

**Module 5:
Interoperating with
Managed Objects**

Overview

- Using .NET and COM Components in a Windows Forms Application
- Calling Win32 APIs from Windows Forms Applications
- Upgrading Visual Basic 6.0 Applications to Visual Basic .NET

Lesson: Using .NET and COM Components in a Windows Forms Application

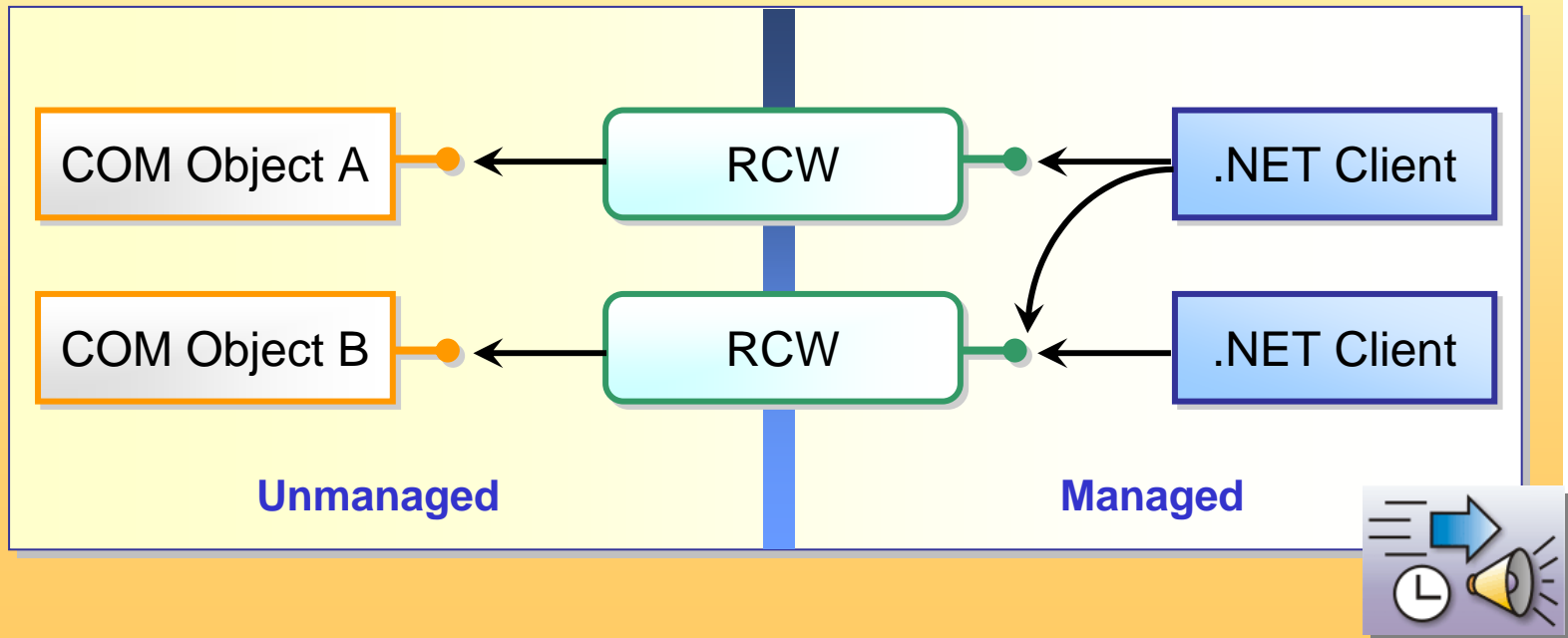
- COM vs .NET
- How to Call COM Components from .NET
- Role of the RCW
- How to Generate Interop Assemblies
- Private, Shared, and Primary Interop Assemblies
- Practice: Using COM Components in .NET-Based Applications

COM vs .NET

COM Model	.NET Framework Model
Interface based	Object based
GUIDs	Strong names
Binary standard	Type standard
Type library	Metadata
HResults	Exceptions
Reference counting	Garbage collection
Immutable	Resilient

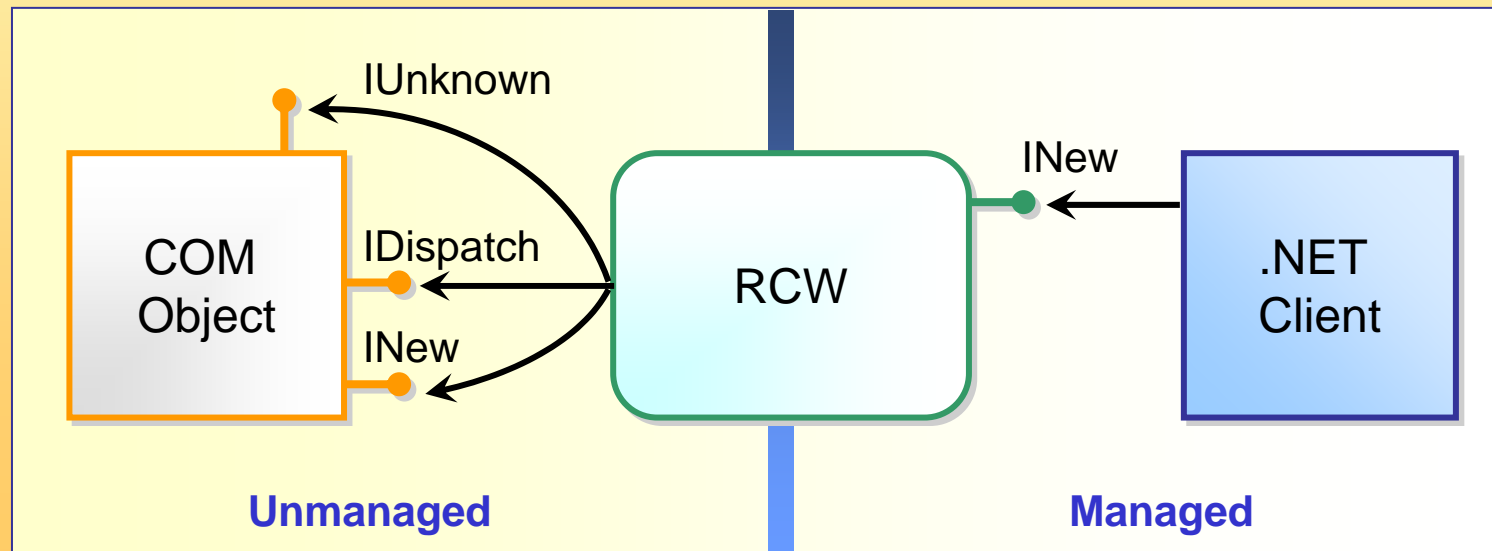
How to Call COM Components from .NET

- The CLR enables managed code to call COM objects through a proxy called the Runtime Callable Wrapper (RCW)
- The RCW is a managed object and subject to garbage collection

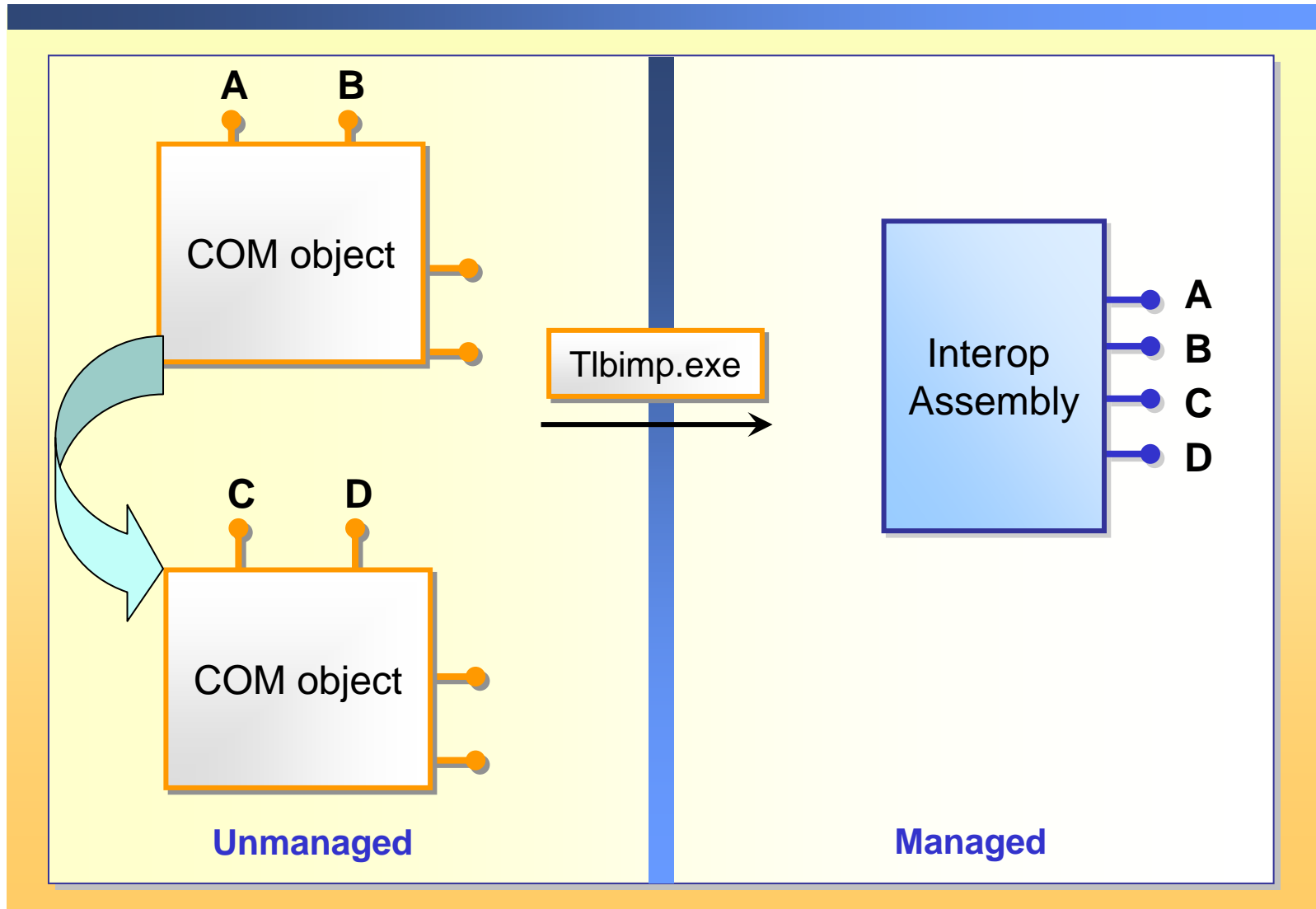


Role of the RCW

- Maintains object lifetime
- Marshals method calls between managed and unmanaged code
- Consumes selected COM interfaces without exposing them to the .NET client

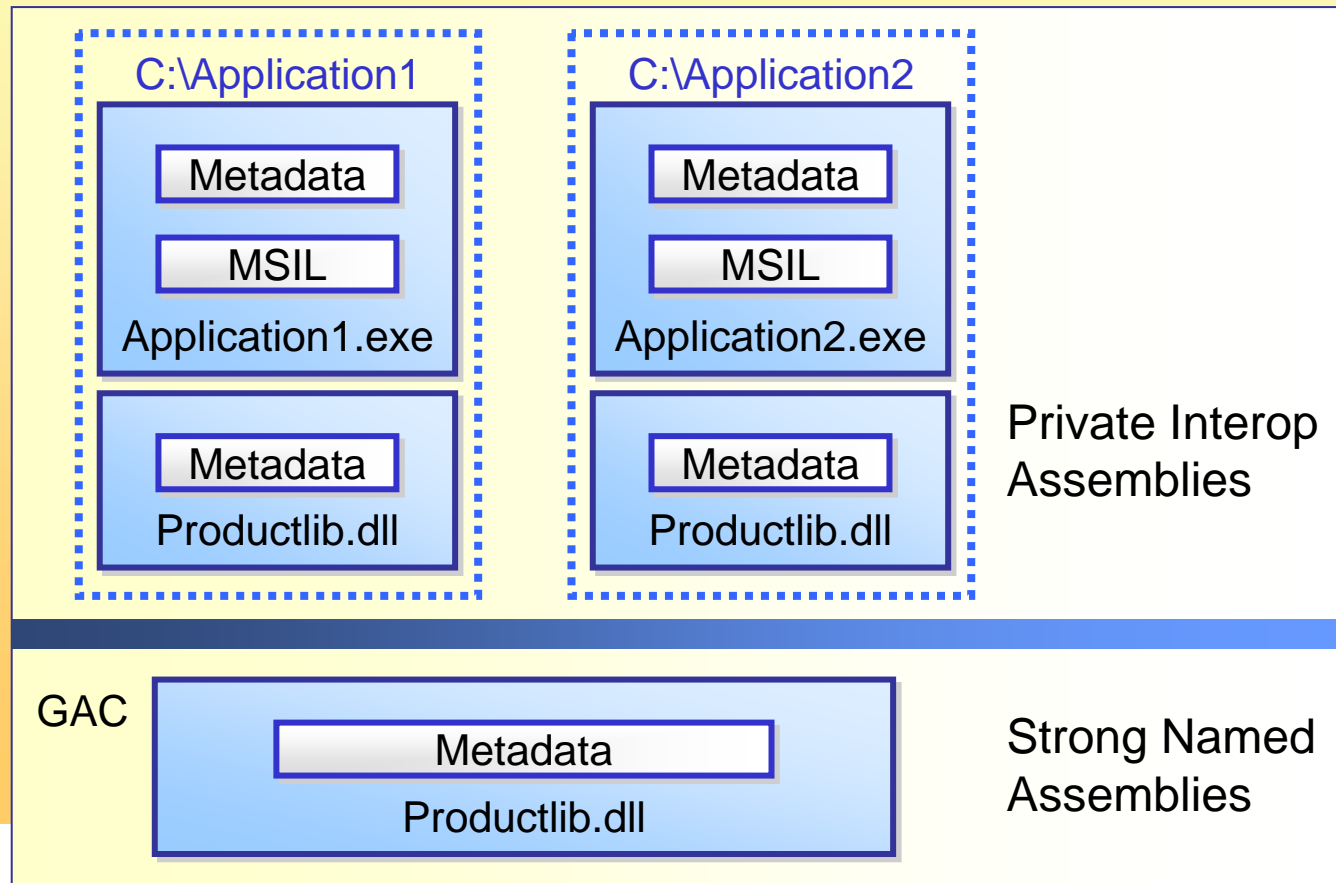


How to Generate Interop Assemblies



Private, Shared, and Primary Interop Assemblies

Primary Interop Assembly is an assembly signed by the company that produced the original type library

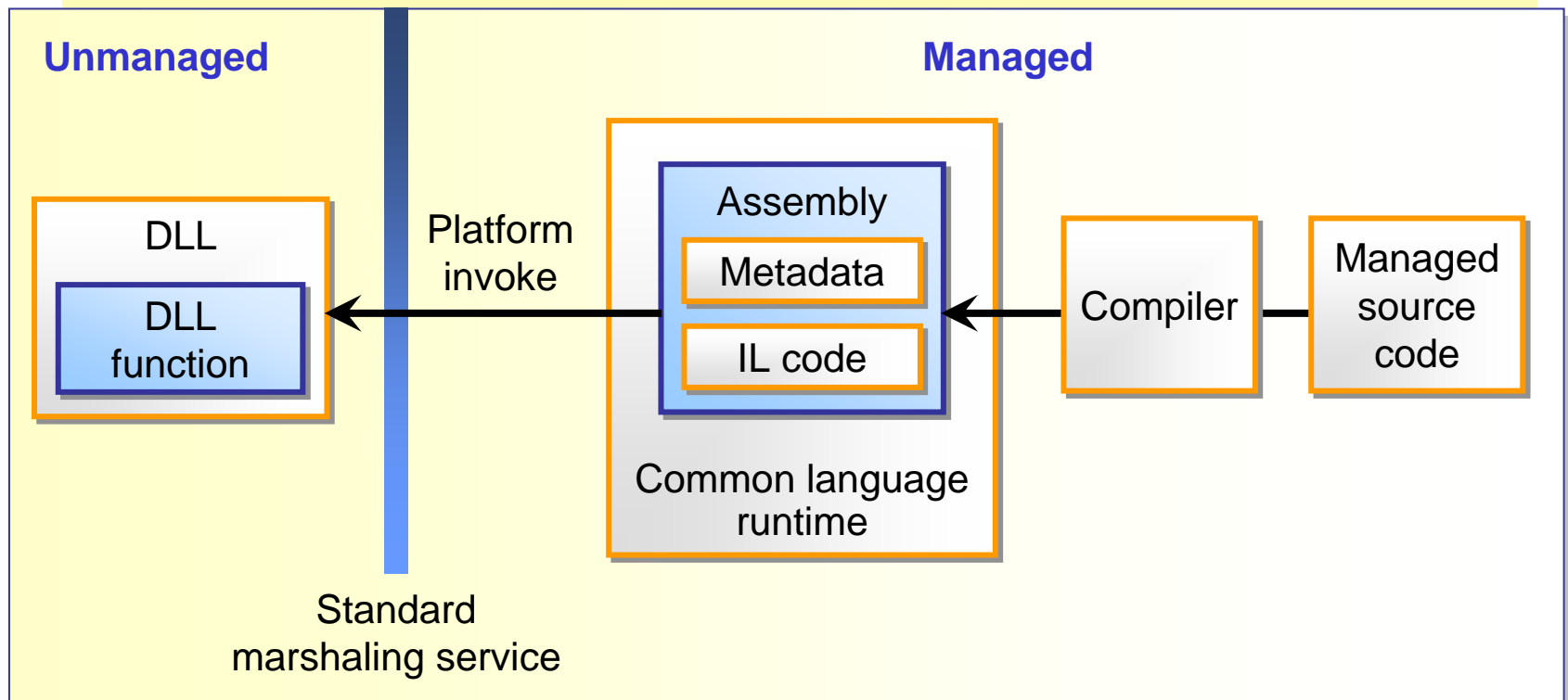


Lesson: Calling Win32 APIs from Windows Forms Applications

- Platform Invocation Services
- How to Define Functions by Using the Declare Statement
- How to Define Functions by Using the DllImport Attribute
- How to Call Win32 APIs from a Windows Forms Application
- Practice: Calling Win32 APIs

Platform Invocation Services

The PInvoke Model



A platform invoke call to an unmanaged DLL function

How to Define Functions by Using the Declare Statement

- The Declare keyword can be used to create function definitions in Visual Basic .NET
- Can also use DllImport for more flexibility

```
Public Class Win32PlaySound
Public Declare Auto Function PlaySound Lib
    "winmm.dll" Alias "PlaySound" _
    (ByVal pszSound As String, ByVal hmod As Long,
    ByVal fdwSound As Long) As Boolean
End Class
```

How to Define Functions by Using the DllImport Attribute

- DllImport attribute is used to define functions
- Parameters are used to specify specific behavior
 - CallingConvention
 - EntryPoint
 - CharSet
 - ExactSpelling

```
Imports System.Runtime.InteropServices
```

```
<DllImport("KERNEL32.DLL",  
EntryPoint:="MoveFileW", SetLastError:=True, _  
CharSet:=CharSet.Unicode, ExactSpelling:=True, _  
CallingConvention:=CallingConvention.StdCall)> _
```

```
Public Shared Function _
```

```
MoveFile(ByVal src As String, ByVal dst As String)  
As Boolean
```

```
End Function
```

How to Call Win32 APIs from a Windows Forms Application

Determine the details of the function you want to call

Create a new project in Visual Studio .NET

Import the **System.Runtime.InteropServices** namespace

```
Imports System.Runtime.InteropServices
```

Define a function using **Declare** or **DllImport**

Add code to call the Win32 API from a Windows Form

```
Private myWin32Object As New Win32PlaySound ()  
myWin32Object.PlaySound("C:\WINDOWS\Media\tada.wav", 0)
```

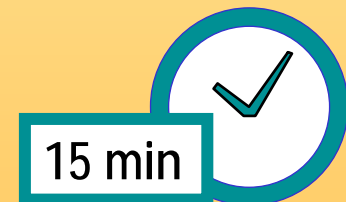
Practice: Calling Win32 APIs



In this practice, you will

- Call a Win32 API from a .NET-based application using the **Declare** statement
- Call a Win32 API from a .NET-based application using the **DllImport** attribute

Begin reviewing the objectives
for this practice activity



Lesson: Upgrading Visual Basic 6.0 Applications to Visual Basic .NET

- Advantages of Upgrading
- Results of the Upgrade Wizard
- Demonstration: Upgrading Visual Basic 6.0 Applications to Visual Basic .NET

Advantages of Upgrading

Advantages

Deployment

Language integration

Object-oriented language

Improved RAD experience

No DLL versioning issues

Results of the Upgrade Wizard

■ Language Changes

- Code upgraded to be syntactically correct in Visual Basic .NET

■ Form Changes

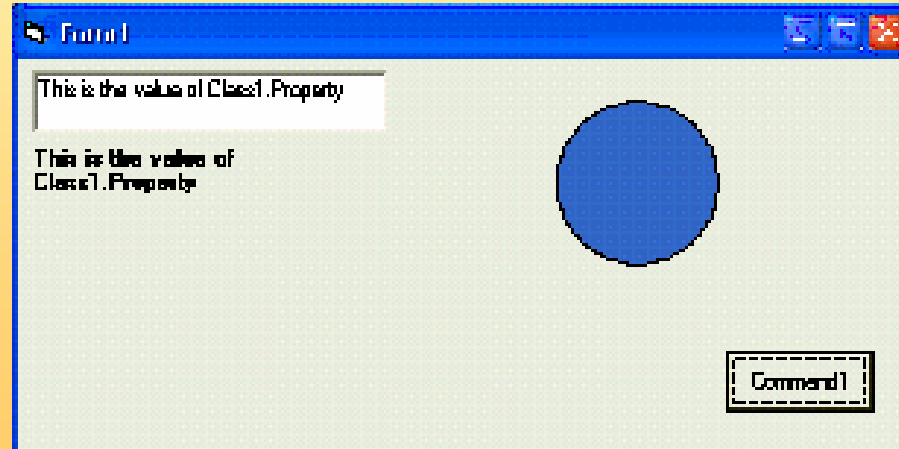
- Visual Basic Forms upgrade to Windows Forms
- Most controls will upgrade
- Controls that have no counterparts are upgraded as label controls

■ Other Changes

- Other functionality are upgraded to similar objects
- ADO data environments are not upgraded

Demonstration: Upgrading Visual Basic 6.0 Applications to Visual Basic .NET

In this demonstration, you will see how to upgrade a Visual Basic 6.0 application to Visual Basic .NET using the Upgrade Wizard



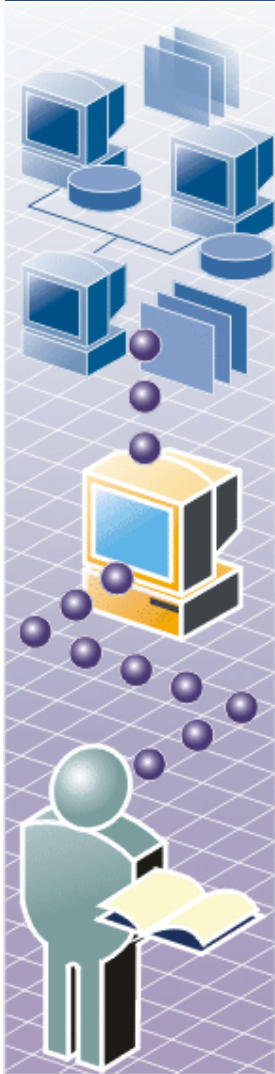
Visual Basic 6.0 Application



Review

- Using .NET and COM Components in a Windows Forms Application
- Calling Win32 APIs from Windows Forms Applications
- Upgrading Visual Basic 6.0 Applications to Visual Basic .NET

Lab 5.1: Interoperating with COM and Calling Win32 APIs



- Exercise 1: Using a COM Component in a .NET-Based Application
- Exercise 2: Calling Win32 APIs from a .NET-Based Application