

```
1 Imports System
2 Imports System.Reflection
3 Imports System.Runtime.InteropServices

4 ' General Information about an assembly is controlled through the following
5 ' set of attributes. Change these attribute values to modify the information
6 ' associated with an assembly.

7 ' Review the values of the assembly attributes

8 <Assembly: AssemblyTitle("SampleProject.VB")>
9 <Assembly: AssemblyDescription("SampleProject.VB")>
10 <Assembly: AssemblyCompany("Sub Main(), a Division of vbCity.com, LLC")>
11 <Assembly: AssemblyProduct("SampleProject.VB")>
12 <Assembly: AssemblyCopyright("vbCity.com, LLC")>
13 <Assembly: AssemblyTrademark("vbCity.com, LLC")>
14 <Assembly: CLSCompliant(True)>

15 'The following GUID is for the ID of the typelib if this project is exposed to COM
16 <Assembly: Guid("E138E340-F455-4D4E-A166-5115BEA4F13F")>

17 ' Version information for an assembly consists of the following four values:
18 '
19 '     Major Version
20 '     Minor Version
21 '     Build Number
22 '     Revision
23 '
24 ' You can specify all the values or you can default the Build and Revision
25 ' by using the '*' as shown below:
26 <Assembly: AssemblyVersion("1.0.*")>
```

```
1 Option Explicit On
2 Option Strict On

3 '=====
4 ' File:      FindType.vb
5 ' Summary:  Command Line utility to help find and display information
6 '           on .NET types.
7 ' Classes:  FindType
8 '           Indented Writer
9 ' Origin:   .NET Samples Team
10 '=====

11 Imports System
12 Imports System.Reflection
13 Imports System.IO

14 Namespace SubMain.Sample
15     Public Class App
16         Private SHOW_INTERFACES As Integer = &H1
17         Private myWriter As New IndentedWriter
18         Private exactMatchOnlyField As Boolean = False
19         Private DirList As New ArrayList

20         ' Do an exact match of passed types - the name passed must
21         ' be the same as the fully qualified type name.
22         Public Property ExactMatchOnly() As Boolean
23             Get
24                 Return exactMatchOnlyField
25             End Get
26             Set(ByVal Value As Boolean)
27                 exactMatchOnlyField = Value
28             End Set
29         End Property

30     Public Shared Sub Main(ByVal args() As String)
31         Dim ft As New FindType
32         Try
33             SetOptions(args, ft)
34             ft.Search()
35         Catch ex As SecurityException
36             Console.WriteLine "This sample has failed to run due to a security
37                 limitation!\n" &
38                 "Try running the sample from a local drive or using CasPol.exe to turn
39                 off security.";
40         End Try
41     End Sub 'Main

42     ' Processes all of the options specified in the arguments
43     Private Shared Sub SetOptions(ByVal args() As String, ByVal ft As FindType)
44         Try
45             Dim backslash As String
46             backslash = "\"

47             Dim i As Integer
48             For i = 0 To args.Length - 1
```

```
47     Dim curArg As String = args(i)
48     If curArg.ToCharArray()(0) = "-"c Or curArg.ToCharArray()(0) = "/"c
49     Then
50         If Char.ToUpper(curArg.ToCharArray()(1)) = "D"c Then
51             If curArg.ToCharArray().Length > 2 And curArg.ToCharArray()(2) =
52             ":"c Then
53                 Dim dir As String = curArg.Substring(3).TrimEnd(backslash.
54                 ToCharArray()).ToUpper()
55
56                 If dir <> "" Then
57                     ft.DirAdd(dir)
58                 Else
59                     Console.WriteLine("Directory not specified")
60                 End If
61             Else
62                 Console.WriteLine("Directory not specified")
63             End If
64         Else
65             Dim j As Integer
66             For j = 1 To curArg.ToCharArray().Length - 1
67                 Select Case Char.ToUpper(curArg.ToCharArray()(j))
68                 Case "A"c
69                     ft.ShowAll()
70                 Case "D"c
71                     Console.WriteLine("Directory not specified")
72                 Case Else
73                     Console.WriteLine("Invalid Option[{0}]", curArg.ToCharArray(
74                     (j))
75                 End Select
76             Next j
77         End If
78     Else
79         ft.ClassAdd(curArg)
80     End If
81     Next i
82 Catch rcle As ReflectionTypeLoadException
83     ' The following line would output the TypeLoadException.
84     ' myWriter.WriteLine("Unable to load a type because {0}",
85     exceptions(exceptionCount))
86     exceptionCount += 1
87 Catch fnfe As FileNotFoundException
88     myVerboseWriter.WriteLine(fnfe.Message)
89 Catch
90 End Try
91 End Sub 'SetOptions
92
93 ' Receive chat from remote client
94 Private Sub ReceiveTalk()
95     Dim readMode As Integer = 1
96
97     While readMode <> 0
98         If reader.Read(oneBuffer, 0, 1) = 0 Then
99             readMode = 0
100        Else
101            Select Case readMode
```

```
94     Case 1
95         { If counter = commandBuffer.Length Then
96             readMode = 0
97         End If
98         { If oneBuffer(0) <> ":"c Then
99             commandBuffer(counter) = oneBuffer(0)
100         Else
101             counter = Convert.ToInt32(New String(commandBuffer, 1, counter +
102                 1))
103             { If counter > 0 Then
104                 textObj.Length = 0
105             Else
106                 { If commandBuffer(0) = "R"c Then
107                     prevReceiveText = String.Empty
108                     RaiseEvent Notifications(Notification.ReceivedRefresh,
109                         prevReceiveText)
110                 End If
111             End If
112         End If
113     Case 2
114         counter = counter - 1
115         { If counter = 0 Then
116             { Select Case commandBuffer(0)
117                 Case "R"c
118                     prevReceiveText = textObj.ToString()
119                     RaiseEvent Notifications(Notification.ReceivedRefresh,
120                         prevReceiveText)
121                 Case Else
122                     Dim newText As String
123                     newText = textObj.ToString()
124                     RaiseEvent Notifications(Notification.ReceivedAppend,
125                         newText)
126             End Select
127             readMode = 1
128         End If
129     Case Else
130         readMode = 0
131     End Select
132 End While
133 { For Each f In Dir.GetFiles(Path.GetFileName(switchValue))
134     pathnames.Add(f.FullName)
135 Next f
136 End Sub 'ReceiveTalk
137 End Class 'IndentedWriter
138 End Namespace
```

Index

AssemblyInfo.vb**sample.vb**

Namespace SubMain.Sample	2
Class App	2
Field SHOW_INTERFACES	2
Field myWriter	2
Field exactMatchOnlyField	2
Field DirList	2
Property ExactMatchOnly	2
Method Main	2
Method SetOptions	2
Method ReceiveTalk	3

Printed with
PrettyCode.Print for .NET
<http://www.prettycode.com>