



MVSP

Lab Guide

Developed by
D3 MVU Team



Lab Overview

Abstract

The purpose of this lab is to demonstrate the use of MVSP with Enterprise licensing. Using Enterprise licensing allows you to connect several applications to the same D3 server from the same PC while only consuming a single user license. This lab will guide you through how to create a VB.NET application that uses Enterprise licensing.

About the Lab Environment

The lab environment uses the following:

D3 Windows 9.2 and Visual Studio Professional 2013

Lab Overview

- Time estimate: 30 minutes
- There are four sections to this lab:
 - Section 1: Creating the solution
 - Section 2: Creating the form
 - Section 3: Creating the code
 - Section 4: Testing the code

Exercise 1: Creating the solution

Purpose of the Exercise

This exercise will show you how to create a Visual Studio solution with a reference to MVSP.

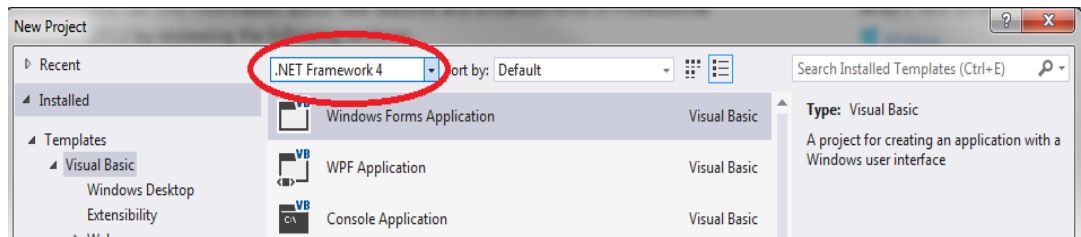
After this exercise you will be able to:

- Create a new Visual Studio solution
- Create a reference to MVSP

Exercise Instructions

Perform the following steps:

- ___ 1. Creating the Visual Studio solution.
 - a. From the Windows Start menu, select Visual Studio 2013 > Visual Studio 2013.
 - b. Click New Project.
 - c. Select Templates > Visual Basic and Windows Form Application.
 - d. Ensure .NET Framework 4 is selected in the dropdown list as shown below.



- e. Type a name for this project.
- f. Right-click on the project name and then select Add > Reference .
- g. Click Browse.
- h. Select the RocketMVSP.dll in the C:\MVU\MVSP\MVSPNetAPI\net4.0 directory.
- i. Click OK and then click OK again to add the reference.

Exercise 1 summary: Creating a Visual Studio solution and adding a reference to MVSP.

End of Exercise 1

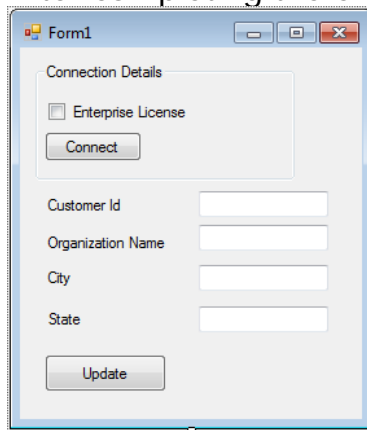
Exercise 2: Creating the form

Purpose of the Exercise

This exercise will show you how to create a form and add some components to it.

After this exercise you will be able to:

- Create a form and populate it with some components
- After completing the exercise your form should look like this

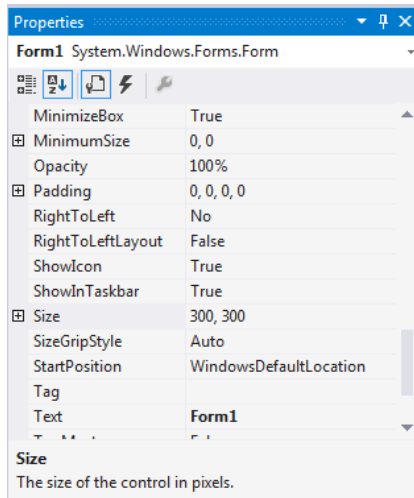


Exercise Instructions

Perform the following steps:

- ___ 1. Resizing the form.
 - a. Click inside the form.

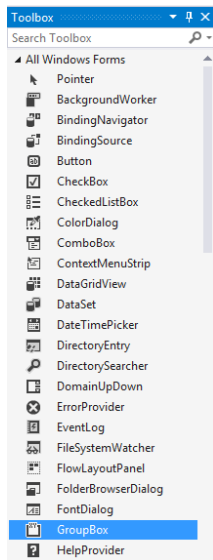
- b. In the Properties window, set the Size property to 300,325.



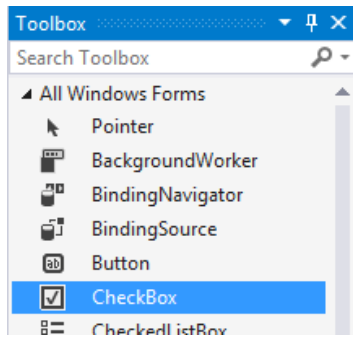
The form should resize to the new size.

___ 2. Creating the Connection Detail components.

- a. Add a GroupBox to the form by selecting the GroupBox control from the Toolbox window and dragging it to the open form window.

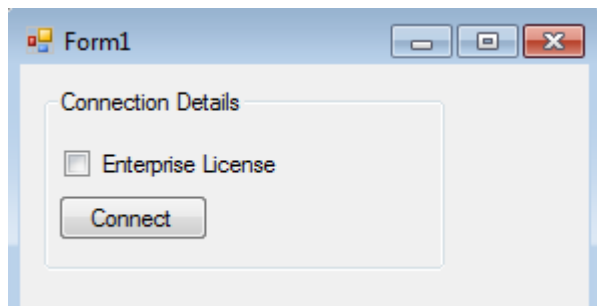


- b. In the Properties window, enter **Connection Details** for the Text property and change the Size property to **200,91**.
- c. Drag-and-drop a CheckBox control from the Toolbox window on to the GroupBox control of the form.



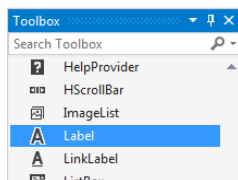
- d. In the Properties window, set the Text property for the new CheckBox control to **Enterprise License** and set the Name property to **cbEnterprise**.
- e. Drag-and-drop a Button control from the Toolbox to the form, placing it below the CheckBox control.
- f. In the Properties window, set the Text property for the new Button control to **Connect** and set the Name property to **btnConnect**.

Your form should now look like this.



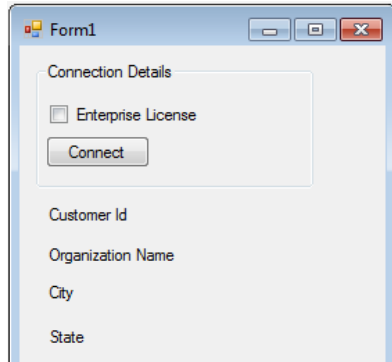
___ 3. Adding data related components to the form.

- a. Drag-and-drop a Label control from the Toolbox to the form, placing it below the group box.

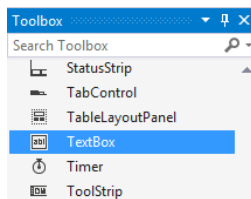


- b. In the Properties window, set the Text property for the new Label control to **Customer Id**.
- c. Drag-and-drop another Label control from the Toolbox to the form, placing it below the **Customer Id** label. Set the Text property for the new Label control to **Organization Name**.

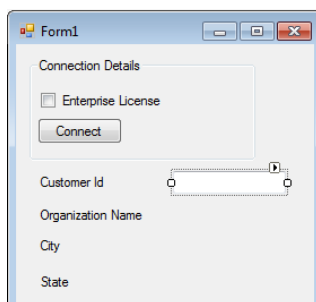
- d. Drag-and-drop another Label control from the Toolbox to the form, placing it below the **Organization Name** label. Set the Text property for the new Label control to **City**.
- e. Drag-and-drop another Label control from the Toolbox to the form, placing it below the **City** label. Set the Text property for the new Label control to **State**. Your form should now look like this.



- f. Drag-and-drop a TextBox Control from the Toolbox to the form, placing it to the right of the **Customer Id** label and beyond the end of the **Organization Name** label.



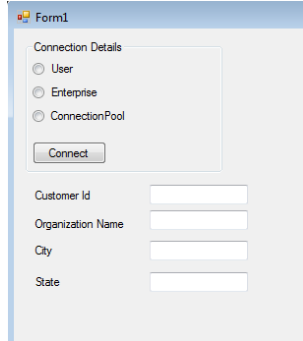
Your form should now look like this.



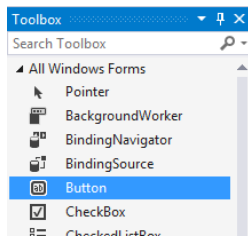
- g. Drag-and-drop another TextBox Control from the Toolbox to the form, placing it to the right of the **Organization Name** label.
- h. Drag-and-drop another TextBox Control from the Toolbox to the form, placing it to the right of the **City** label.

- i. Drag-and-drop another TextBox Control from the Toolbox to the form, placing it to the right of the **State** label.

Your form should now look like this

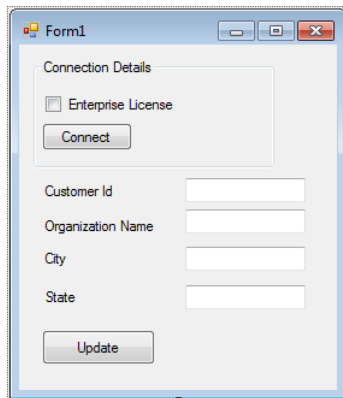


- j. Drag-and-drop a Button Control from the Toolbox to the form, placing it below the **State** label.



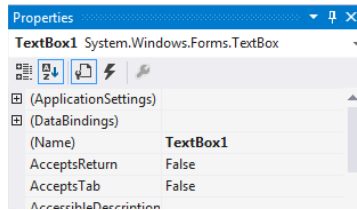
- k. Select the new button and, from the Properties window, set the Name property to **btnUpdate** and the Text property to **Update**.

Your final form should look like this.



___ 4. Name the Text Boxes.

- a. Select the **Customer Id** TextBox control and, from the Properties window, set the Name property to **txtCustId**.



- b. Select the **Organization Name** TextBox control and, from the Properties window, set the Name property to **txtOrgName**.
- c. Select the **City** TextBox control and, from the Properties window, set the Name property to **txtCity**.
- d. Select the **State** TextBox control and, from the Properties window, set the Name property to **txtState**.

Exercise 2 summary: Creating a form to connect to D3 and ready for data display/update.

End of Exercise 2

Exercise 3: Adding code to connect to D3

Purpose of the Exercise

This exercise demonstrates how to connect to D3.

After this exercise you will be able to:

- Construct code to connect to D3 using MVSP

Exercise Instructions

Perform the following steps:

- ___ 1. Opening the code window.
 - a. Double-click the title bar of your form.

```

(Form1 Events) Load
Public Class Form1
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    End Sub
End Class
    
```

You should see the above code displayed.

___ 2. Creating a MVSP object.

- a. Click on the line above the **Private Sub Form1_Load**.
- b. Enter the following code:

```
Dim MVSP As New rocketsoftware.MVSP.Pick
```

```

Form1 (Declarations)
Public Class Form1
    Dim MVSP As New rocketsoftware.MVSP.Pick
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    End Sub
End Class
    
```

___ 3. Adding code to the Connect button.

- a. Click the down arrow (circled below) and select **btnConnect** from the list.

```

Form1 (Declarations)
Public Class Form1
    Dim MVSP As New rocketsoftware.MVSP.Pick
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    End Sub
End Class
    
```

- b. Click the down arrow (this time, to the right of Declarations as illustrated below) and select **Click** from the list.

```

(Declarations)
    
```

- c. Add the following code for the **btnConnect Click** event:

```

Dim rc As Boolean = False ' return code variable

rc = MVSP.Connect("localhost", 9000, "DM", "", False,
cbEnterprise.Checked)
If rc = True Then ' Connected
    rc = MVSP.Logto("MVDEMO", "") ' logto the MVDEMO account
    
```

```

    If rc = False Then ' Logto failed
        MsgBox("Logto failed: " & MVSP.statusMessage)
        Exit Sub ' Exit out of connect
    End If
Else
    MsgBox("Connect to D3 failed: " & MVSP.statusMessage)
    Exit Sub ' Exit out of connect
End If
btnConnect.Enabled = False \ Disable after connecting

```

- d. Your code should now look like this:

```

Private Sub btnConnect_Click(sender As Object, e As EventArgs) Handles btnConnect.Click
    Dim rc As Boolean = False ' return code variable

    rc = MVSP.Connect("localhost", 9000, "DM", "", False, cbEnterprise.Checked)
    If rc = True Then ' Connected
        rc = MVSP.Logto("MVDEMO", "") ' logto the MVDEMO account
        If rc = False Then ' Logto failed
            MsgBox("Logto failed: " & MVSP.statusMessage)
            Exit Sub ' Exit out of connect
        End If
    Else
        MsgBox("Connect to D3 failed: " & MVSP.statusMessage)
        Exit Sub ' Exit out of connect
    End If
    btnConnect.Enabled = False ' Disable after connecting
End Sub

```

___ 4. Adding code to read the item.

- a. Add the following lines below the `Dim` statement shown below

```

Public Class Form1
    Dim MVSP As New rocketsoftware.MVSP.Pick

```

```

    Dim itm As String = "" \ String to hold item
    Dim mfunctions as New rocketsoftware.MVSP.Functions

```

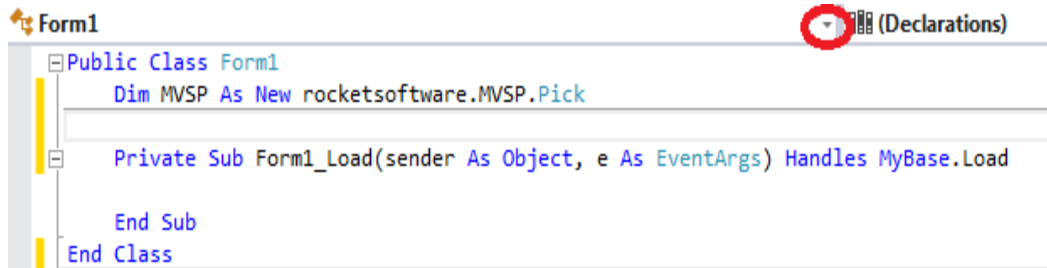
Your code should now look like this:

```

Public Class Form1
    Dim MVSP As New rocketsoftware.MVSP.Pick
    Dim itm As String = "" ' String to hold item
    Dim mfunctions As New rocketsoftware.MVSP.Functions

```

- b. Select the down arrow (circled below) and select **txtCustId** from the list.



```

Form1
Public Class Form1
    Dim MVSP As New rocketsoftware.MVSP.Pick

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load

    End Sub
End Class
  
```

- c. Select the down arrow (to the right of Declarations as illustrated below) and select **KeyUp** from the list.



- d. Enter the following code:

```

If e.KeyCode = Keys.Enter Then
    If MVSP.dbConnected = False Then
        MsgBox("Must connect to D3 first")
        Exit Sub
    End If
    itm = MVSP.FileReadu("CUSTOMERS", txtCustId.Text)
    If MVSP.statusCode = 201 Then
        MsgBox("Unable to open CUSTOMERS")
        btnUpdate.Enabled = False ' Disable update button
        Exit Sub
    End If
    If MVSP.statusCode = 202 Then
        MsgBox("Item does not exist")
        btnUpdate.Enabled = False ' Disable update button
        Exit Sub
    End If
    If MVSP.statusCode = -1 Then
        MsgBox("Item is locked")
        btnUpdate.Enabled = False ' Disable update button
        Exit Sub
    End If
    btnUpdate.Enabled = True ' Enable update button
    txtOrgName.Text = mfunctions.Extract(itm, 1)
    txtCity.Text = mfunctions.Extract(itm, 3)
    txtState.Text = mfunctions.Extract(itm, 4)
End If
  
```

- e. The completed code should look like this:

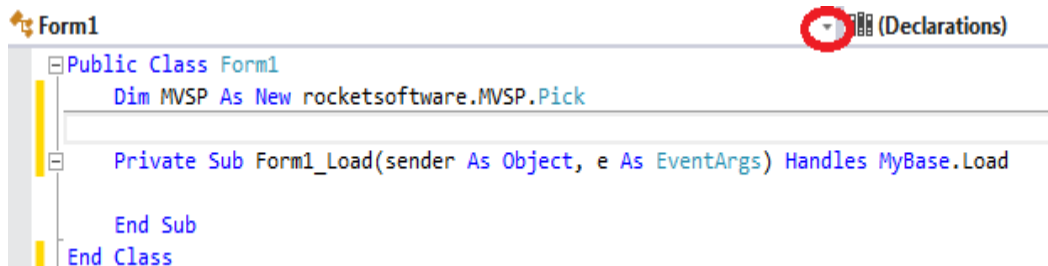
```

Private Sub txtCustId_KeyUp(sender As Object, e As KeyEventArgs) Handles txtCustId.KeyUp
    If e.KeyCode = Keys.Enter Then
        If MVSP.dbConnected = False Then
            MsgBox("Must connect to D3 first")
            Exit Sub
        End If
        itm = MVSP.FileRead("CUSTOMERS", txtCustId.Text)
        If MVSP.statusCode = 201 Then
            MsgBox("Unable to open CUSTOMERS")
            btnUpdate.Enabled = False ' Disable update button
            Exit Sub
        End If
        If MVSP.statusCode = 202 Then
            MsgBox("Item does not exist")
            btnUpdate.Enabled = False ' Disable update button
            Exit Sub
        End If
        If MVSP.statusCode = -1 Then
            MsgBox("Item is locked")
            btnUpdate.Enabled = False ' Disable update button
            Exit Sub
        End If
        btnUpdate.Enabled = True ' Enable update button
        txtOrgName.Text = mfunctions.Extract(itm, 1)
        txtCity.Text = mfunctions.Extract(itm, 3)
        txtState.Text = mfunctions.Extract(itm, 4)
    End If
End Sub

```

___ 5. Adding code to update the item.

- a. Select the down arrow (circled below) and select **btnUpdate** from the list.



- b. Select the down arrow (to the right of Declarations as illustrated below) and select **Click** from the list.



- c. Add the following code for the **btnUpdate Click** event:

```
itm = mfunctions.Replace(itm, 1, txtOrgName.Text) ' Replace Org
name
```

```
itm = mfunctions.Replace(itm, 3, txtCity.Text) ' Replace City
itm = mfunctions.Replace(itm, 4, txtState.Text) ' Replace State
MVSP.FileWrite("CUSTOMERS", txtCustId.Text, itm)
If MVSP.statusCode <> 0 Then
    MsgBox("Error updating item: " & MVSP.statusMessage)
    Exit Sub
End If
txtCustId.Text = ""
txtCity.Text = ""
txtOrgName.Text = ""
txtState.Text = ""
txtCustId.Focus()
```

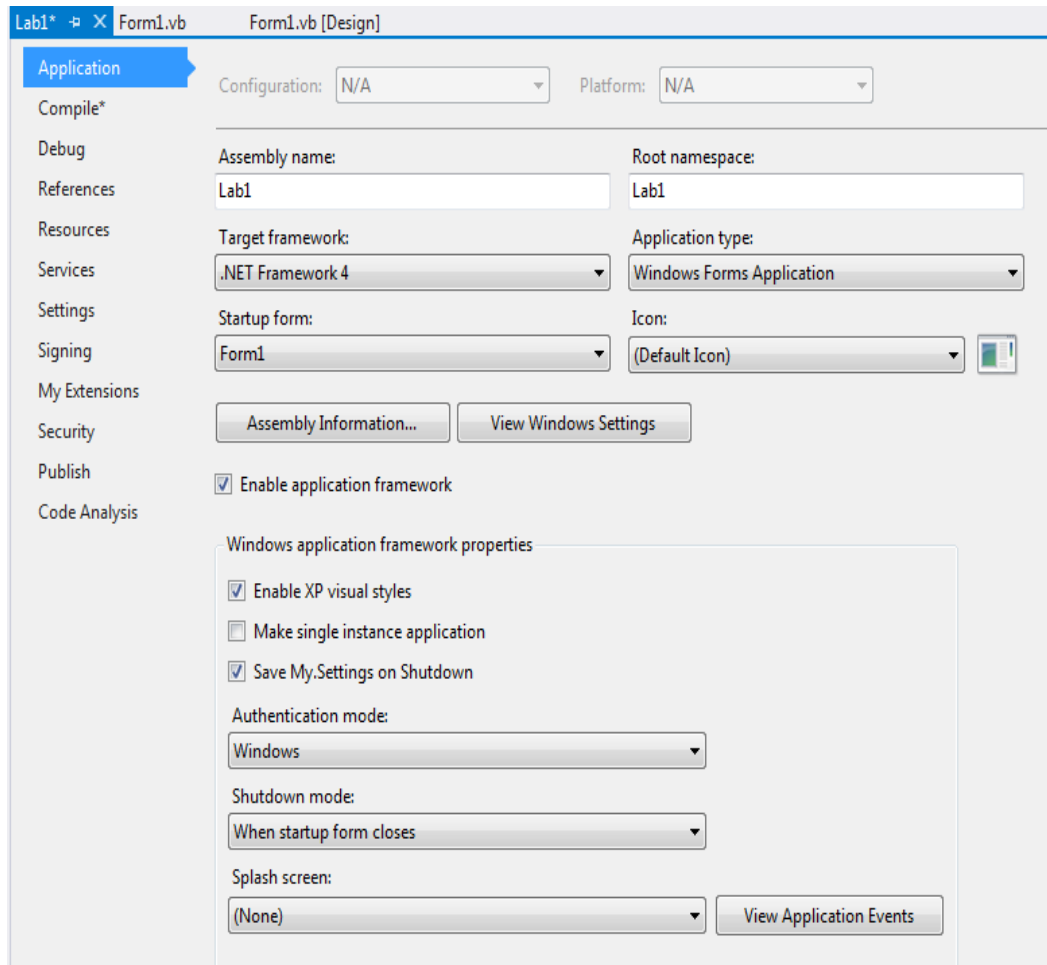
d. The code to handle the update should now look like this:

```
Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
    itm = mfunctions.Replace(itm, 1, txtOrgName.Text) ' Replace Org name
    itm = mfunctions.Replace(itm, 3, txtCity.Text) ' Replace City
    itm = mfunctions.Replace(itm, 4, txtState.Text) ' Replace State
    MVSP.FileWrite("CUSTOMERS", txtCustId.Text, itm)
    If MVSP.statusCode <> 0 Then
        MsgBox("Error updating item: " & MVSP.statusMessage)
        Exit Sub
    End If
    txtCustId.Text = ""
    txtCity.Text = ""
    txtOrgName.Text = ""
    txtState.Text = ""
    txtCustId.Focus()

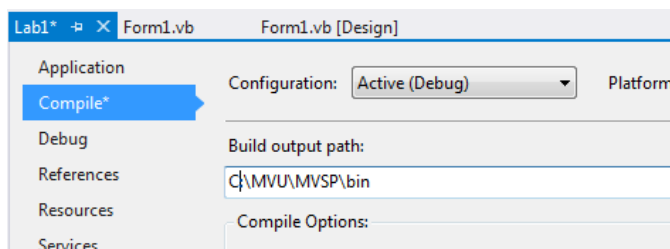
End Sub
```

___ 6. Changing the output location and build.

- a. On the Project menu, select the last item (should be the properties for the name of your project).
- b. A screen like the one below displays.



- c. Click **Compile*** from the menu on the left.
- d. Set the Build output path to **C:\MVU\MVSP\bin**.



- e. From the BUILD menu, select the Build option to build your project.
- f. If this is built successfully, you should see a Build Successful message at the bottom of the Visual Studio window.
- g. From Windows Explorer, navigate to the following directory:
C:\MVU\MVSP\MVSPNETAPI\d3iphelper
- h. Copy the d3iphelper64.dll file into the C:\MVU\MVSP\bin directory.

Exercise 3 summary: Creating the code to connect, read and update data.

End of Exercise 3

Exercise 4: Testing the code

Purpose of the Exercise

This exercise demonstrates using an Enterprise license from MVSP.

After this exercise you will be able to:

- See how Enterprise licenses are used
- Run your newly created application to connect, read and write data using MVSP

Exercise Instructions

Perform the following steps:

- ___ 1. Check the existing license count.
 - a. From the Windows Start menu, open a Command Prompt.
 - b. Connect to D3.

```
telnet localhost
dm: dm
master dictionary: dm
term ,48
```
 - c. Enter the `maxusers` command.

You should see 1 User licenses used and 0 Enterprise licenses used.
- ___ 2. Running your application.
 - a. From Windows Explorer, navigate to the C:\MVU\MVSP\bin directory.
 - b. Double-click the executable that you generated.
 - c. Check the **Enterprise License** checkbox and click the **Connect** button.

If the button becomes disabled, then you are connected. Otherwise, an error will be displayed.
- ___ 3. Check the license usage again.
 - a. In the telnet session, enter `maxusers` again. Now the User count should have incremented to 2 and the Enterprise count should have incremented to 1.
- ___ 4. Opening additional copies of your application.

- a. Repeat steps 2 and 3 above, 4 more times. With each repeat of steps 2 and 3, you should see that the User count remains at 2 and the Enterprise count remains at 1.
- b. Enter **listu** to list the connected users. You should see 5 pibs logged on to the MVDEMO account and your pib logged on to the DM account.

___ 5. Testing your application

- a. In one of the open instances of your application, enter a customer number (from 1 to 50) and press Enter.
- b. Change any or all of the 3 fields (**Organization Name**, **City** and **State**) and click the **Update** button.
- c. From another copy of your application, enter the same customer number and press Enter. You should see your updated data.

___ 6. Testing item locking in your application

- a. In another instance of your application, enter the same customer number again and press Enter. You should see a message box informing you that the item is locked.

___ 7. Testing using User licenses

- a. Close the 5 copies of your application.
- b. Check **maxusers** in your telnet session. It should be 1 User license used and 0 Enterprise licenses used.
- c. Now repeat steps 1 to 4, but this time do not check the **Enterprise License** checkbox. After you have opened 5 copies, check **maxusers** in your telnet session again. It should be 5 User licenses used and 0 Enterprise.

Exercise 4 summary: Understand how Enterprise licenses are used and how to connect, read and write data using MVSP.

End of Exercise 4