



C-Sharpener For VB
Limitations and Workarounds
Version 2.2

Copyright © Elegance Technologies, Inc. 2003-2006
www.elegancetech.com

Table Of Contents

1	INTRODUCTION	3
2	OPTION EXPLICIT AND OPTION STRICT	3
2.1	LIMITATION DESCRIPTION	3
2.2	VB EXAMPLE CODE	3
2.3	MANUAL WORK-AROUND	3
3	PREPROCESSOR DIRECTIVES	3
3.1	LIMITATION DESCRIPTION	3
3.2	VB EXAMPLE CODE	3
3.3	MANUAL WORK-AROUND	3
4	COMPILATION OPTIONS.....	4
4.1	LIMITATION DESCRIPTION	4
4.2	VB EXAMPLE CODE	4
4.3	MANUAL WORK-AROUND	4
5	SHADOWS KEYWORD	4
5.1	LIMITATION DESCRIPTION	4
5.2	VB EXAMPLE CODE	4
5.3	MANUAL WORK-AROUND	4
6	EMBEDDED ENUMERATIONS/INTERFACES.....	5
6.1	LIMITATION DESCRIPTION	5
6.2	VB EXAMPLE CODE	5
6.3	MANUAL WORK-AROUND	5
7	UNSTRUCTURED EXCEPTION HANDLING.....	5
7.1	LIMITATION DESCRIPTION	5
7.2	VB EXAMPLE CODE	5
7.3	MANUAL WORK-AROUND	6
8	MYCLASS KEYWORD	6
8.1	LIMITATION DESCRIPTION	6
8.2	VB EXAMPLE CODE	6
8.3	MANUAL WORK-AROUND	7
9	NAMED ARGUMENT	7
9.1	LIMITATION DESCRIPTION	7
9.2	VB EXAMPLE CODE	7
9.3	MANUAL WORK-AROUND	7
10	PROJECT FILES	8
10.1	LIMITATION DESCRIPTION	8
10.2	VB EXAMPLE CODE	8
10.3	MANUAL WORK-AROUND	8
11	SHORT-CIRCUITED BOOLEAN LOGIC	8
11.1	LIMITATION DESCRIPTION	8
11.2	VB EXAMPLE CODE	8
11.3	MANUAL WORK-AROUND	8
12	EXPRESSION IN TRY/CATCH STATEMENTS	9

12.1	LIMITATION DESCRIPTION	9
12.2	VB EXAMPLE CODE	9
12.3	MANUAL WORK-AROUND	9
13	LATE-BINDING	10
13.1	LIMITATION DESCRIPTION	10
13.2	VB EXAMPLE CODE	10
13.3	MANUAL WORK-AROUND	10
14	USING VB 2005 KEYWORDS IN VB 2003.....	10
14.1	LIMITATION DESCRIPTION	10
14.2	VB EXAMPLE CODE	10
14.3	MANUAL WORK-AROUND	10
15	OPERATOR OVERLOADING (VISUAL STUDIO 2005).	11
15.1	LIMITATION DESCRIPTION	11
15.2	VB EXAMPLE CODE	11
15.3	MANUAL WORK-AROUND	11
16	SMART DEVICE PROJECTS.....	11
16.1	LIMITATION DESCRIPTION	11
16.2	VB EXAMPLE CODE	11
16.3	MANUAL WORK-AROUND	11
17	FTP WEBSITE IN VS 2005.....	11
17.1	LIMITATION DESCRIPTION	11
17.2	VB EXAMPLE CODE	11
17.3	MANUAL WORK-AROUND	11

1 Introduction

This document describes the known limitations and workarounds of C-Sharpener For VB. The translator automatically adds "TRANSERROR:" and "TRANSWARNING:" comments into the translated code. You can search for these comments and resolve any translation issues. You can do most of the workarounds either in the original VB code or in the equivalent C# code after the project is translated.

2 Option Explicit and Option Strict

2.1 Limitation Description

You must compile your VB code with Option Explicit **On**. C# has stricter requirements for type checking and conversions, and requires that all variables be defined. Option Strict can be **On** or **Off**, but you will get better results if it is **On**.

2.2 VB example code

NA

2.3 Manual work-around

- Open the **Project Settings** dialog for the VB.NET project.
- Go to the **Build** page under **Common Properties**.
- Select the option **On** in **Option Explicit** combo box.
- Compile the project.

3 Preprocessor Directives

3.1 Limitation Description

#Const in VB is converted to **#define** in C#, but there is no way to associate a value with the **#define**. We don't support preprocessor directives embedded inside a class or method. They must appear at the top of the file outside the scope of any class. Also, **#ExternalSource** directives are not supported.

3.2 VB example code

```
#Const MyLocation = "USA"  
#Const Version = "8.0.0012"  
#Const CustomerNumber = 36
```

```
Public Class Class1  
  
    Public Sub New ()  
  
    End Sub  
  
End Class
```

3.3 Manual work-around

Move the constants to a class in the original VB.NET project:

```
Public Class Constants  
  
    Const MyLocation = "USA"
```

```
Const Version = "8.0.0012"  
Const CustomerNumber = 36
```

```
End Class
```

4 Compilation Options

4.1 Limitation Description

Compilation Options (**Explicit**, **Strict**, and **Compare**) are ignored. If you use **Option Compare Text** then you need to be aware that all string comparisons are case-insensitive in VB while C# is case-sensitive in C#. **Option Compare Binary** or no **Option Compare** is case-sensitive which means there would be no problem in C#.

4.2 VB example code

NA

4.3 Manual work-around

NA

5 Shadows keyword

5.1 Limitation Description

VB supports hide-by-name semantics by using the **Shadows** keyword. C# only supports hide-by-signature. As a result, the **Shadows** keyword is changed to the **new** keyword in C# even though they have slightly different meanings.

5.2 VB example code

```
Public Class Class1  
    Inherits BaseClass  
  
    Public Sub New ()  
  
    End Sub  
  
    Public Shadows Sub Function1 ()  
  
    End Sub  
  
End Class
```

5.3 Manual work-around

Choose Virtual functions if it is logically correct.

```
Public Class Class1  
    Inherits BaseClass  
  
    Public Sub New ()  
  
    End Sub  
  
    Public Overrides Sub Function1 ()
```

```
End Sub
```

```
End Class
```

6 Embedded enumerations/interfaces

6.1 Limitation Description

According to the VB Grammar Summary, interfaces aren't allowed to have embedded enumerations or interfaces. However, the VB compiler allows this. C# does not support embedded enumerations or interfaces in interfaces.

6.2 VB example code

```
Public Interface ITest1
```

```
    Enum Enum1
        One
        Two
    End Enum
```

```
End Interface
```

6.3 Manual work-around

Move the enumeration outside the interface declaration:

```
Public Interface ITest1
```

```
End Interface
```

```
Enum Enum1
    One
    Two
End Enum
```

7 Unstructured exception handling

7.1 Limitation Description

Unstructured exception handling is not supported. This includes the On Error Resume Next and Resume Next statements. A "TRANSERROR:" comment is inserted for any of these statements.

7.2 VB example code

```
Public Class TestClass
```

```
    Public Sub New ()
```

```
    End Sub
```

```
    Public Sub Function1 ()
```

```
        On Error Resume Next
```

```
        Dim x As Integer = 32
```

```
        Dim y As Integer = 0
```

```
        Dim z As Integer
```

```
        z = x / y ' Creates a divide by zero error
```

```
        Console.WriteLine (z)
    End Sub
End Class
```

7.3 Manual work-around

Use **try catch** block instead:

```
Public Class TestClass
    Public Sub New ()
    End Sub

    Public Sub Function1 ()
        Try
            Dim x As Integer = 32
            Dim y As Integer = 0
            Dim z As Integer

            z = x / y ' Creates a divide by zero error

        Catch ex As DivideByZeroException
        End Try

        Console.WriteLine (z)

    End Sub
End Class
```

8 MyClass keyword

8.1 Limitation Description

VB has a `MyClass` keyword to always call the current implementation even when it is overridden in a derived class. C# has no equivalent capability. All `MyClass` keywords are translated to "this" and are marked with a `TRANSERROR` comment.

8.2 VB example code

```
Public Class TestClass
    Public Sub New ()
    End Sub

    Public Sub Function1 ()
        MyClass.Function2 ()
    End Sub

    Public Sub Function2 ()
```

```
End Sub
```

```
End Class
```

8.3 Manual work-around

NA

9 Named argument

9.1 Limitation Description

VB allows named argument while C# does not. Name arguments are not translated and a "TRANSERROR:" comment is inserted near any named argument.

9.2 VB example code

```
Public Class TestClass

    Public Sub New ()

    End Sub

    Sub Function1 ()

        StudentInfo (Age:=19, Name:="Test")

    End Sub

    Sub StudentInfo (ByVal Name As String, ByVal Age As Short)

    End Sub

End Class
```

9.3 Manual work-around

Pass arguments by position:

```
Public Class TestClass

    Public Sub New ()

    End Sub

    Sub Function1 ()

        StudentInfo ("Test", 19)

    End Sub

    Sub StudentInfo (ByVal Name As String, ByVal Age As Short)

    End Sub

End Class
```

10 Project Files

10.1 Limitation Description

C-Sharpener For VB can only convert files that are listed in your project. If you have files that your project depends on that are not part of the project they aren't copied to the new project directory structure. This is common for image files in a web application for instance.

10.2 VB example code

NA

10.3 Manual work-around

Add the files used by the application to the C# project directory manually or add the missing files to the VB project.

11 Short-Circuited Boolean Logic

11.1 Limitation Description

C# uses short-circuited Boolean logic and VB always evaluates the entire logical expression. If you have Boolean expression that relies on all parts of the expression being executed, you will manually have to fix the code. For instance, if you have a Boolean expression that calls a method and the method has a side effect of doing something besides returning True/False, then you will have to add code to insure that the method is called.

11.2 VB example code

```
Public Class TestClass

    Sub Function1 ()

        If 45 < 12 And Function2(3) = 81 Then

            End If

        End Sub

        Function Function2 (ByVal arg1 As Integer) As Boolean

            End Function

    End Class
```

11.3 Manual work-around

```
Public Class TestClass

    Sub Function1 ()

        Dim function2Result As Integer = Function2(3)

        If 45 < 12 And function2Result = 81 Then

            End If

        End Sub

    End Class
```

```
Function Function2 (ByVal arg1 As Integer) As Boolean
    End Function
End Class
```

12 Expression in Try/Catch statements

12.1 Limitation Description

VB supports When expressions in Try/Catch statements and C# does not. A "TRANSERROR:" comment is written for this situation.

12.2 VB example code

```
Public Class TestClass
    Sub Function1 ()
        Dim x As Double = 5
        Dim y As Double = 0

        Try
            x /= y ' Cause a "Divide by Zero" error.

            Catch ex As Exception When y = 0
                MsgBox (ex.ToString)
            End Try
        End Sub
    End Class
```

12.3 Manual work-around

Try to avoid expressions in Try/Catch statements

```
Public Class TestClass
    Sub Function1 ()
        Dim x As Double = 5
        Dim y As Double = 0

        Try
            x /= y ' Cause a "Divide by Zero" error.

            Catch ex As Exception
                If (y = 0) Then
                    MsgBox (ex.ToString)
                End If
            End Try
        End Sub
    End Class
```

```
End Sub
```

```
End Class
```

13 Late-binding

13.1 Limitation Description

VB supports late binding and C# does not. Translating late-binding is not supported and will generally cause a compiler error.

13.2 VB example code

```
Public Class TestClass

    Sub Function1 ()

        Dim obj As Object = New Checking()

        obj.MakeDeposit(100)

    End Sub

End Class
```

13.3 Manual work-around

```
Public Class TestClass

    Sub Function1 ()

        Dim chk As Checking = New Checking()

        Chk.MakeDeposit(100)

    End Sub

End Class
```

14 Using VB 2005 keywords in VB 2003.

14.1 Limitation Description

C-Sharpener For VB does not support the conversion of Visual Studio 2005 keywords in Visual Studio 2003. C-Sharpener For VB will throw a parsing error on the following keywords: `Continue`, `CSByte`, `CUInt`, `CULng`, `CUShort`, `Custom`, `IsNot`, `Of`, `Operator`, `Partial`, `SByte`, `TryCast`, `UInteger`, `Ulong`, `Ushort`, `Using`, `Widening` and `Narrowing`

14.2 VB example code

```
Dim operator As String
Public Sub continue()

End Sub
```

14.3 Manual work-around

You have to rename these keywords in Visual Studio 2003.

```
Dim operator1 As String
Public Sub continuel()

End Sub
```

15 Operator Overloading (Visual Studio 2005).

15.1 Limitation Description

C-Sharpener For VB does not support some operators in VB because there are no equivalents in C#. The VB.NET operators [IsTrue](#), [IsFalse](#), [Like](#), [^](#) and [Ctype](#) are not supported. Also, VB.NET operator overloading supports the following keywords [Shadows](#), [Widening](#) and [Narrowing](#) are not supported.

15.2 VB example code

NA

15.3 Manual work-around

NA

16 Smart Device projects.

16.1 Limitation Description

C-Sharpener For VB does not support the translation of Smart Device projects.

16.2 VB example code

NA

16.3 Manual work-around

NA

17 FTP website in VS 2005.

17.1 Limitation Description

C-Sharpener For VB does not support the translation of FTP websites in VS 2005.

17.2 VB example code

NA

17.3 Manual work-around

NA