Programming Manage 2000 Awareness Into Excel

With Microsoft Visual Studio Tools for Office (VSTO), you can develop solutions in Visual Studio to customize Office applications and add the specific features you need for your business processes. With Manage 2000 Web Services, you can make your VSTO solution aware of Manage 2000 validation routines and business objects and even allow your VSTO solution to push information back into Manage 2000 business objects.

At the conclusion of this lab, you will be able to:

Create Visual Studio Tools For Office Excel Projects

Add Manage 2000 validation to Excel sheet entry

Create Office Ribbon Tabs with a task-oriented user interface

Create Office Ribbon tasks with Manage 2000 Awareness

Create a New VSTO Excel Template Project

1. In Visual Studio, select ***File | New Project***
2. Select ***Office***
3. Select ***Excel 2010 Template***
4. Type **MyM2KExcelTemplate** for *Name*
5. Click the ***OK*** button
6. Click the ***OK*** button to accept the *Create a New Document* option
7. If prompted for permission Click the ***OK*** button to allow access to the Visual Basic for Applications project system

Add an Event Handler To Detect Changes In Column 1

1. From the Solution Explorer, right-click on ***Sheet1.vb*** and select ***View Code***
2. Select ***(Sheet 1 Events)*** in the left dropdown list at the top of the coding editor window
3. Select ***Change*** in the right dropdown list at the top of the coding editor window
4. Add code to call a validation routine when the user is changing column 1

```
'Validate Column 1 to CLASS.TRAIN Customer Master File
If Target.Column = 1 Then
   ValidateItem(Target, "CM", "0")
End If
```

5. After the *End Sub*, stub out the Validate Item Method for testing

```
Private Sub ValidateItem(Target As Excel.Range, FileName As String, TableNbr
As String)
   MessageBox.Show("Validating user entry " & Target.Value & " in column " &
      Target.Column.ToString & " to the " & FileName & " file.")
End Sub
```

6. Click the ***Start*** button in the toolbar to test
   a. Make sure your event handler runs when the user exits column 1 after changing data and does not run from other columns
   b. Click the ***Don't Save*** button when you close the window

Add a Web Reference to the Manage 2000 ERPBusinessObject Service

```
Adding this web reference will provide your project with access to the SOAP client
class containing web services from a specific Manage 2000 account.
```

1. From the Solution Explorer, right-click on the project level item ***MyM2kExcelTemplate*** (in bold) and select ***Properties***
2. Click on ***References***
3. Click on the down arrow on the right side of the ***Add*** button and select ***Service Reference***
4. Type **http://localhost/CLASS.TRAIN/MT/ERPBusinessObjectService/ERPBusinessObjectService.asmx** in the *Address* box
   4.1. Alternatively run WEB.SERVICES from a PWS session. This will open the ERPBusinessObjectService API specification in a browser.
   4.2. Copy the address upto the .asmx file extension
   4.3. Paste the address into the Visual Studio Service Reference wizard
5. Click the ***Go*** button
6. Type **ClassTrainWebService** for *Namespace*
7. Click the ***OK*** button

Write Code to Implement Manage 2000 Awareness in Your ValidateItem Subroutine

```
You can now write code to create an instance of this class which will provide your
code with methods to access Manage 2000.
```

1. From Sheet1.vb code view in your ValidateItem stub, comment out the existing call to MessageBox

```
' MessageBox.Show("Validating user entry " & Target.Value & " in column " &
Target.Column.ToString & " to the " & FileName & " file.")
```

2. Create an instance variable for the SoapClient class from the ClassTrainWebService ServiceReference

```
Dim m2kWebService As New ClassTrainWebService.ERPBusinessObjectServiceSoapClient
```

3. Get the value to validate from the Excel Range Target triggering the event

```
Dim ItemID As String = Target.Value
```

4. Dim some work variables for the arguments in the web service method

```
Dim IsValid As Boolean = False
Dim InItemID As String = ItemID
Dim ResolvedID As String = ItemID
Dim ValidationDisplay As String = String.Empty
```

5. Use the web service to validate the entered value

```
    Try
     'Talk to Manage 2000 Validation Routines
     m2kWebService.ValidateRecordKey(FileName, TableNbr, ItemID, IsValid,
ValidationDisplay)
     'The returned ItemID may have been resolved,i.e. Fac code appended
     If ItemID <> InItemID Then
         ResolvedID = ItemID
         ItemID = InItemID
     End If

     'Update ActiveCell with results
       If IsValid Then
           Application.EnableEvents = False
           Target.Value = ResolvedID
           Target.Next.Value = ValidationDisplay
           Application.EnableEvents = True
       Else
           MessageBox.Show(ItemID & " is not a valid customer in CLASS.TRAIN.")
           Target.Next.Value = "Not On File"
           Target.Activate()
          End If
  Catch ex As Exception
      MessageBox.Show(ex.Message)
  End Try
```

6. In the ***Sheet1_Startup*** event handler add the following code to automatically resize and format Column 1 as Text

```
 Application.EnableEvents = False
 With Globals.ThisWorkbook.ActiveSheet.Application.Range("A1")
     .Value = "Customer Nbr"
     .ColumnWidth = 15
     .style.NumberFormat = "@"
 End With
 With Globals.ThisWorkbook.ActiveSheet.Application.Range("B1")
     .Value = "Customer Name"
     .ColumnWidth = 30
     .style.NumberFormat = "@"
 End With
 'Start the user in column 1 row 2
 Globals.ThisWorkbook.ActiveSheet.Application.Range("A2").Activate()
 Application.EnableEvents = True
```

Test Manage 2000 Validated Excel Sheet Entry

1. Click the ***Start*** button in the toolbar
2. Try typing in valid and non-valid customer numbers (Valid numbers will include 1008, 1016, 1024, 1032, 1048, 1064)
3. Click the ***Don't Save*** button when you close the window

Add a Ribbon Item to Your Project

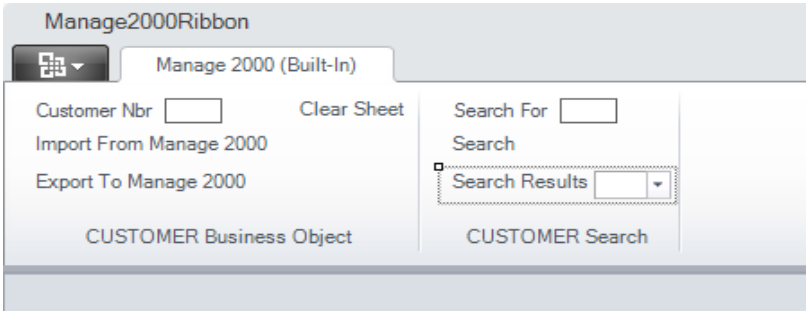
```
Ribbon items provide a tabbed user interface space at the top of Excel. By adding one
to your project you can add-in to Excel your own tab of controls.
```

1. From the Solution Explorer, right-click on the project level item ***MyM2kExcelTemplate*** (in bold) and select ***Add*** | ***New Item…***
2. From *Installed | Common Items| Office* select **Ribbon (Visual Designer)**
3. Type **Manage2000Ribbon** for *Name*
4. Click the ***Add*** button

Design the Ribbon User Interface for the Features You Want

```
Ribbon tabs must contain special UI controls from the Microsoft.Office.Ribbon.Tools
namespace.  You will find them in the toolbox under Office Ribbon Controls.
```

1. Select the tab and open the Properties Window
2. Change *(Name)* to **tabManage2000** and change *Label* to **Manage 2000**
3. Drag controls from the ***Office Ribbon Controls*** tab in the ***Toolbox*** onto the Manage 2000 ribbon tab in the Design window and set the following properties in the Property Editor
   3.1. Change the group *Label* property to **CUSTOMER Business Object**
   3.2. Add an EditBox control, *(Name)* = **edtCustomerNbr,** *Label* **= Customer Nbr**
   3.3. Add a Button control, *(Name)* = **btnImport,** *Label* = **Import From Manage 2000**
   3.4. Add a Button control, *(Name)* = **btnExport,** *Label* = **Export To Manage 2000**
   3.5. Add a Button control, *(Name)* = **btnClear,** *Label* **= Clear Sheet**
   3.6. Add a Group control, *Label* = **CUSTOMER Search**
   3.7. Add an EditBox control, *(Name)* = **edtSearchFor,** *Label* = **Search For**
   3.8. Add a Button control, *(Name)* = **btnSearch,** *Label* = **Search**
   3.9. Add a Dropdown control, *(Name)* = **ddlSearchResults,** *Label* **Search Results**
   3.10. Your ribbon will look like this



Test the New Manage 2000 Tab UI

1. Click the ***Start*** button in the toolbar
2. Select the ***Manage 2000*** tab in the ribbon
3. Click the ***Don't Save*** button when you close the window

Create an Instance Variable for the SoapClient Class from the ClassTrainWebService ServiceReference

1. Right click on the Ribbon designer and select ***View Code***
2. Immediately after the *Public Class Manage2000Ribbon* declaration add the following code

```
Private M2kBTOWebService As New
ClassTrainWebService.ERPBusinessObjectServiceSoapClient
Private dsCUSTOMER As New DataSet
Private CustomerList As Microsoft.Office.Tools.Excel.ListObject
Private InvoiceList As Microsoft.Office.Tools.Excel.ListObject
```

Write the Code Needed to Read a Manage 2000 Business Object

```
Manage 2000 business objects provide and consume XML which can also be accessed in
the form of a .Net DataSet. Accessing the dataset of a business object allows you to
bind datatable columns to spreadsheet columns.
```

1. In the designer, double-click on the ***Import*** button to create the event handler
2. Add the following code within the btnImport_Click subroutine

```
ImportCUSTOMERFromM2k(edtCustomerNbr.Text)
```

3. Add a subroutine after the End Sub to get the Customer from the web service

```
    Private Sub ImportCUSTOMERFromM2k(CustomerNbr As String)
        Try
            'Ask the ERPBusinessObjectService for the CUSTOMER dataset for this
CustomerNbr
            dsCUSTOMER = M2kBTOWebService.GetBusinessObjectDS("CUSTOMER", CustomerNbr)
            'Move the data from the dataset to the sheet cells
            DataBindDataToSheet()
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    End Sub
```

4. Add a subroutine to databind the data from dataset to the sheet

```
Private Sub DataBindDataToSheet()
    Dim WorkSheet As Excel.Worksheet 'Microsoft Office Interop Excel Worksheet Interface
    Dim ExtendedWorkSheet As Worksheet 'Microsoft Office Tools Excel Worksheet Interface
    'Make a list of desired columns from within the tables
    Dim CustColumns() As String = {"Name", "Address", "City", "State", "Zip"}
    Dim InvoiceColumns() As String = {"Invoice_Nbr", "Invoice_Date", "Invoice_Amt"}
    'Create list objects for tables that are to be bound
    With Globals.ThisWorkbook.ActiveSheet.Application
           'Cast a value of worksheet from Globals as a typed worksheet
        WorkSheet = DirectCast(.ActiveWorkbook.Worksheets(1), Excel.Worksheet)
        'Extend the worksheet using GetVstoObject so it has a Controls collection to hold
    the ListObject for databinding
        ExtendedWorkSheet = Globals.Factory.GetVstoObject(WorkSheet)
        CustomerList = ExtendedWorkSheet.Controls.AddListObject(.Range("C1", "G1"),
    "CustList")
        InvoiceList = ExtendedWorkSheet.Controls.AddListObject(.Range("H1", "J1"),
    "InvList")
    End With
    CustomerList.AutoSetDataBoundColumnHeaders = True
    InvoiceList.AutoSetDataBoundColumnHeaders = True
    'Fill the cells in the ListObject range from the dataset table values
    'Changes typed into the cells will also automatically update the dsCUSTOMER table
values
    CustomerList.SetDataBinding(dsCUSTOMER, "CUSTOMER", CustColumns)
    InvoiceList.SetDataBinding(dsCUSTOMER, "CUSTOMER_Invoices", InvoiceColumns)
    'Autofit columns to data
    Globals.ThisWorkbook.ActiveSheet.Columns("A:J").AutoFit()
End Sub
```

Test Importing Manage 2000 Information Into Excel

1. Click the ***Start*** button in the toolbar
2. Select the ***Manage 2000*** tab in the ribbon
3. Enter 1006 in the Customer Nbr textbox
4. Click on the Import Button
5. Enter 1010 in the Customer Nbr textbox
6. Click on the Import Button (We will fix this problem in the next section)
7. Click the ***Don't Save*** button when you close the window

Write the Code to Clear the Sheet

1. In the designer, double-click on the ***Clear*** button to add an event handler
2. Add the following code within the btnClear_Click subroutine

```
    'Clear Existing Contents
    If Not CustomerList Is Nothing Then CustomerList.Delete()
    If Not InvoiceList Is Nothing Then InvoiceList.Delete()
    With Globals.ThisWorkbook.ActiveSheet.Application
        .EnableEvents = False
        'Clear cells
```

```
            .Range("2:99").ClearContents()
            'Set Validated Column Titles
            .Range("A1").Value2 = "Customer Nbr"
            .Range("A1").ColumnWidth = 15
            .Range("B1").Value2 = "Customer Name"
            .Range("B1").ColumnWidth = 30
            .EnableEvents = True
        End With
```

Test Reading a Business Object from Manage 2000

1. Click the ***Start*** button in the toolbar
2. Select the ***Manage 2000*** tab in the ribbon
3. Type **1006** in the Customer Nbr textbox and click the ***Import From Manage 2000*** button
   a. Other valid CUSTOMER records include, 1010, 1017, 1018, 1021
   b. Click the ***Clear*** button before importing a new Customer
4. Click the ***Don't Save*** button when you close the window

Write the Code Needed to Write a Manage 2000 Business Object

```
The Business Object Framework is capable of posting data into Manage 2000 applying
business rules and optimistic record locking.
```

1. In the designer, double-click on the ***Export To Manage 2000*** button to add an event handler
2. Add the following code within the btnExport_Click subroutine

   ```
   ExportCUSTOMERToM2k(edtCustomerNbr.Text)
   ```

3. Add a subroutine after the End Sub to post the business object

```
    Private Sub ExportCUSTOMERToM2k(CustomerNbr As String)
        Try
            'Post the CUSTOMER dataset back into Manage 2000 account CLASS.TRAIN
            Dim Result As String = String.Empty
            Result = M2kBTOWebService.PostBusinessObjectDS("CUSTOMER", CustomerNbr,
dsCUSTOMER)
            If Result <> String.Empty Then
                MessageBox.Show(Result)
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
 End Sub
```

Test Writing a Business Object from Manage 2000

1. Click the ***Start*** button in the toolbar
2. Select the ***Manage 2000*** tab in the ribbon

3. Enter **1006** in the Customer Nbr edit box and click the ***Import From Manage 2000*** button
   a. Other valid CUSTOMER records include, 1010, 1017, 1018, 1021
4. Select the ***State*** textbox and change it to another state like **'TX**
   a. Normal Excel behavior for entering non-numeric data in cells is to begin text with a single unmatched quote
5. Click on the ***Export To Manage 2000*** button
6. Click on the ***Clear*** button to clear the display
7. Click on the ***Import From Manage 2000*** button to see your updated record
   a. You can also check the update from PWS. Simple type **ED CUSTOMER 1006** and check the value with the line editor (or use EDP if you prefer)
8. Try to set the state to an invalid state code like 'XX. After entering the value click on the Export To Manage 2000 button. This should error out if the states table does not include XX.
9. Click the ***Don't Save*** button when you close the window

Write the Code Needed to Search Manage 2000 for a Customer

Manage 2000 Web and PWS interfaces provide a robust cross referencing experience for users. Here is one approach to providing similar features to Excel users.

1. Immediately after the *Public Class Manage2000Ribbon* declaration add the following global search result variables

```
Private CookieID As String = String.Empty
Private CurrentPage As Integer = 1
Private ItemCount As Integer = 0
Private PageSize As Integer = 10
Private PageCount As Integer = 0
```

2. In the designer, double-click on the ***Search*** button to add an event handler
3. Add the following code within the btnSearch_Click subroutine

```
DoSearch(edtSearchFor.Text)
```

4. Create a subroutine after the End Sub to do the Search using Web Service to call CROSS.REF

```
 Private Sub DoSearch(Target As String)
     'Get possible records
     PageCount = 0
     ItemCount = 0
     CookieID = ""
     'Clear Found Items
     ddlSearchResults.Items.Clear()
     CurrentPage = 1
     'Talk to Manage 2000 Cross Referencing
     Try
         M2kBTOWebService.CrossRefSelect("CUSTOMER", "0", "ALL", Target, PageSize,
PageCount, ItemCount, CookieID)
     Catch ex As Exception
         MessageBox.Show(ex.Message)
     End Try
```

```
        If ItemCount > 0 Then
            LoadItemsFound(CurrentPage)
        Else
            Dim NoItems As Microsoft.Office.Tools.Ribbon.RibbonDropDownItem
            NoItems = Factory.CreateRibbonDropDownItem
            NoItems.Label = "No Items Found."
            ddlSearchResults.Items.Add(NoItems)
        End If
    End Sub
```

5. Create a subroutine to Load Items found into the Search result dropdown

```
    Private Sub LoadItemsFound(PageNbr As Integer)
        'Clear Found Items
        ddlSearchResults.Items.Clear()
        'Get a page of keys and validation displays
        Dim KeyList() As String
        Try
            KeyList = M2kBTOWebService.GetKeysFromCookie(CookieID, PageNbr)
            Dim ValDisplays() As String
            Dim sKeyList As String = String.Empty
            Dim delim As String = String.Empty
            For Each key As String In KeyList
                sKeyList = sKeyList & delim & key
                delim = ","
            Next
            ValDisplays = M2kBTOWebService.GetValidationDisplays("CUSTOMER", "0", ",",
sKeyList, "Name")
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try

        'Add display of items found
        Dim FirstItem As Microsoft.Office.Tools.Ribbon.RibbonDropDownItem
        FirstItem = Factory.CreateRibbonDropDownItem
        Dim StartItem As Integer = (PageNbr - 1) * PageSize + 1
        Dim EndItem As Integer = StartItem + PageSize - 1
        If EndItem > ItemCount Then EndItem = ItemCount
        FirstItem.Label = StartItem.ToString & " - " & EndItem.ToString & " of " &
ItemCount.ToString
        ddlSearchResults.Items.Add(FirstItem)

        If PageNbr > 1 Then
        'Add previous page option
            Dim PreviousItems As Microsoft.Office.Tools.Ribbon.RibbonDropDownItem
            PreviousItems = Factory.CreateRibbonDropDownItem
            PreviousItems.Label = "[Previous]"
            ddlSearchResults.Items.Add(PreviousItems)
        End If

        'Add this page of found keys
        For Idx As Integer = 0 To KeyList.Length - 1
            Dim ribItem As Microsoft.Office.Tools.Ribbon.RibbonDropDownItem
            ribItem = Factory.CreateRibbonDropDownItem
            ribItem.Label = KeyList(Idx) & " " & ValDisplays(Idx)
            ddlSearchResults.Items.Add(ribItem)
        Next
```

```
        If PageCount > CurrentPage Then
        'Add next page option
            Dim MoreItems As Microsoft.Office.Tools.Ribbon.RibbonDropDownItem
            MoreItems = Factory.CreateRibbonDropDownItem
            MoreItems.Label = "[More]"
            ddlSearchResults.Items.Add(MoreItems)
        End If
    End Sub
```

6. Test here if you would like a break from coding before finishing up the search feature.
7. In the designer, double-click on the ***Result*** dropdown to add an event handler
8. Add the following code within the ddlSearchResults_SelectionChanged subroutine

```
        If ddlSearchResults.SelectedItem.Label = "[More]" Then
            CurrentPage += 1
            LoadItemsFound(CurrentPage)
        ElseIf ddlSearchResults.SelectedItem.Label = "[Previous]" Then
            CurrentPage -= 1
            LoadItemsFound(CurrentPage)
        Else
            'Put results in current active cell
            'Globals.ThisWorkbook.ActiveSheet.Application.ActiveCell.Value2 =
ddlSearchResults.SelectedItem.Label
            'Put results in Business Object Customer Number edit box
            edtCustomerNbr.Text = ddlSearchResults.SelectedItem.Label.Split(" ")(0)
        End If
```

Test Searching the CUSTOMER File for a Key

1. Click the ***Start*** button in the toolbar
2. Select the ***Manage 2000*** tab in the ribbon
3. Type **Plating** in the *Search For* edit box and then click the ***Search*** button
    a. Other words to search for include ltd, inc, cogs, stamping, engineering
    b. You should have 3 items found from Plating
    c. Select the Search Results dropdown and choose one of the customer
    d. It should appear in the Business Object Customer Nbr textbox ready for importing
    e. Click the ***Clear*** button before importing a new Customer
4. Click the ***Don't Save*** button when you close the window