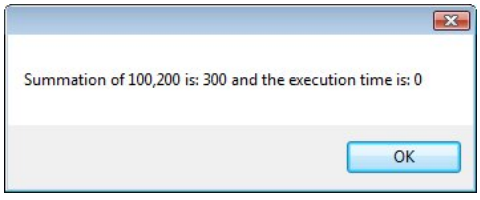
قياس زمن التنفيذ

يحتوى المسمى **System.Environment** على الخاصية **TickCount** التى تقوم بإرجاع الوقت بالملى ثانية نسبةً إلى الوقت الحالى للحاسب. من ثمَّ يمكنك استخدام هذه الخاصية للتعرف على زمن استدعاء الوظيفة أو الدالة. للتعرف على استخدام هذه الخاصية، قم بإنشاء تطبيق نوافذى جديد ثم قم بإدخال الكود التالى إلى تصنيف النموذج **Form1**:

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles MyBase.Load
  Dim intStartTime As Integer
  Dim intEndTime As Integer
  Dim intTimeToExecute As Integer
  Dim Result As Integer
  intStartTime = System.Environment.TickCount
  Result = sum(100, 200)
  intEndTime = System.Environment.TickCount
  intTimeToExecute = intEndTime - intStartTime
  MessageBox.Show("Summation of 100,200 is: " & Result & "
            and the execution time is: " & intTimeToExecute)
End Sub

Function sum(ByVal a, ByVal b)
    Return a + b
End Function
```

يتم فى هذا الكود تخزين قيمة الخاصية **System.Environment.TickCount** قبل وبعد استدعاء الدالة **()Sum**. وبطرح القيمتين، نحصل على زمن التنفيذ بالملى ثانية، وأخيراً نقوم بطباعة هذه القيمة على الشاشة. قم بتنفيذ التطبيق، تحصل على مربع رسالة يخبرك بالوقت المستغرق فى استدعاء الدالة **()Sum** والحصول على مجموع القيمتين **100** و**200** (انظر شكل ١٧-١).



شكل ١٧-١ الحصول على زمن استدعاء الدالة

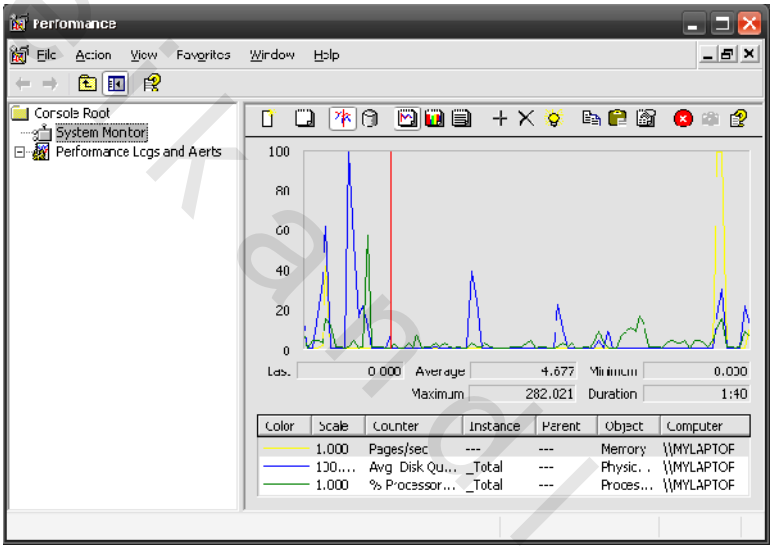
وعلى الرغم من أن الكود اللازم لقياس زمن تنفيذ إحدى عمليات الاستدعاء فى غاية السهولة كما ترى، إلا أن طباعة زمن التنفيذ على الشاشة للتطبيقات الكبيرة التى تحتوى على المئات من عمليات الاستدعاء غير عملى بالمرة. وإنما يمكنك فى هذه الحالة إنشاء جدول داخل قاعدة بيانات صغيرة وتخزين زمن استدعاء كل دالة داخل سجل بهذا الجدول حتى يمكنك تحليلها والتعرف على عمليات الاستدعاء التى تستغرق المزيد من الوقت حتى يتم معالجتها قبل توزيع تطبيقك إلى المستخدمين.

استخدام عدادات الكفاءة

تعرفنا فى الفقرة السابقة على كيفية استخدام الخاصية **TickCount** لقياس زمن تنفيذ أو استدعاء الدوال الموجودة بالتطبيق. ولكن عند رفع كفاءة التطبيق، تجد أن هناك العديد من العوامل التى تحدد الخطوات المستخدمة فى رفع هذه الكفاءة. فعلى سبيل المثال، يجب أن تعرف الدوال المستخدمة بكثرة داخل التطبيق وزمن تنفيذها. فعلى فرض أنك تستخدم تطبيقاً لإدخال البيانات **Data Entry** وتستغرق عملية تحميل هذا التطبيق عشر ثوانى، كما تستغرق عملية حفظ كل سجل جديد عشر ثوانى أيضاً، أى أن زمن التحميل مساوى لزمن حفظ كل سجل، فلا شك أن الأولية تكون لرفع كفاءة كود الزر **Save** المستخدم فى عملية الحفظ.

يحتوى نظام تشغيل **Windows** على عدادات الكفاءة **Performance Counter** التى يمكنك استخدامها للتعرف على ما يحدث بحاسبك، حيث يمكنك مشاهدة عدد مرات تنفيذ جزء معين من الكود أو مراقبة أحد موارد النظام مثل الذاكرة أو طلبات الإدخال والإخراج. ويعمل العداد فى **Windows** بنفس طريقة عداد السرعة داخل السيارة حيث

يحتوى كل منهما على قيمة تزداد وتنقص تبعاً لحالة معينة بالنظام. كما يحتوى Windows أيضاً على أداة تسمى "مراقب الكفاءة" Performance Monitor والتى يمكنك من خلالها مشاهدة عدادات النظام فى صورة رسومية. لفتح نافذة الكفاءة Performance، افتح لوحة التحكم Control Panel ثم قم بفتح المجموعة Administrative Tools ومنها انقر رمز Performance نقراً مزدوجاً (انظر شكل ١٧-٢).



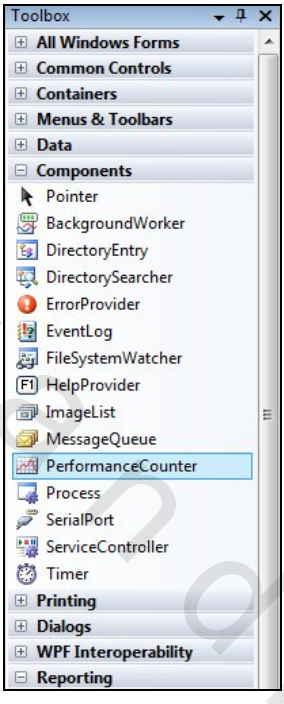
شكل ١٧-٢ مشاهدة عدادات النظام من خلال نافذة Performance

سنقوم فيما يلى بتوضيح كيفية قراءة البيانات الموجودة بعدادات الكفاءة بنظام التشغيل من خلال كود Visual Basic 2008 وكيفية إضافة عداد كفاءة إلى النظام، وذلك من خلال مثال عملى بسيط.

قراءة البيانات من عدادات الكفاءة

يعتبر استخدام أداة التحكم PerformanceCounter أسهل الطرق للتعامل مع عدادات الكفاءة الموجودة بنظام التشغيل، حيث توجد هذه الأداة داخل الجزء Components بمربع الأدوات (انظر شكل ١٧-٣).

ويعتبر استخدام الأداة **PerformanceCounter** أحد الطرق السهلة للحصول على معلومات عن الموارد المختلفة المرتبطة بالنظام مثل المعالج والذاكرة وغيرها. ولكى تختار أحد عدادات النظام، يجب أن تقوم بتعيين ثلاثة خصائص مصاحبة لهذه الأداة وهى:

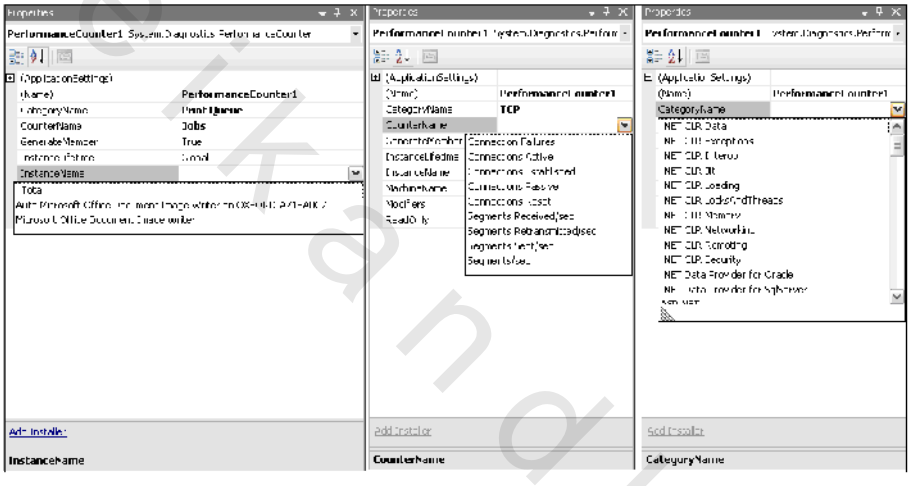


شكل ١٧-٣ الأداة **PerformanceCounter** داخل مربع الأدوات

- *الخاصية* ***CategoryName*** وتحتوى على مجموعة من العدادات المرتبطة. فالمجموعة **Memory** على سبيل المثال، تحتوى على مجموعة من العدادات المرتبطة بعمل الذاكرة على حاسبك.
- *الخاصية* ***CounterName*** وتحتوى على العداد الذى ترغب فى اختباره أو مراقبته مثل عدد البايت المتاحة داخل الذاكرة **Available Bytes** فى حالة تخصيص القيمة **Memory** للخاصية **CategoryName**.

- الخاصية ***InstanceName*** وتحتوى على حالة من عداد الكفاءة. فعلى سبيل المثال إذا كنت تستخدم جهازاً يحتوى على أربعة معالجات، فكل من هذه المعالجات يمثل حالة من العدادات المرتبطة بعمل المعالج داخل النظام، وهذه الخاصية اختيارية مع عدد من العدادات.

وعند تعيين هذه الخواص، يتم الاختيار من بين مجموعة من الاختيارات المتاحة والتى تمثل العدادات الموجودة بنظام التشغيل (انظر شكل ١٧-٤).



شكل ١٧-٤ يوجد العديد من الخيارات التى يمكنك الاختيار من بينها

لتعيين أحد عدادات كفاءة النظام، تابع معنا الخطوات الآتية:

١. من داخل بيئة تطوير **Visual Studio 2008**، قم بإنشاء تطبيق نوافذى جديد باسم مناسب وليكن **Performance**.

٢. قم بإضافة الأداة PerformanceCounter من الجزء **Component** بمربع الأدوات إلى النموذج مع تعيين القيمة **TCP** للخاصية **CategoryName** والقيمة **Connections Established** للخاصية **CounterName** من قائمة الخيارات المتاحة لكلٍ من الخاصيتين.

٣. قم بإضافة أداة المؤقت Timer من الجزء نفسه إلى النموذج ثم قم بتعيين القيمة 1500 للخاصية Interval والقيمة True للخاصية Enabled.

٤. قم بإضافة مربع نص من مربع الأدوات إلى النموذج.

٥. قم بإدخال الكود التالى داخل إجراء الحدث Elapsed الخاص بالمؤقت:

```
Private Sub Timer1_Elapsed(ByVal sender As Object, ByVal e As
System.Timers.ElapsedEventArgs) Handles Timer1.Elapsed

    TextBox1.Text = PerformanceCounter1.RawValue.ToString
End Sub
```

فبمجرد إعداد الأداة PerformanceCounter، يمكنك الوصول إلى قيمة العداد المصاحب من خلال الخاصية RawValue. فيمكنك على سبيل المثال مراقبة الذاكرة المتاحة أو مساحة التخزين المتاحة على القرص الصلب وإرسال رسالة إلى مسئول الشبكة أو النظام بمجرد وصولها لقيمة معينة.

٦. قم بتنفيذ التطبيق، تلاحظ احتواء مربع النص على عدد الاتصالات الحالية لحاسبك بالشبكة.

٧. قم بإجراء عدد من الاتصالات الأخرى عن طريق فتح صفحات مختلفة على الخادم Localhost أو من خلال الإنترنت، تلاحظ تغيير قيمة مربع النص لتعكس العدد الفعلى للاتصالات مع ملاحظة تغيير هذه القيمة كل ١٥ ثانية.

الكتابة فى عدادات الكفاءة

بالإضافة إلى الوصول إلى عدادات الكفاءة الموجودة مسبقاً لنظام التشغيل، يمكنك إضافة عداد كفاءة مخصص من خلال بعض سطور الكود البسيطة. وبمجرد إضافة هذا العداد، يكون متاحاً داخل نافذة مراقبة الكفاءة الموجودة بنظام التشغيل. للتعرف على كيفية إضافة عدادات جديدة، سنقوم بإنشاء مجموعة عدادات جديدة باسم VBAppCounters تحتوى على عدادين باسم TestCounter1 وTestCounter2. لأداء ذلك، تابع معنا الخطوات الآتية:

١. قم بإضافة ثلاثة أزرار من مربع الأدوات إلى نموذج التطبيق Performance الذى بين أيدينا.

٢. قم بتغيير أسماء الأزرار الثلاثة إلى btnAdd وbtnSubtract وbtnClear وتغيير عناوينها إلى Add وSubtract وClear على الترتيب.

٣. قم بإضافة الكود التالى إلى تصنيف النموذج Form1:

```
Private TestCtr1 As PerformanceCounter
Private TestCtr2 As PerformanceCounter

Private Sub InitCustomCounters()

    Dim CCD As CounterCreationDataCollection
    Dim CNT As CounterCreationData

     If Not PerformanceCounterCategory.Exists
                                    ("VBAppCounters") Then
        CCD = New CounterCreationDataCollection()
        CNT = New CounterCreationData("TestCounter1", "test.",
        PerformanceCounterType.NumberOfItems32)
        CCD.Add(CNT)
        CNT = New CounterCreationData("TestCounter2",
        "test2." , PerformanceCounterType.
        RateOfCountsPerSecond32)
       CCD.Add(CNT)

       PerformanceCounterCategory.Create("VBAppCounters",
                                         "Test Category", CCD)
     End If

   TestCtr1 = New PerformanceCounter("VBAppCounters",
                                         "TestCounter1", False)
   TestCtr2 = New PerformanceCounter("VBAppCounters",
                                         "TestCounter2", False)
End Sub

Private Sub btnAdd_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnAdd.Click
```

```
        InitCustomCounters()
        TestCtr1.Increment()
        TestCtr2.Increment()

End Sub

Private Sub btnSubtract_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnSubtract.Click

        InitCustomCounters()
        TestCtr1.Decrement()
        TestCtr2.Decrement()
End Sub

Private Sub btnClear_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnClear.Click

        InitCustomCounters()
        TestCtr1.RawValue = 0
        TestCtr2.RawValue = 0
End Sub
```

٤. قم بتنفيذ التطبيق ثم انقر زر **Add** لإضافة مجموعة العدادات الجديدة إلى تلك الموجودة بنظام التشغيل.

٥. افتح نافذة لوحة التحكم **Control Panel** ثم افتح مجموعة **Administrative Tools** ومنها انقر رمز **Performance** نقراً مزدوجاً، تظهر النافذة **Performance** (راجع شكل ١٧-٢).

٦. انقر المجموعة **System Monitor** من العمود الأيسر بالنافذة ثم انقر أى مكان بالعمود الأيمن بزر الفأرة الأيمن ثم اختر **Add Counters** من القائمة الموضعية، يظهر المربع الحوارى **Add Counters** (انظر شكل ١٧-٥).

٧. نشط زر الاختيار **Use local computer counters** ثم اختر المجموعة التى قمنا بإضافتها **VBAppCounters** من مربع السرد **Performance Object**،

تلاحظ ظهور العدادان **TestCounter1** و**TestCounter2** داخل مربع العدادات المتاحة.

٨. قم بتنشيط زر الاختيار **All Counters**.

٩. انقر زر **Add** ثم زر **Close** لإغلاق المربع الحوارى.

شكل ١٧-٥ المربع الحوارى **Add Counters**

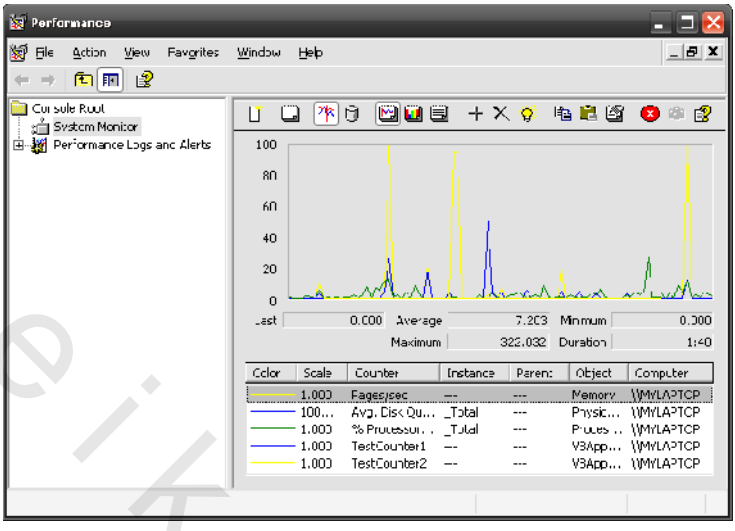
١٠. تلاحظ ظهور خط أحمر رأسى يتحرك رويداً رويداً بعرض الشاشة. كما يظهر اسم العدادين الجديدين بالقائمة الموجودة أسفل النافذة (انظر شكل ١٧-٦).

١١. قم بترتيب نموذج التطبيق والنافذة **Performance** على الشاشة حتى تتمكن من مشاهدتهما معاً.

١٢. انقر زر **Add** عدة مرات متعاقبة ولاحظ التغييرات داخل العدادين الجديدين.

١٣. قم بنقر زر **Subtract** لتقليل قيمة العدادين أو زر **Clear** لتخصيص القيمة **0** إليهما.

وبذلك قد تعرفنا على كيفية إنشاء عداد كفاءة جديد وتغيير قيمته، حيث يعمل كائن عداد الكفاءة كبقية الكائنات التى تقوم بإنشائها داخل **Visual Basic 2008** حيث يمكنك معالجته والتعامل معه من خلال الوظائف والخصائص المختلفة المرتبطة بهذا الكائن. ولكن يجب أن تضع النقاط التالية نصب عينيك عند استخدام عدادات الكفاءة:

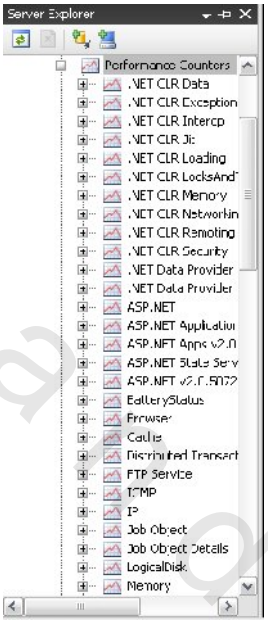


شكل ١٧-٦ إضافة العدادات الجديدة إلى شاشة المراقبة

- عندما تقوم بإنشاء أحد عدادات الكفاءة المخصصة، يجب أن تقوم بإنشاء المجموعة وعناصر هذه المجموعة من عدادات فى نفس الوقت كما فعلنا فى الكود السابق حينما قمنا بإنشاء المجموعة أولاً ثم إنشاء جميع العدادات من خلال الوظيفة **Create()**.
- يوجد أنواع مختلفة من العدادات، ففى الخطوات السابقة، حينما تقوم بنقر زر **Add**، تلاحظ إضافة العدادين الجديدين برسوم مختلفة. فقد قمنا على سبيل المثال باستخدام الثابت **RateOfCountersPerSecond32** مع العداد **TestCounter2** لقياس الكفاءة كل ثانية، ويمكنك بالطبع اختيار ثوابت أخرى.
- عند تعريف حالة من الكائن **PerformanceCounter** الذى ترغب فى تغيير قيمته، قم بتعيين قيمة الخاصية **ReadOnly** أثناء إعطاء القيم الابتدائية للكائن (أثناء استهلال الكائن) وإلا فلن يكون له أى تأثير.

النقطة الأخيرة التى نرغب فى ذكرها فى هذا الفصل هى إمكانية إنشاء عدادات الكفاءة والوصول إليها أو حتى حذفها من خلال بيئة تطوير **Visual Studio 2008** نفسها وذلك باستخدام مستكشف الخادم **Server Explorer**.

يوضح شكل ١٧-٧ قائمة عدادات الكفاءة الموجودة بنافذة مستكشف الخادم.



شكل ١٧-٧ التحكم فى عدادات الكفاءة من خلال نافذة مستكشف الخادم