Lab 12: Application Setup and Optimization

For background information on this lab, click each of these topics:

Objectives

By the end of this lab, you will be able to:

- Create a resource file for your application.
- Save application settings in the registry.
- Use the Setup Wizard to create a Setup program for your application.
- Use the Visual Basic Code Profiler.

Prerequisites

Before starting this lab, you should be familiar with the following concepts:

- The purpose and structure of a resource file.
- The purpose and structure of the Windows registry
- The purpose and structure of a Setup program

Lab Setup

To complete this lab, you need the following:

- Visual Basic 5.0 or later

To see a demonstration of the completed lab solution, click this icon.

Estimated time to complete this lab: **45 minutes**

Note There are project and solution files associated with each lab. If you installed the labs during Setup, these files are in the folder <*Install Folder*>\Labs on your hard disk. If you did not install the labs during Setup, you can find them in the \Labs folder of the *Mastering Microsoft Visual Basic 5* CD-ROM.

Exercises

The following exercises provide practice working with the concepts and techniques covered in Chapter 12.

Exercise 1: Using Resource Files

In this exercise, you will use a resource file within your application.

Exercise 2: Using the Registry

In this exercise, you will store application-specific information in the Windows registry.

Exercise 3: Using the Setup Wizard

In this exercise, you will use the Setup Wizard to build a Setup program that will install the application you created in this lab.

Exercise 4: (Optional) Using Visual Basic Code Profiler

In this exercise, you will use the Visual Basic Code Profiler to examine the performance of a Visual Basic-based application.

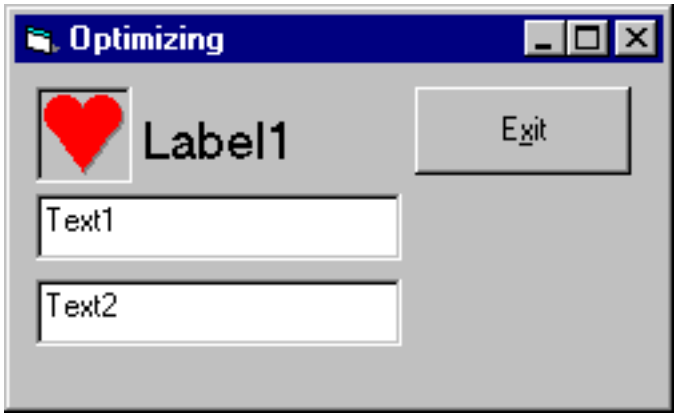
Note Make sure that you copy the folder \Tools\Unsupprt\Vbcp from your Visual Basic 5.0 CD-ROM to the Visual Basic folder on your hard drive.

Exercise 1: Using Resource Files

In this exercise, you will use a resource file within your application. For more information about resource files, see Using Resource Files.

▶ **Create a form**

1. Create a new project.
2. Add controls to the form, as shown in the following illustration.



   The control in the upper-left corner is a picture control. Place any icon you want in the **Picture** property. Icons are provided in the Visual Basic \Icons folder.
3. Save the application.

▶ **Use resources**

1. In the folder <*Install Folder*>\Labs\Lab12, add the resource file Lab.res to the project.
2. In the Form_Load event, load resource picture 100, and assign it to the **Picture** property of the **Image** control.
3. Load resource string 100, and assign it to the **Caption** property of the **Label** control.

```
Private Sub Form_Load()
    Label1.Caption = LoadResString(100)
    Image1.Picture = LoadResPicture(100, 1)
End Sub
```

4. Test the application. If the resources are loaded correctly, your form should resemble the following illustration.

5. To load the picture and the string resources numbered 200, change the code in the Form_Load event appropriately.
6. Test the application. How has it changed?

Exercise 2: Using the Registry

In this exercise, you will store application-specific information in the Windows registry.

▶ **Save information in the Windows registry**

1. In the form **Unload** event procedure, use the **SaveSetting** statement to save the information in the first text box to the Windows registry. Use the arguments listed in the following table.

Argument	Value
AppName	Polishing
Section	General
Key	Text1
Setting	Text1.Text

To see an example of how your code should look, click this icon.

```
Private Sub Form_Unload(Cancel As Integer)
    SaveSetting AppName:="Polishing", Section:="General", _
        Key:="Text1", Setting:=Text1.Text
End Sub
```

2. In the form **Load** event procedure, use the **GetSetting** function to read the information from the Windows registry into the first text box. Use the arguments listed in the following table.

Argument	Value
AppName	Polishing
Section	General
Key	Text1
Setting	Default

To see an example of how your code should look, click this icon.

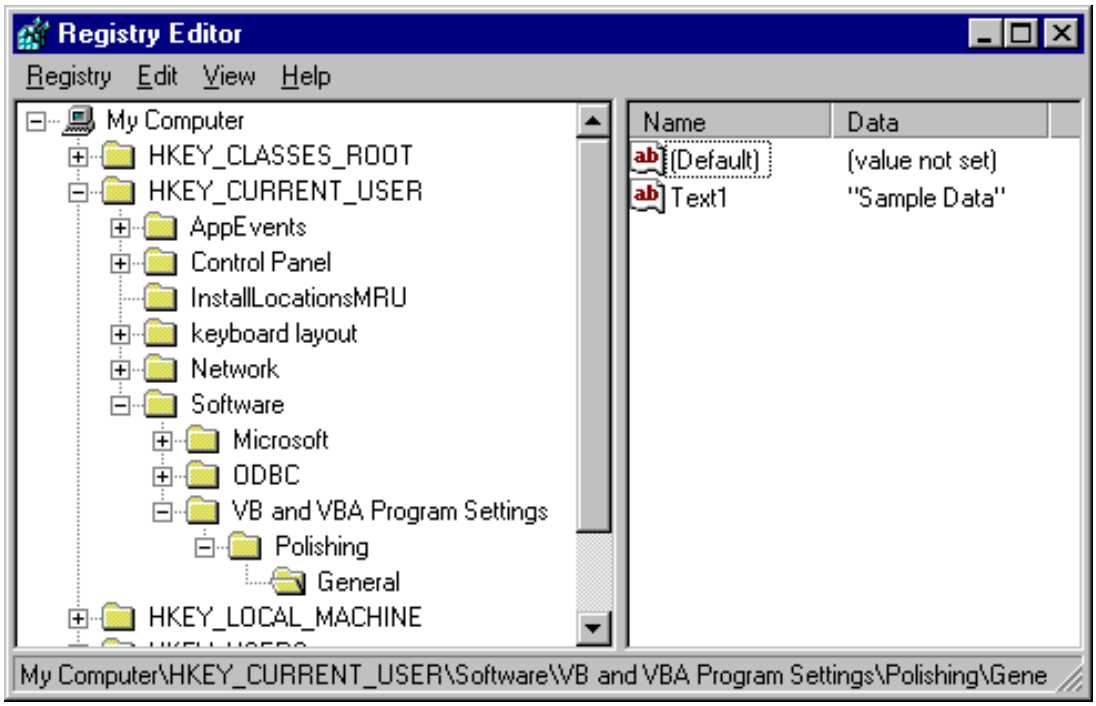
```
Private Sub Form_Load()
    Label1.Caption = LoadResString(100)
    Image1.Picture = LoadResPicture(100, 1)
```

```
    ' get registry settings
    Text1.Text = GetSetting(AppName:="Polishing", Section:="General",
_
        Key:="Text1", Default:="Default")
End Sub
```

3. Test the application. Is the information saved and loaded correctly?

   You can also examine the Windows registry by running Regedit.exe.

   If you run Regedit after completing this exercise, you can find the application-specific information in the registry, as shown in the following illustration.



Exercise 3: Using the Setup Wizard

In this exercise, you will use the Setup Wizard to build a Setup program that installs the application you created in this lab.

▶ Build a Setup program

1. Run the Setup Wizard.
2. In the **Select Project and Options** dialog box, choose to create a Setup program without a dependency file, and to have the executable rebuilt.
3. For the distribution method, select **Single Directory**.
4. For the destination folder, choose <*Install Folder*>\Labs\Lab12.
5. Do not add any ActiveX servers to the Setup.
6. In the **File Summary** dialog box, add the Readme.txt file located in the folder <*Install Folder*>\Labs\Lab12, and review the file summary information.
7. In the **Finished** dialog box, click the **Finish** button to build the Setup program.

   For more information about the Setup Wizard, see Using the Setup Wizard.

▶ Test Setup

1. Run the Setup program you just created, and install the application in the default folder.

2. On the **Start** menu, click **Programs**.

   The application should be listed.

3. Run the application.
4. View the contents of the application folder. Has the file Readme.txt been installed correctly?
5. On the **Start** menu, point to **Settings**, and then click **Control Panel**.
6. Inn the **Control Panel** dialog box, double-click **Add/Remove Programs**.
7. On the **Install/Uninstall** tab, remove the application you installed.

Exercise 4: (Optional) Using Visual Basic Code Profiler

In this exercise, you will use the Visual Basic Code Profiler to examine the performance of a Visual Basic application.

Note Make sure that you copy the folder \Tools\Unsupprt\Vbcp from your Visual Basic CD-ROM to the Visual Basic folder on your hard drive.

▶ **Install the Code Profiler add-in**

1. Register the DLL for the Code Profiler (Vbcp.dll) by using the file Regsvr32.exe.
2. Edit the file VBAddin.ini located in the Windows folder to include the entry **VBcp.VBcpClass=0** under the section **[Add-Ins32]**.
3. In the **Add-In Manager** dialog box, select the **VB Code Profiler** to add it to the Visual Basic design environment.

▶ **Use the Code Profiler**

1. Save the project before using the Code Profiler.
2. To profile code, click the **VB Code Profiler** on the **Add-Ins** menu.
3. To add code to the project, click **Add Profiler Code** in the **Code Profiler** dialog box.
4. Close the **Code Profiler** dialog box, and run the application.
5. Execute various functions, and end the application.
6. To view the results, click **VB Code Profiler** on the **Add-Ins** menu.
7. On the **File** menu in the **Code Profiler** dialog box, click **View Results**.

   Line-timing analysis is displayed.