

Lab 9: Using Microsoft Transaction Server

To complete the lab exercises in this chapter, you must have the required software. For detailed information about the labs and setup for the labs, click Labs in this course.

Objectives

After completing this lab, you will be able to:

- ◆ Create a new package by using Microsoft Transaction Server Explorer.
- ◆ Add components to a package, and set their properties by using Microsoft Transaction Server Explorer.
- ◆ Monitor transaction statistics by using Microsoft Transaction Server Explorer.
- ◆ Create a component that supports MTS transactions by calling either the SetComplete method or the SetAbort method.
- ◆ Use the MTS Context object to create other components in the same transaction.

Prerequisites

Before completing this lab you must be able to:

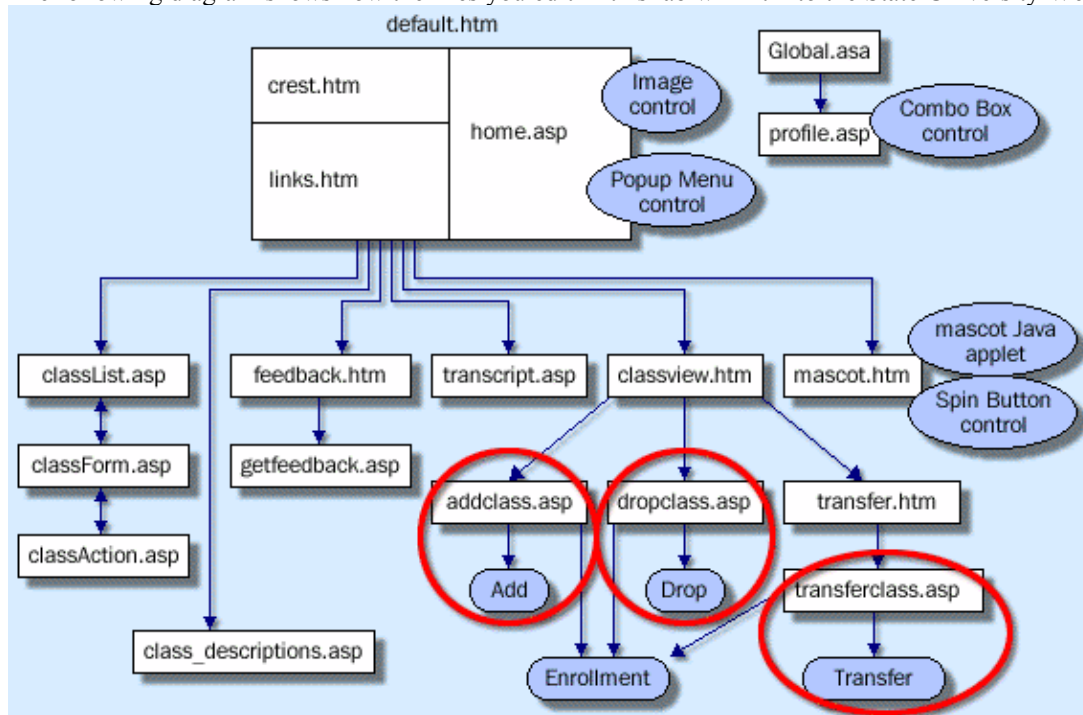
- ◆ Create Web pages, .asp files, and projects in Microsoft Visual InterDev.
- ◆ Call an ActiveX server component from an .asp file.
- ◆ Create an ActiveX server component by using Visual Basic 5.0

Lab Setup

To complete this lab, you need the following:

- ◆ The Visual InterDev State University Project
- ◆ The State University SQL database
- ◆ Microsoft Transaction Server
- ◆ Visual Basic 5.0

The following diagram shows how the files you edit in this lab will fit into the State University Web site.



Estimated time to complete this lab: **90 minutes**

Note There are project and solution files associated with this lab. If you installed the labs during Setup, these files are in the folder <Install Folder>\Labs\Lab09 on your hard disk. If you did not install the labs during Setup, you can find them in the \Labs\Lab09 folder of this *Mastering Web Site Development* CD-ROM.

Exercises

The following exercises provide practice working with the concepts and techniques covered in Chapter 9: Using Microsoft Transaction Server.

Exercise 1: Creating the State University Package

In this exercise, you will create a new package in Microsoft Transaction Server that contains all of the components for State University. You will then package the **Add** component, and test it by calling it from an .asp file.

Exercise 2: Adding Transaction Support

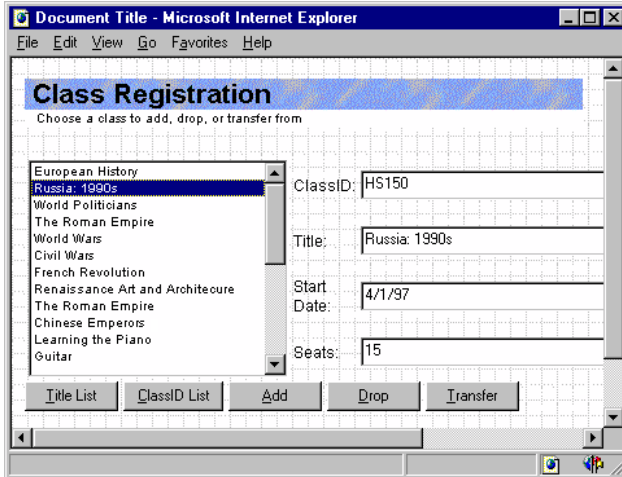
In this exercise, you will modify the **Drop** object to call **SetComplete** or **SetAbort** to support transactions in Microsoft Transaction Server. You will also modify the **Transfer** object to create the **Add** and **Drop** objects using **CreateInstance**.

The **Transfer** object will then call the **Add** and **Drop** objects to perform the work of transferring a student. Finally, you will add the **Drop** and **Transfer** objects to the new package, and then test the package by calling the objects from .asp files.

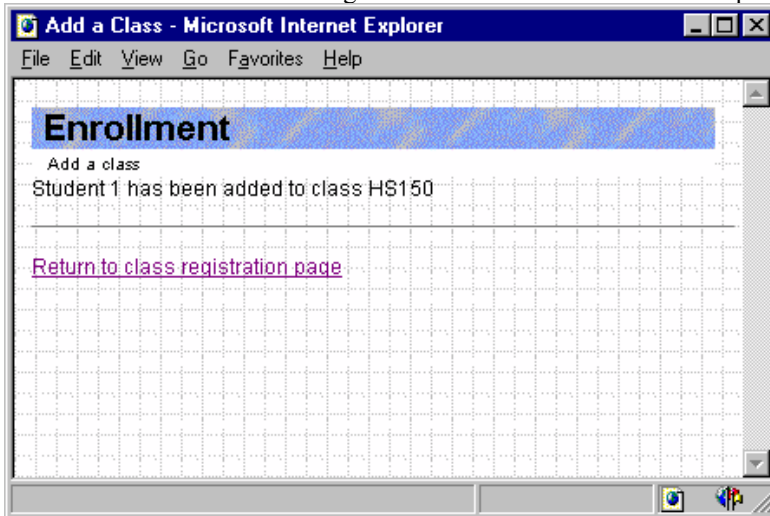
Exercise 1: Creating the State University Package

In this exercise, you will create a new package in Microsoft Transaction Server that will contain all of the ActiveX server components for the State University Web site. You will add the **Add** component to the new package, and then test the package by calling it from an .asp file.

You will use the classview.htm file to add and drop classes. The following illustration shows how classview.htm looks.



When you add a class using the classview.htm file, the addclass.asp file calls the **Add** object to add a student to a class. The following illustration shows how addclass.asp looks when returned to the student.



➤ Create the State University package

1. Open the Microsoft Transaction Server Explorer.
2. Create a new package named State University. When asked to set the package identity, set it to **Interactive user**.

➤ **Add a component to the State University package**

1. Create a folder on the Web server named \busobjects.
2. Copy the following files from the \MWD\Labs\Lab09 folder to the \busobjects folder:
 - add.vbp
 - add.cls
 - add.dll
 - drop.vbp
 - drop.cls
 - transfer.vbp
 - transfer.cls

All of the above files are Visual Basic 5.0 project files. They have already been compiled into a DLL named **Add** component.

3. Open the State University Web project in Visual InterDev, and add the following files from the \MWD\Labs\Lab09 folder to the root of the Web project:
 - transferclass.asp
 - dropclass.asp
 - addclass.asp

If copies of these files are already in the project, overwrite the existing versions with the files from \MWD\Labs\Lab09.

4. Add the component add.dll to the State University package by dragging the file from the Windows Explorer, or by clicking **New** from the **File** menu in Microsoft Transaction Server Explorer.
5. In the State University package, set the transaction properties for the **Add** component to **Requires a transaction**. Set the **Activation** property for the **Add** component to **In the creator's process**.
6. Test the **Add** object by opening classview.htm in the browser.

View the transaction statistics in Microsoft Transaction Server Explorer. Log in to the profile using student ID 1 or any other valid student ID, and try adding yourself to classes.

If you attempt to add yourself to a class that you are not enrolled in, the transaction should complete successfully and the statistics will show that a transaction completed.

If you attempt to add yourself to a class that you are already enrolled in, the transaction should fail and the statistics will show that a transaction aborted.

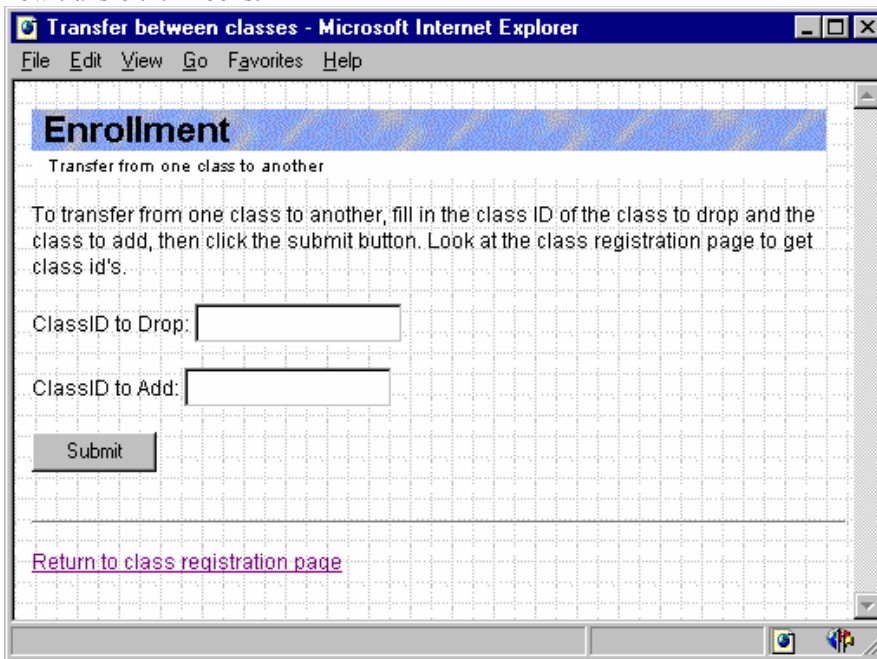
7. Use DataView to verify that the Enrollment table has been updated correctly when classes are added. You can also verify that classes are added by retrieving the transcript.asp file.

Exercise 2: Adding Transaction Support

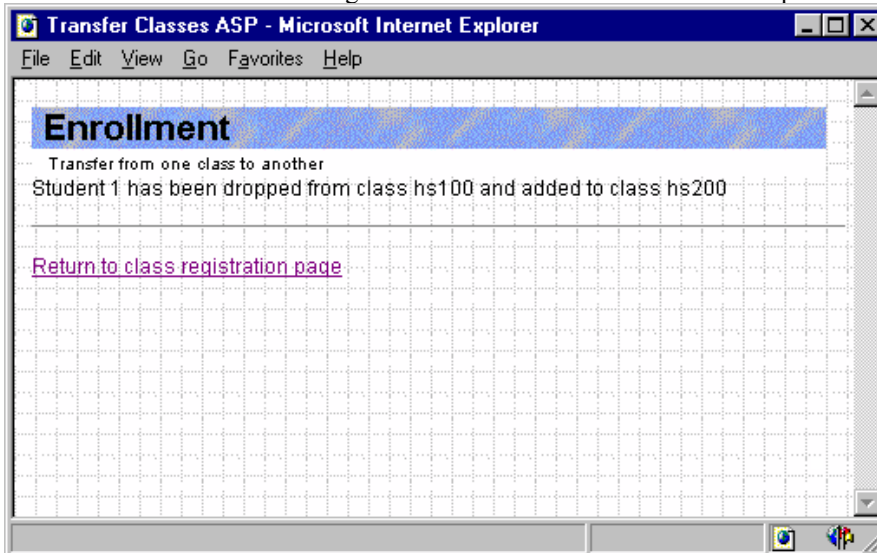
In this exercise, you will modify the **Drop** object to call the **SetComplete** or **SetAbort** method. You will also modify the **Transfer** object to use the **CreateInstance** method, so that the object can create the **Add** and **Drop** objects. The **Transfer** object will then call the **Add** and **Drop** objects to perform the work of transferring a student.

Finally, you will add the **Drop** and **Transfer** objects to the new package you created in the previous exercise, and test the package by calling the objects from .asp files.

You will use the transfer.htm file to submit a form that transfers a student. The following illustration shows how transfer.htm looks.



When you transfer a student using the transfer.htm file, the transferclass.asp file calls the Transfer object to transfer a student. The following illustration shows how transferclass.asp looks when returned to the student.



➤ Add the Drop object

1. In Visual Basic 5.0, open the Drop project from the \busobjects folder.
2. On the **Project** menu, click **References**, select **Microsoft Transaction Server 1.0 Type Library**, then click OK.
3. Open the class module drop.cls.
4. At the beginning of the **Drop** function, assign the **Drop** function 0 as a default return value. The return value is used by the .asp file that calls the object to determine if it worked or not.
5. Add code to retrieve the **Context** object from Microsoft Transaction Server.
6. Find the call to the **ClassCompleted** function. Add a line of code to call **SetAbort** if **ClassCompleted** is True.
If the function returns a value of True, it indicates that the student has completed the class, so the **Drop** function aborts.
7. In the error handler, add code to call the **SetAbort** method to indicate that any changes made by the **Drop** function should be undone.
8. After the conn.close statement, add code that calls the **SetComplete** method.
9. Build drop.dll by clicking **Make Drop.dll** from the **File** menu.
10. Use the Microsoft Transaction Server Explorer to add the **Drop** object to the State University package.
11. In the State University package, set the transaction properties for the **Drop** component to **Requires a transaction**.
12. In the State University package, set the **Activation** property for the **Drop** component to **In the creator's process**.
13. Test the **Drop** component by opening the file classview.htm in the browser.
Log in to the profile using student ID 1 and try dropping the MT100 class. The attempt should fail because the class is already completed. The failure should be logged as an aborted transaction.
Try dropping other classes. These drops should succeed and be logged as completed transactions.
14. Use DataView to verify that the Enrollment table has been updated correctly when classes are dropped. You can also verify that classes are dropped by retrieving the transcript.asp file.

Note If you drop a class in which the current student ID is not enrolled, the drop will still succeed. This is because the **Drop** method uses the SQL DELETE statement to remove records from the enrollment table. The SQL DELETE statement will succeed even if the record does not exist.

➤ Add transaction support to the Transfer object

1. In Visual Basic 5.0, open the Transfer project from the \busobjects folder.
2. In the project, add the library references **Microsoft Transaction Server 1.0 Type Library**, **State University Add Object**, and **State University Drop Object**.

Note For their references to be available, the **Add** and **Drop** objects must be registered. To register the objects, add them to the State University package in Microsoft Transaction Server.

3. Open the class module transfer.cls.
4. At the beginning of the **Transfer** function, assign the **Transfer** function 0 as a default return value. The return value is used by the .asp file that calls the object to determine if it worked or not.
5. Add code to retrieve the **Context** object from Microsoft Transaction Server.
6. Add code to create the **Add** and **Drop** objects by calling the **CreateInstance** method of the **Context** object. Assign the created objects to variables named **addobj** and **dropobj**, respectively.

7. Before the **Exit Function** call, add code that calls the **SetComplete** method to indicate that the **Transfer** method has completed successfully.
8. In the error handler, add code that calls the **SetAbort** method to indicate that any changes should be undone.
9. To build the Transfer.dll, click **Make Transfer.dll** on the **File** menu.
10. Use Microsoft Transaction Server Explorer to add the **Transfer** object to the State University package.
11. In the State University package, set the transaction properties for the **Transfer** component to **Requires a transaction**. Set the **Activation** property for the **Transfer** component to **In the creator's process**.
12. Test the **Transfer** object by opening transfer.htm in the browser.
Log in to the profile using student ID 1 and try transferring to a class in which that student is already enrolled. The transfer should fail and be logged as an aborted transaction.
Then, try transferring between valid classes. The transfer should succeed and be logged as a completed transaction.
13. Use DataView to verify that the Enrollment table has been updated correctly. You can also verify that classes are transferred by retrieving the transcript.asp file.