```
                                    m.CC = 600 And _
                                    m.weight > 300 And _
                                    m.weight < 400

    If check(motorcycle) Then
        ' do something here
    End If

    ' do something here

    If check(motorcycle) Then
        ' do something here
    End If
End Sub
```

قمنا هنا بتعديل منطق تفحص بعض شروط Motorcycle ليستخدم تعابير لمدا عوضا عن سيئات الطرائق الخاصة حيث سيقوم المترجم تلقائيا بإنشاء النوع المفوض ويقوم بالعمل لكي نستطيع استدعاء تعابير لمدا أينما احتاج ذلك وهذه الطريقة مفيدة لأنها تضع المنطق قريب من التصريح حيث نقوم بتصنيع نسخة واحدة ويقوم المترجم بعدها بمعظم عمليات الصيانة ويعتبر هذا مفيدا لأنه يمكنك من بناء تعبير معقد كجسم لتعبير لمدا وباستخدام الربط المتأخر والاستدلال على النوع في هذا السيناريو فلا نحدد نوع تعبير لمدا أو المتغير

```
Dim lambda = Function(x) x * x
```

وهنا أيضا يولد المعالج مفوض مجهول من أجلك ولكن يحدد نوع تعبير لمدا كـ System.Object وهذا يعني أنه قد تم تفعيل الربط المتأخر في هذا السيناريو عندما يكون الخيار Option Strict على الوضع Off ويعتبر هذا السيناريو جيدا بالنسبة لأولئك الذين يعتمدون على الربط المتأخر حيث أن تعابير لمدا تدعم عمليات الربط المتأخر بشكل كامل ففي المثال السابق طالما أن المعامل * معرف على الأنواع الممررة إلى تعبير لمدا فسوف يعمل

```
Dim a = lambda(10)
Dim b = lambda(CDec(10))
Dim c = lambda("This will throw an exception because " & _
                "strings don't support the * operator")
```

وكما ترى من المثال السابق طالما أن المعامل * موجود في مكتبات زمن التشغيل بالنسبة للنوع الممرر فسوف يجري كل شئ بشكل جيد كما أن تعابير لمدا تتألق بشكل رائع مع الربط المتأخر في فيجول بايزيك.

الكود المولد تحت الغطاء

بعدما استكشفنا تعابير لمدا دعنا نلقي نظرة على الكود الذي يتم توليده من قبل المترجم. انظر للكود السابق

```
Sub TestLambda()
    Dim doubleIt As Func(Of Integer, Integer) = _
            Function(x As Integer) X * 2
    Console.WriteLine(doubleIt(10))
End Sub
```

أنت تعلم أن Func هو مفوض والمفوضات هي مؤشرات للوظائف فكيف يقوم المترجم إذا بالعمل؟ في هذه الحالة يقوم المترجم بإصدار وظيفة جديدة ويربطها بمفوض يشير إلى تلك الوظيفة الجديدة

```
Private Function $GeneratedFunction$(x As Integer) As Integer
    Return x * 2
End Function

Sub TestLambda()
    Dim doubleIt As Func(Of Integer, Integer) = _
        AddressOf $GeneratedFunction$
```