

STARPRINT LIMITED

Student Name	Joginder S Nahil
Lecturer's Name	Dr. Aaron Mann
General Date	07/10/2007 14:50:00
Long Date	07 October 2007
Short Date	07/10/2007
Long Time	14:50:00
Short Time	14:50

```

00001 ' Windows Script Host Sample Script
00002 '
00003 ' -----
00004 ' Copyright (C) 1996 Microsoft Corporation
00005 '
00006 ' You have a royalty-free right to use, modify, reproduce and distribute
00007 ' the Sample Application Files (and/or any modified version) in any way
00008 ' you find useful, provided that you agree that Microsoft has no warranty,
00009 ' obligations or liability for any Sample Application Files.
00010 ' -----
00011 '
00012 'This script is adds users from the Windows NT DS
00013 'via ADSI. The script reads an EXCEL spreadsheet that contains a page
00014 'of users to add.
00015 '
00016 'The sample uses the directory root "LDAP://DC=ArcadiaBay,DC=Com,O=Internet"
00017 'Change the directory path in the EXCEL spreadsheet to match your DS
00018 'before running this sample.
00019 '
00020 '
00021 '
00022 'To add users, run ADDUSERS.VBS with %windir%\Your Samples Directory
» Here"\AddUsers.XLS.
00023 'To Delete users, run DELUSERS.VBS with %windir%\Your Samples Directory
» Here"\DelUsers.XLS.
00024
00025
00026 Dim oXL
00027 Dim u
00028 Dim c
00029 Dim root
00030 Dim ou
00031 Dim TextXL
00032 Dim CRLF
00033 Dim oArgs
00034
00035
00036 'Get the command line args
00037 Set oArgs=wscript.arguments
00038
00039 CRLF = Chr(13) & Chr(10)
00040
00041 'If no command line arguments provided, prompt for file containing users to
» add/delete
00042 If oArgs.Count = 0 Then
00043     TextXL = InputBox("This scripts reads an Excel spreadsheet and adds" & _
00044         "users from the Windows NT DS via ADSI." & CRLF & CRLF & _
00045         "Before starting, change the DS root in the EXCEL spreadsheet to match " & _
»
00046         "your DS." & CRLF & CRLF & _
00047         "Type in the path of a file containing users to add or delete" & CRLF & CRLF
» & _
00048         "Sample Add User file: ADDUSERS.XLS" & CRLF & _
00049         "Sample Delete User file: DELUSERS.XLS" & CRLF)
00050     'Else file containing users is the first argument
00051 Else
00052     TextXL = oArgs.item(0)
00053 End If
00054
00055 If TextXL = "" Then
00056     WScript.Echo "No input file provided. Stopping the script now."
00057     WScript.Quit(1)
00058 End If
00059
00060 'We will use ou to control loop, so set initial value to null
00061 ou = ""
00062
00063 'Start EXCEL and display it to the user
00064 Set oXL = WScript.CreateObject("EXCEL.application")
00065 oXL.Visible = True
00066
00067 'Open the workbook passed in the command line
00068 oXL.workbooks.open TextXL
00069
00070 'Activate the Add page
00071 oXL.sheets("Add").Activate
00072

```

```
00073 'Put the cursor in the starting cell and read the DS root
00074 oXL.ActiveSheet.range("A2").Activate ' this cell has the DS root in it
00075
00076 'Show it to the user
00077 'WScript.Echo oXL.activecell.Value
00078
00079 'This is the starting point in the ds
00080 root = oXL.activecell.Value
00081
00082 'Step to the next row
00083 oXL.activecell.offset(1, 0).Activate
00084
00085 'Until we run out of rows
00086 Do While oXL.activecell.Value <> ""
00087
00088     'if the requested OU is a new one...
00089     If oXL.activecell.Value <> ou Then
00090         'Pick up the OU name...
00091         ou = oXL.activecell.Value
00092
00093         'Compose the ADSI path...
00094         s = "LDAP://" + ou + "," + root
00095
00096         'Show it to the user...
00097         WScript.Echo s
00098
00099         'And get the object
00100         Set c = Getobject(s)
00101     End If
00102
00103     'Compose the user common name name from first and last names...
00104     » uname = "CN=" + oXL.activecell.offset(0, 1).Value + " " + oXL.activecell.offset(
00105     » 0, 2).Value
00106
00107     'Create the new user object...
00108     Set u = c.Create("user", uname)
00109
00110     'Set the properties of the new user
00111     u.Put "givenName", oXL.activecell.offset(0, 1).Value 'givenName
00112     u.Put "sn", oXL.activecell.offset(0, 2).Value 'sn
00113     u.Put "mail", oXL.activecell.offset(0, 3).Value 'Email
00114     u.Put "sAMAccountName", oXL.activecell.offset(0, 4).Value 'Sam Acct
00115     u.Put "telephoneNumber", oXL.activecell.offset(0, 5).Value 'Phone
00116
00117     'Enable the account, must change pw @ logon
00118     u.Put "userAccountControl",16
00119
00120     '...and update the DS
00121     u.SetInfo
00122
00123     'Done with this object, discard it
00124     Set u = Nothing
00125
00126     'Step to the next user...
00127     oXL.activecell.offset(1, 0).Activate 'Next row
00128 Loop
00129
00130 'Done. close excel spreadsheet
00131 oXL.application.quit
```

```

00001  '*****
00002  '
00003  '           © International Computers Limited 2000
00004  '
00005  '*****
00006  '
00007  ' Description
00008  '
00009  '     This batch calls ICL [COMBO] Auto Bank Tenders.
00010  '
00011  ' Authors (first name is last person to make any change)
00012  '
00013  '     J.Tuthill
00014  '     R.Parkinson
00015  '
00016  '*****
00017  '
00018  Option Explicit
00019
00020 Private Const DEBUG_Required = 0 '* 1/0 turns on/off message boxes
00021 Private Const DEBUG_StatusLog_Required = 0 '* 1/0 turns on/off tasklog debug
00022
00023 Private Const INIT_ErrorRerun = 0 '* 1/0 turns on/off rerun on error
00024 Private Const INIT_RunOnce = 0 '* 1/0 turns on/off rerun checking
00025
00026 Private Const CONF_SignalBatch = 0 ' 1/0 turns on/off batch signalling
00027
00028 Private Const ERROR_NoError = 0
00029 Private Const ERROR_Generic = -10
00030 Private Const ERROR_ObjectCreation = -20
00031
00032 Private Const TASKSTATUS_NotStarted = -2
00033 Private Const TASKSTATUS_Started = -1
00034 Private Const TASKSTATUS_Success = 0
00035 Private Const TASKSTATUS_Warnings = 1
00036 Private Const TASKSTATUS_Interrupted = 2
00037 Private Const TASKSTATUS_Error = 3
00038
00039 Private Const STORESTATUS_Open = 1
00040 Private Const STORESTATUS_Closing = 2
00041 Private Const STORESTATUS_Closed = 3
00042
00043 Private Const PATH_EventActivator =
00044 » "D:\Gstr\ApplicationServices\Tools\Bin\ICLSKEventActivator.exe"
00045
00046 Dim lngDebugCtr
00047 Dim strDebugName
00048 Dim strParms
00049
00050 '*****
00051 ' *
00052 ' *
00053 ' *
00054 '*****
00055 Sub Main
00056
00057 ' -- Ignore all errors.
00058 On Error Resume Next
00059
00060 Dim nTaskStatus ' The status of this particular task
00061 Dim nStoreStatus ' Store status (trading, not trading etc.)
00062 Dim nReturnCode ' General return code constant
00063
00064 Dim strScriptName ' The name of this particular script
00065 Dim nNewStoreStatus ' The new store status
00066 Dim pMachine ' Machine role
00067
00068 strScriptName = "Auto Bank"
00069 strDebugName = "Auto Bank (" & formatdatetime(Now, 4) & ") -"
00070 lngDebugCtr = 0
00071
00072 DEBUG("Start " & strScriptName)
00073

```

```

00074
00075
00076
00077
00078
00079
00080
00081
00082
00083
00084
00085
00086
00087
00088
00089
00090
00091
00092
00093
00094
00095
00096
00097
00098
00099
00100
00101
00102
00103
00104
00105
00106
00107
00108
00109
00110
00111
00112
00113
00114
00115
00116
00117
00118
00119
00120
00121
00122
00123
00124
00125
00126
00127
00128
00129
00130
00131
00132
00133
00134
00135
00136
00137
00138
00139
00140
00141
00142
00143
00144
00145
00146
00147
00148
00149
00150
00151

```

```

'*****
' * Initialize
'*****
nTaskStatus = Initialize (strScriptName)
DEBUG( "Initialize() returns nTaskStatus = " & nTaskStatus)
If (nTaskStatus = ERROR_Generic) Or (nTaskStatus = TASKSTATUS_Success) Then
  Exit Sub
End If

'*****
' * Log the beginning of the task.
'*****
setStatusLog strScriptName, TASKSTATUS_Started, 0, ""

'*****
' * Retrieve any parameters.
'*****
strParms = GetParms (strScriptName)
strParms = ""

'*****
' * Run ICL [COMBO] Auto Bank Tenders
'*****
nReturnCode = RunComboCOAutoBank (strScriptName, strParms)
If nReturnCode <> ERROR_NoError Then
  DEBUG("Error running ICL [COMBO] Auto Bank Tenders = " & nReturnCode)
  Exit Sub
End If

'*****
' * Kick off another batch (if configured)
'*****
If CONF_SignalBatch = 1 Then
  ' Note that by default we just pass the parameters we were given.
  nReturnCode = AsyncEvent (strScriptName, "Unspecified", strParms)
  If nReturnCode <> ERROR_NoError Then
    DEBUG("Error signalling Unspecified = " & nReturnCode)
    Exit Sub
  End If
End If

'*****
' * Log the end of the task.
'*****
setStatusLog strScriptName, TASKSTATUS_Success, 0, ""

'*****
' * Remove the message from the queue
'*****
RemoveMessage

DEBUG("ICL [COMBO] Auto Bank Tenders terminated OK...")

End Sub

'*****
' *
' * Initialize
' *
'*****
Function Initialize (strScriptName)

  '-- Ignore all errors.
  On Error Resume Next

  Dim oStatusLog
  Dim oTaskLog
  Dim strWarning
  Dim nStatus

  ' Set the initial value for the function. This may get overwritten.
  nStatus = TASKSTATUS_NotStarted

```

```

00152
00153 ' -- Retrieve a reference to StatusLog service.
00154 Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00155 If oStatusLog Is Nothing Then
00156     strWarning = "Error creating StatusLog object. Err(" & Err.Number & ") = " &
» Err.Description
00157     DEBUG(strWarning)
00158     Initialize = ERROR_Generic
00159     Exit Function
00160 End If
00161
00162 ' Note: the next blocks can be bypassed by setting INIT_<whatever> at the head
» of this file.
00163 ' These variables are initially set by the batch app wizard.
00164
00165 ' First block: check if we've been run before *with errors* and if so bomb out
» with an error message.
00166 '
00167 If INIT_ErrorRerun = 1 Then
00168
00169     ' -- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00170     Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00171     If Err <> 0 Then
00172         If Err.Number - vbObjectError <> 100 Then ' Ignore log record does not exit
00173
00174             strWarning = "Error getting Last StatusLog. Err(" & Err.Number & ") = " &
» Err.Description
00175             'msgbox strWarning
00176             Initialize = ERROR_Generic
00177             Exit Function
00178         End If
00179
00180         Err.Clear
00181         Set oTaskLog = Nothing
00182     End If
00183
00184     ' -- Retrieve the last StatusLog (TaskLog.Status).
00185     If Not oTaskLog Is Nothing Then
00186         nStatus = oTaskLog.Status
00187         If nStatus <> TASKSTATUS_Success Then
00188             strWarning = "Warning: '" & strScriptName & "' already run with errors."
00189             SetStatusLog strScriptName, oTaskLog.Status, ERROR_NoError, strWarning
00190             DEBUG(strWarning)
00191         End If
00192     Else
00193         nStatus = TASKSTATUS_NotStarted
00194     End If
00195     Set oTaskLog = Nothing
00196 End If
00197
00198 ' Second block: check if we've been run before and if so bomb out with an error
» message.
00199 '
00200 If INIT_RunOnce = 1 Then
00201
00202     ' -- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00203     Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00204     If Err <> 0 Then
00205         strWarning = "Error getting Last StatusLog. Err(" & Err.Number & ") = " &
» Err.Description
00206         DEBUG(strWarning)
00207         Initialize = ERROR_Generic
00208         Exit Function
00209     End If
00210
00211     ' -- Retrieve the last StatusLog (TaskLog.Status).
00212     If Not oTaskLog Is Nothing Then
00213         nStatus = oTaskLog.Status
00214         If nStatus = TASKSTATUS_Success Then
00215             strWarning = "Warning: '" & strScriptName & "' already run."
00216             SetStatusLog strScriptName, TASKSTATUS_Success, ERROR_NoError, strWarning
00217             DEBUG(strWarning)
00218         End If
00219     Else
00220         nStatus = TASKSTATUS_NotStarted
00221     End If
00222     Set oTaskLog = Nothing
00223 End If

```

```

00224
00225     Set oStatusLog = Nothing
00226
00227     ' Return what we've found.
00228
00229     Initialize = nStatus
00230
00231 End Function
00232
00233
00234
00235 ' *****
00236 ' *
00237 ' * Get the parameters
00238 ' *
00239 ' *****
00240 Function GetParms (strScriptName)
00241
00242     '-- Ignore all errors.
00243     On Error Resume Next
00244
00245     Dim strEventName ' Name of the associated event for this batch
00246     Dim objEventDetail ' The associated event tdetail for the above event
00247     Dim objParms ' Object containing parameters
00248     Dim nCount ' General count
00249     Dim nLoopCount ' Loop-control count
00250     Dim strParameters ' Parameters
00251     Dim strWarning ' Warning message
00252
00253     strEventName = GSBOEvent.IGSBOEvent_EventName
00254     If Err.Number <> 0 Then
00255         strWarning = "Getting Event Name : Err(" & Err.Number & ") = " &
    » Err.Description
00256     End If
00257
00258     '-- Create a reference to the EventDetail object.
00259     If strWarning = "" Then
00260         Set objEventDetail = GSBOEvent.IGSBOEvent_EventDetail
00261         If Err.Number <> 0 Then
00262             strWarning = "Getting Event Detail : Err(" & Err.Number & ") = " &
    » Err.Description
00263         End If
00264     End If
00265
00266     '-- Create a reference to the AdditionalParameters collection.
00267     If strWarning = "" Then
00268         Set objParms = objEventDetail.AdditionalParameters
00269         If Err.Number <> 0 Then
00270             strWarning = "Get Additional Parameters : Err(" & Err.Number & ") = " &
    » Err.Description
00271         End If
00272     End If
00273
00274     '-- Count the number of the elements in the collection.
00275     If strWarning = "" Then
00276         nCount = objParms.Count
00277         If Err.Number <> 0 Then
00278             strWarning = "Get parameter count : Err(" & Err.Number & ") = " &
    » Err.Description
00279         End If
00280     End If
00281
00282     '-- Retrieve the parameters.
00283     If strWarning = "" Then
00284         If nCount = 0 Then
00285             strWarning = "No parameters"
00286         Else
00287             strParameters = ""
00288             For nLoopCount = 1 to nCount
00289                 strParameters = strParameters & "," & objParms.Item(1)
00290                 If Err.Number <> 0 Then
00291                     strWarning = "Error Getting Parameter " & nLoopCount & " : Err(" &
    » Err.Number & ") = " & Err.Description
00292                 End If
00293             Next
00294         End If
00295     End If
00296

```

```
00297     '-- Log the error.
00298     If strWarning <> "" Then
00299         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00300         DEBUG (strWarning)
00301         strParameters = ""
00302     Else
00303         DEBUG ("Parameters = " & strParameters)
00304     End If
00305
00306     GetParms = strParameters
00307
00308 End Function
00309
00310
00311
00312
00313     ' *****
00314     ' *
00315     ' * Update "TaskLog" table
00316     ' *
00317     ' *****
00318 Sub SetStatusLog (strTaskID, nStatus, nResultCode, strNote)
00319
00320     '-- Ignore all errors.
00321     On Error Resume Next
00322
00323     Dim oStatusLog ' Status log object
00324     Dim strMsg ' The message to log
00325
00326     DEBUG("Logging Status : " & strNote)
00327
00328     '-- Retrieve a reference to StatusLog service.
00329     Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00330     If oStatusLog Is Nothing Then
00331         strMsg = "Error creating StatusLog object"
00332         DEBUG (strMsg)
00333         Exit Sub
00334     End If
00335
00336     '-- Insert/Update "TaskLog" table.
00337     If strNote = "" Then
00338         oStatusLog.Log strTaskID, nStatus, nResultCode
00339     Else
00340         oStatusLog.Log strTaskID, nStatus, nResultCode, , , strNote
00341     End If
00342
00343     Set oStatusLog = Nothing
00344
00345 End Sub
00346
00347
00348
00349     ' *****
00350     ' *
00351     ' * Run the Model Batch Program
00352     ' *
00353     ' *****
00354 Function RunComboCOAutoBank (strScriptName, strParameters)
00355
00356     On Error Resume Next
00357
00358     Dim oStatusLog ' Status Log Object
00359     Dim strMsg ' General messages
00360     Dim oModel ' Model Batch Application
00361     Dim nStatus ' Status code
00362     Dim nCopies
00363     Dim strWarning ' Warning message
00364     Dim lngStatus ' Status code (long)
00365
00366     nStatus = ERROR_NoError
00367
00368     DEBUG("Running ICL [COMBO] Auto Bank Tenders")
00369
00370     ' *****
00371     ' * Create object
00372     ' *****
00373     If (nStatus = ERROR_NoError) Then
00374         DEBUG("Creating ICL [COMBO] Auto Bank Tenders")
```

```

00375     Set oModel = Createobject("ComboCOAutoBank.Application")
00376     If oModel Is Nothing Then
00377         DEBUG("Error creating ICL [COMBO] Auto Bank Tenders")
00378         strWarning = "Error creating ComboCOAutoBank.Application object"
00379         nStatus = ERROR_ObjectCreation
00380     Else
00381         DEBUG("Created ICL [COMBO] Auto Bank Tenders OK")
00382     End If
00383 End If
00384
00385     '*****
00386     '* DO whatever it does, passing the parameter.
00387     '*****
00388     If (nStatus = ERROR_NoError) Then
00389         DEBUG ("Running ICL [COMBO] Auto Bank Tenders")
00390         Err.Clear
00391         nStatus = oModel.PerformTask (strParameters)
00392         If Err <> 0 Then
00393             strWarning = Err.Description
00394             nStatus = ERROR_Generic
00395         End If
00396     End If
00397
00398
00399     '*****
00400     '* Log error
00401     '*****
00402     If (nStatus <> ERROR_NoError) And (strWarning <> "") Then
00403         SetStatusLog strScriptName, TASKSTATUS_Error, nStatus, strWarning
00404     End If
00405
00406     RunComboCOAutoBank = nStatus
00407
00408 End Function
00409
00410
00411
00412     '*****
00413     '*
00414     '* Async Event
00415     '*
00416     '*****
00417 Function AsyncEvent (strScriptName, strEventName, strParameters)
00418
00419     '-- Ignore all errors.
00420     On Error Resume Next
00421
00422     Dim objLauncher
00423     Dim strWarning
00424     Dim lngStatus
00425
00426     DEBUG("About to Fire Asynchronous Event : " & strEventName)
00427
00428
00429     '*****
00430     ' Create the launcher event
00431     '*****
00432     Set objLauncher = Createobject("ICLSKLaunch.LaunchEXE")
00433     If err.number <> 0 Then
00434         strWarning = "Error Creating Launcher Object - Err(" & Err.Number & ") = " &
» Err.Description
00435         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00436         DEBUG(strWarning)
00437         AsyncEvent = ERROR_ObjectCreation
00438         Exit Function
00439     End If
00440
00441     '*****
00442     ' Set up parameters and run the exe !
00443     '*****
00444     objLauncher.ApplicationName = PATH_EventActivator
00445     objLauncher.ApplicationArgs = strEventName & " " & strParameters
00446
00447     objLauncher.Execute
00448     If err.number <> 0 Then
00449         strWarning = "Error Starting Event Activator. Err(" & Err.Number & ") = " &
» Err.Description
00450         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning

```

```
00451     DEBUG(strWarning)
00452     AsyncEvent = ERROR_ObjectCreation
00453     Exit Function
00454 End If
00455
00456     Set objLauncher = Nothing
00457
00458     AsyncEvent = ERROR_NoError
00459
00460 End Function
00461
00462
00463
00464
00465 ' *****
00466 ' *
00467 ' * DEBUG messages (if required)
00468 ' *
00469 ' *****
00470 Sub DEBUG (strMessage)
00471
00472     If (DEBUG_Required = 1) Then
00473         MsgBox strMessage
00474     End If
00475
00476     If (DEBUG_StatusLog_Required = 1) Then
00477         lngDebugCtr = lngDebugCtr + 1
00478         SetStatusLog strDebugName & lngDebugCtr, TASKSTATUS_Success, 0, strMessage
00479     End If
00480
00481
00482 End Sub
00483
00484
00485
```

```

00001  '*****
00002  '
00003  '           © International Computers Limited 1999
00004  '
00005  '*****
00006  '
00007  ' Description
00008  '
00009  '     This batch calls ICL [COMBO] Auto Lift Tenders.
00010  '
00011  ' Authors (first name is last person to make any change)
00012  '
00013  '     R.Parkinson
00014  '
00015  '*****
00016  '
00017  Option Explicit
00018  '
00019  Private Const DEBUG_Required = 0 '* 1/0 turns on/off message boxes
00020  Private Const DEBUG_StatusLog_Required = 0 '* 1/0 turns on/off tasklog debug
00021  '
00022  Private Const INIT_ErrorRerun = 0 '* 1/0 turns on/off rerun on error
00023  Private Const INIT_RunOnce = 0 '* 1/0 turns on/off rerun checking
00024  '
00025  Private Const CONF_SignalBatch = 0 ' 1/0 turns on/off batch signalling
00026  '
00027  Private Const ERROR_NoError = 0
00028  Private Const ERROR_Generic = -10
00029  Private Const ERROR_ObjectCreation = -20
00030  '
00031  Private Const TASKSTATUS_NotStarted = -2
00032  Private Const TASKSTATUS_Started = -1
00033  Private Const TASKSTATUS_Success = 0
00034  Private Const TASKSTATUS_Warnings = 1
00035  Private Const TASKSTATUS_Interrrupted = 2
00036  Private Const TASKSTATUS_Error = 3
00037  '
00038  Private Const STORESTATUS_Open = 1
00039  Private Const STORESTATUS_Closing = 2
00040  Private Const STORESTATUS_Closed = 3
00041  '
00042  Private Const PATH_EventActivator =
00043  » "D:\Gstr\ApplicationServices\Tools\Bin\ICLSKEventActivator.exe"
00044  '
00045  Dim lngDebugCtr
00046  Dim strDebugName
00047  Dim strParms
00048  '
00049  ' *****
00050  ' *
00051  ' *
00052  ' *
00053  ' *****
00054  Sub Main
00055  '
00056  ' -- Ignore all errors.
00057  On Error Resume Next
00058  '
00059  Dim nTaskStatus ' The status of this particular task
00060  Dim nStoreStatus ' Store status (trading, not trading etc.)
00061  Dim nReturnCode ' General return code constant
00062  '
00063  Dim strScriptName ' The name of this particular script
00064  Dim nNewStoreStatus ' The new store status
00065  Dim pMachine ' Machine role
00066  '
00067  strScriptName = "Auto Lift"
00068  strDebugName = "Auto Lift (" & formatdatetime(Now, 4) & ") -"
00069  lngDebugCtr = 0
00070  '
00071  DEBUG("Start " & strScriptName)
00072  '
00073  '

```

```

00074 '*****
00075 '* Initialize
00076 '*****
00077 nTaskStatus = Initialize (strScriptName)
00078 DEBUG( "Initialize() returns nTaskStatus = " & nTaskStatus)
00079 If (nTaskStatus = ERROR_Generic) Or (nTaskStatus = TASKSTATUS_Success) Then
00080     Exit Sub
00081 End If
00082
00083
00084 '*****
00085 '* Log the beginning of the task.
00086 '*****
00087 SetStatusLog strScriptName, TASKSTATUS_Started, 0, ""
00088
00089
00090 '*****
00091 '* Retrieve any parameters.
00092 '*****
00093 ' strParms = GetParms (strScriptName)
00094 strParms = ""
00095
00096
00097 '* Run ICL [COMBO] Auto Lift Tenders
00098 '*****
00099 nReturnCode = RunComboCOAutoLift (strScriptName, strParms)
00100 If nReturnCode <> ERROR_NoError Then
00101     DEBUG("Error running ICL [COMBO] Auto Lift Tenders = " & nReturnCode)
00102     Exit Sub
00103 End If
00104
00105
00106 '*****
00107 '* Kick off another batch (if configured)
00108 '*****
00109 If CONF_SignalBatch = 1 Then
00110     ' Note that by default we just pass the parameters we were given.
00111     nReturnCode = AsyncEvent (strScriptName, "Unspecified", strParms)
00112     If nReturnCode <> ERROR_NoError Then
00113         DEBUG("Error signalling Unspecified = " & nReturnCode)
00114         Exit Sub
00115     End If
00116 End If
00117
00118 '*****
00119 '* Log the end of the task.
00120 '*****
00121 SetStatusLog strScriptName, TASKSTATUS_Success, 0, ""
00122
00123 '*****
00124 '* Remove the message from the queue
00125 '*****
00126 RemoveMessage
00127
00128 DEBUG("ICL [COMBO] Auto Lift Tenders terminated OK...")
00129 End Sub
00130
00131
00132
00133
00134 '*****
00135 '*
00136 '* Initialize
00137 '*
00138 '*****
00139 Function Initialize (strScriptName)
00140
00141     '-- Ignore all errors.
00142     On Error Resume Next
00143
00144     Dim oStatusLog
00145     Dim oTaskLog
00146     Dim strWarning
00147     Dim nStatus
00148
00149     ' Set the initial value for the function. This may get overwritten.
00150     nStatus = TASKSTATUS_NotStarted
00151

```

```

00152     '-- Retrieve a reference to StatusLog service.
00153     Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00154     If oStatusLog Is Nothing Then
00155         strWarning = "Error creating StatusLog object. Err(" & Err.Number & ") = " &
» Err.Description
00156         DEBUG(strWarning)
00157         Initialize = ERROR_Generic
00158         Exit Function
00159     End If
00160
00161     ' Note: the next blocks can be bypassed by setting INIT_<whatever> at the head
» of this file.
00162     ' These variables are initially set by the batch app wizard.
00163
00164     ' First block: check if we've been run before *with errors* and if so bomb out
» with an error message.
00165     '
00166     If INIT_ErrorRerun = 1 Then
00167
00168         '-- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00169         Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00170         If Err <> 0 Then
00171             If Err.Number - vbObjectError <> 100 Then ' Ignore log record does not exit
00172
00173             strWarning = "Error getting Last StatusLog. Err(" & Err.Number & ") = " &
» Err.Description
00174                 'msgbox strWarning
00175                 Initialize = ERROR_Generic
00176                 Exit Function
00177             End If
00178
00179             Err.Clear
00180             Set oTaskLog = Nothing
00181         End If
00182
00183         '-- Retrieve the last StatusLog (TaskLog.Status).
00184         If Not oTaskLog Is Nothing Then
00185             nStatus = oTaskLog.Status
00186             If nStatus <> TASKSTATUS_Success Then
00187                 strWarning = "Warning: '" & strScriptName & "' already run with errors."
00188                 SetStatusLog strScriptName, oTaskLog.Status, ERROR_NoError, strWarning
00189                 DEBUG(strWarning)
00190             End If
00191         Else
00192             nStatus = TASKSTATUS_NotStarted
00193         End If
00194         Set oTaskLog = Nothing
00195     End If
00196
00197     ' Second block: check if we've been run before and if so bomb out with an error
» message.
00198     '
00199     If INIT_RunOnce = 1 Then
00200
00201         '-- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00202         Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00203         If Err <> 0 Then
00204             strWarning = "Error getting Last StatusLog. Err(" & Err.Number & ") = " &
» Err.Description
00205                 DEBUG(strWarning)
00206                 Initialize = ERROR_Generic
00207                 Exit Function
00208             End If
00209
00210             '-- Retrieve the last StatusLog (TaskLog.Status).
00211             If Not oTaskLog Is Nothing Then
00212                 nStatus = oTaskLog.Status
00213                 If nStatus = TASKSTATUS_Success Then
00214                     strWarning = "Warning: '" & strScriptName & "' already run."
00215                     SetStatusLog strScriptName, TASKSTATUS_Success, ERROR_NoError, strWarning
00216                     DEBUG(strWarning)
00217                 End If
00218             Else
00219                 nStatus = TASKSTATUS_NotStarted
00220             End If
00221             Set oTaskLog = Nothing
00222         End If
00223

```

```

00224     Set oStatusLog = Nothing
00225
00226     ' Return what we've found.
00227
00228     Initialize = nStatus
00229
00230 End Function
00231
00232
00233
00234 ' *****
00235 ' *
00236 ' * Get the parameters
00237 ' *
00238 ' *****
00239 Function GetParms (strScriptName)
00240
00241     '-- Ignore all errors.
00242     On Error Resume Next
00243
00244     Dim strEventName ' Name of the associated event for this batch
00245     Dim objEventDetail ' The associated event tdetail for the above event
00246     Dim objParms ' Object containing parameters
00247     Dim nCount ' General count
00248     Dim nLoopCount ' Loop-control count
00249     Dim strParameters ' Parameters
00250     Dim strWarning ' Warning message
00251
00252     strEventName = GSBOEvent.IGSBOEvent_EventName
00253     If Err.Number <> 0 Then
00254         strWarning = "Getting Event Name : Err(" & Err.Number & ") = " &
00255     Err.Description
00256     End If
00257
00258     '-- Create a reference to the EventDetail object.
00259     If strWarning = "" Then
00260         Set objEventDetail = GSBOEvent.IGSBOEvent_EventDetail
00261         If Err.Number <> 0 Then
00262             strWarning = "Getting Event Detail : Err(" & Err.Number & ") = " &
00263         Err.Description
00264         End If
00265     End If
00266
00267     '-- Create a reference to the AdditionalParameters collection.
00268     If strWarning = "" Then
00269         Set objParms = objEventDetail.AdditionalParameters
00270         If Err.Number <> 0 Then
00271             strWarning = "Get Additional Parameters : Err(" & Err.Number & ") = " &
00272         Err.Description
00273         End If
00274     End If
00275
00276     '-- Count the number of the elements in the collection.
00277     If strWarning = "" Then
00278         nCount = objParms.Count
00279         If Err.Number <> 0 Then
00280             strWarning = "Get parameter count : Err(" & Err.Number & ") = " &
00281         Err.Description
00282         End If
00283     End If
00284
00285     '-- Retrieve the parameters.
00286     If strWarning = "" Then
00287         If nCount = 0 Then
00288             strWarning = "No parameters"
00289         Else
00290             strParameters = ""
00291             For nLoopCount = 1 to nCount
00292                 strParameters = strParameters & "," & objParms.Item(1)
00293                 If Err.Number <> 0 Then
00294                     strWarning = "Error Getting Parameter " & nLoopCount & " : Err(" &
00295                 Err.Number & ") = " & Err.Description
00296                 End If
00297             Next
00298         End If
00299     End If
00300
00301     '-- Log the error.

```

```

00297     If strWarning <> "" Then
00298         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00299         DEBUG (strWarning)
00300         strParameters = ""
00301     Else
00302         DEBUG ("Parameters = " & strParameters)
00303     End If
00304
00305     GetParms = strParameters
00306 End Function
00307
00308
00309
00310
00311
00312 ' *****
00313 ' *
00314 ' * Update "TaskLog" table
00315 ' *
00316 ' *****
00317 Sub SetStatusLog (strTaskID, nStatus, nResultCode, strNote)
00318
00319     '-- Ignore all errors.
00320     On Error Resume Next
00321
00322     Dim oStatusLog ' Status log object
00323     Dim strMsg ' The message to log
00324
00325     DEBUG("Logging Status : " & strNote)
00326
00327     '-- Retrieve a reference to StatusLog service.
00328     Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00329     If oStatusLog Is Nothing Then
00330         strMsg = "Error creating StatusLog object"
00331         DEBUG (strMsg)
00332         Exit Sub
00333     End If
00334
00335     '-- Insert/Update "TaskLog" table.
00336     If strNote = "" Then
00337         oStatusLog.Log strTaskID, nStatus, nResultCode
00338     Else
00339         oStatusLog.Log strTaskID, nStatus, nResultCode, , , strNote
00340     End If
00341
00342     Set oStatusLog = Nothing
00343 End Sub
00344
00345
00346
00347
00348 ' *****
00349 ' *
00350 ' * Run the Model Batch Program
00351 ' *
00352 ' *****
00353 Function RunComboCOAutoLift (strScriptName, strParameters)
00354
00355     On Error Resume Next
00356
00357     Dim oStatusLog ' Status Log Object
00358     Dim strMsg ' General messages
00359     Dim oModel ' Model Batch Application
00360     Dim nStatus ' Status code
00361     Dim nCopies
00362     Dim strWarning ' Warning message
00363     Dim lngStatus ' Status code (long)
00364
00365     nStatus = ERROR_NoError
00366
00367     DEBUG("Running ICL [COMBO] Auto Lift Tenders")
00368
00369     ' *****
00370     ' * Create object
00371     ' *****
00372     If (nStatus = ERROR_NoError) Then
00373         DEBUG("Creating ICL [COMBO] Auto Lift Tenders")
00374         Set oModel = Createobject("ComboCOAutoLift.Application")

```

```

00375     If oModel Is Nothing Then
00376         DEBUG("Error creating ICL [COMBO] Auto Lift Tenders")
00377         strWarning = "Error creating ComboCOAutoLift.Application object"
00378         nStatus = ERROR_ObjectCreation
00379     Else
00380         DEBUG("Created ICL [COMBO] Auto Lift Tenders OK")
00381     End If
00382 End If
00383
00384     '*****
00385     '* DO whatever it does, passing the parameter.
00386     '*****
00387     If (nStatus = ERROR_NoError) Then
00388         DEBUG ("Running ICL [COMBO] Auto Lift Tenders")
00389         Err.Clear
00390         nStatus = oModel.PerformTask (strParameters)
00391         If Err <> 0 Then
00392             strWarning = Err.Description
00393             nStatus = ERROR_Generic
00394         End If
00395     End If
00396
00397     '*****
00398     '* Log error
00399     '*****
00400     If (nStatus <> ERROR_NoError) And (strWarning <> "") Then
00401         SetStatusLog strScriptName, TASKSTATUS_Error, nStatus, strWarning
00402     End If
00403
00404     RunComboCOAutoLift = nStatus
00405
00406 End Function
00407
00408
00409
00410
00411 ' *****
00412 ' *
00413 ' * Async Event
00414 ' *
00415 ' *****
00416 Function AsyncEvent (strScriptName, strEventName, strParameters)
00417
00418     '-- Ignore all errors.
00419     On Error Resume Next
00420
00421     Dim objLauncher
00422     Dim strWarning
00423     Dim lngStatus
00424
00425     DEBUG("About to Fire Asynchronous Event : " & strEventName)
00426
00427     '*****
00428     ' Create the launcher event
00429     '*****
00430     Set objLauncher = Createobject("ICLSKLaunch.LaunchEXE")
00431     If err.number <> 0 Then
00432         strWarning = "Error Creating Launcher Object - Err(" & Err.Number & ") = " &
00433     » Err.Description
00434         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00435         DEBUG(strWarning)
00436         AsyncEvent = ERROR_ObjectCreation
00437         Exit Function
00438     End If
00439
00440     '*****
00441     ' Set up parameters and run the exe !
00442     '*****
00443     objLauncher.ApplicationName = PATH_EventActivator
00444     objLauncher.ApplicationArgs = strEventName & " " & strParameters
00445
00446     objLauncher.Execute
00447     If err.number <> 0 Then
00448         strWarning = "Error Starting Event Activator. Err(" & Err.Number & ") = " &
00449     » Err.Description
00449         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00450         DEBUG(strWarning)

```

```
00451     AsyncEvent = ERROR_ObjectCreation
00452     Exit Function
00453 End If
00454
00455     Set objLauncher = Nothing
00456
00457     AsyncEvent = ERROR_NoError
00458
00459 End Function
00460
00461
00462
00463 '*****
00464 ' *
00465 ' * Remove the message from the queue
00466 ' *
00467 '*****
00468 Function RemoveMessage
00469
00470     Dim iQueueInfo
00471     Dim iQueue
00472     Dim iMessageDest
00473
00474     Set iQueueInfo = Createobject("MSMQ.MSMQQueueInfo")
00475     iQueueInfo.PathName = ".\GSBOEventTargetQueue1"
00476     Set iQueue = iQueueInfo.Open(1,0)
00477
00478     Set iMessageDest = iQueue.Receive(,,1000)
00479
00480     If iMessageDest Is Nothing Then
00481         DEBUG ("There are no messages in the queue.")
00482     Else
00483         DEBUG ("The first message was removed from the queue.")
00484     End If
00485
00486 End Function
00487
00488
00489
00490 ' *****
00491 ' *
00492 ' * DEBUG messages (if required)
00493 ' *
00494 ' *****
00495 Sub DEBUG (strMessage)
00496
00497     If (DEBUG_Required = 1) Then
00498         MsgBox strMessage
00499     End If
00500
00501     If (DEBUG_StatusLog_Required = 1) Then
00502         lngDebugCtr = lngDebugCtr + 1
00503         SetStatusLog strDebugName & lngDebugCtr, TASKSTATUS_Success, 0, strMessage
00504     End If
00505
00506
00507 End Sub
00508
00509
00510
```

```
00001  '*****
00002  '
00003  '           © International Computers Limited 2000
00004  '
00005  '*****
00006  '
00007  ' Description
00008  '
00009  '     This batch calls ICL [COMBO] Auto Pay Tenders.
00010  '
00011  ' Authors (first name is last person to make any change)
00012  '
00013  '     J.D.Welch
00014  '     R.Parkinson
00015  '
00016  '*****
00017  » *****
00018  Option Explicit
00019
00020  Private Const DEBUG_Required = 0 '* 1/0 turns on/off message boxes
00021  Private Const DEBUG_StatusLog_Required = 0 '* 1/0 turns on/off tasklog debug
00022
00023  Private Const INIT_ErrorRerun = 0 '* 1/0 turns on/off rerun on error
00024  Private Const INIT_RunOnce = 0 '* 1/0 turns on/off rerun checking
00025
00026  Private Const CONF_SignalBatch = 0 ' 1/0 turns on/off batch signalling
00027
00028  Private Const ERROR_NoError = 0
00029  Private Const ERROR_Generic = -10
00030  Private Const ERROR_ObjectCreation = -20
00031
00032  Private Const TASKSTATUS_NotStarted = -2
00033  Private Const TASKSTATUS_Started = -1
00034  Private Const TASKSTATUS_Success = 0
00035  Private Const TASKSTATUS_Warnings = 1
00036  Private Const TASKSTATUS_Interrupted = 2
00037  Private Const TASKSTATUS_Error = 3
00038
00039  Private Const STORESTATUS_Open = 1
00040  Private Const STORESTATUS_Closing = 2
00041  Private Const STORESTATUS_Closed = 3
00042
00043  Private Const PATH_EventActivator =
00044  » "D:\Gstr\ApplicationServices\Tools\Bin\ICLSKEventActivator.exe"
00045
00046  Dim lngDebugCtr
00047  Dim strDebugName
00048  Dim strParms
00049
00050  ' *****
00051  ' *
00052  ' *
00053  ' *
00054  ' *****
00055  Sub Main
00056
00057  ' -- Ignore all errors.
00058  On Error Resume Next
00059
00060  Dim nTaskStatus ' The status of this particular task
00061  Dim nStoreStatus ' Store status (trading, not trading etc.)
00062  Dim nReturnCode ' General return code constant
00063
00064  Dim strScriptName ' The name of this particular script
00065  Dim nNewStoreStatus ' The new store status
00066  Dim pMachine ' Machine role
00067
00068  strScriptName = "Auto Pay"
00069  strDebugName = "Auto Pay (" & formatdatetime(Now, 4) & ") -"
00070  lngDebugCtr = 0
00071
00072  DEBUG("Start " & strScriptName)
00073
```

```

00074
00075
00076
00077
00078
00079
00080
00081
00082
00083
00084
00085
00086
00087
00088
00089
00090
00091
00092
00093
00094
00095
00096
00097
00098
00099
00100
00101
00102
00103
00104
00105
00106
00107
00108
00109
00110
00111
00112
00113
00114
00115
00116
00117
00118
00119
00120
00121
00122
00123
00124
00125
00126
00127
00128
00129
00130
00131
00132
00133
00134
00135
00136
00137
00138
00139
00140
00141
00142
00143
00144
00145
00146
00147
00148
00149
00150
00151

```

```

'*****
' * Initialize
'*****
nTaskStatus = Initialize (strScriptName)
DEBUG( "Initialize() returns nTaskStatus = " & nTaskStatus)
If (nTaskStatus = ERROR_Generic) Or (nTaskStatus = TASKSTATUS_Success) Then
  Exit Sub
End If

'*****
' * Log the beginning of the task.
'*****
setStatusLog strScriptName, TASKSTATUS_Started, 0, ""

'*****
' * Retrieve any parameters.
'*****
strParms = GetParms (strScriptName)
strParms = ""

'*****
' * Run ICL [COMBO] Auto Pay Tenders
'*****
nReturnCode = RunComboCOAutoPay (strScriptName, strParms)
If nReturnCode <> ERROR_NoError Then
  DEBUG("Error running ICL [COMBO] Auto Pay Tenders = " & nReturnCode)
  Exit Sub
End If

'*****
' * Kick off another batch (if configured)
'*****
If CONF_SignalBatch = 1 Then
  ' Note that by default we just pass the parameters we were given.
  nReturnCode = AsyncEvent (strScriptName, "Unspecified", strParms)
  If nReturnCode <> ERROR_NoError Then
    DEBUG("Error signalling Unspecified = " & nReturnCode)
    Exit Sub
  End If
End If

'*****
' * Log the end of the task.
'*****
setStatusLog strScriptName, TASKSTATUS_Success, 0, ""

'*****
' * Remove the message from the queue
'*****
RemoveMessage

DEBUG("ICL [COMBO] Auto Pay Tenders terminated OK..")
End Sub

'*****
' *
' * Initialize
' *
'*****
Function Initialize (strScriptName)
  '-- Ignore all errors.
  On Error Resume Next

  Dim oStatusLog
  Dim oTaskLog
  Dim strWarning
  Dim nStatus

  ' Set the initial value for the function. This may get overwritten.
  nStatus = TASKSTATUS_NotStarted

```

```

00152 1
00153      '-- Retrieve a reference to StatusLog service.
00154      Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00155      If oStatusLog Is Nothing Then
00156          strWarning = "Error creating StatusLog object. Err(" & Err.Number & ") = " &
» Err.Description
00157          DEBUG(strWarning)
00158          Initialize = ERROR_Generic
00159          Exit Function
00160      End If
00161
00162      ' Note: the next blocks can be bypassed by setting INIT_<whatever> at the head
» of this file.
00163      ' These variables are initially set by the batch app wizard.
00164
00165      ' First block: check if we've been run before *with errors* and if so bomb out
» with an error message.
00166      '
00167      If INIT_ErrorRerun = 1 Then
00168
00169          '-- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00170          Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00171          If Err <> 0 Then
00172              If Err.Number - vbObjectError <> 100 Then ' Ignore log record does not exit
00173
00174          » Err.Description
00175              'msgbox strWarning
00176              Initialize = ERROR_Generic
00177              Exit Function
00178          End If
00179
00180              Err.Clear
00181              Set oTaskLog = Nothing
00182          End If
00183
00184          '-- Retrieve the last StatusLog (TaskLog.Status).
00185          If Not oTaskLog Is Nothing Then
00186              nStatus = oTaskLog.Status
00187              If nStatus <> TASKSTATUS_Success Then
00188                  strWarning = "Warning: '" & strScriptName & "' already run with errors."
00189                  SetStatusLog strScriptName, oTaskLog.Status, ERROR_NoError, strWarning
00190                  DEBUG(strWarning)
00191              End If
00192          Else
00193              nStatus = TASKSTATUS_NotStarted
00194          End If
00195          Set oTaskLog = Nothing
00196      End If
00197
00198      ' Second block: check if we've been run before and if so bomb out with an error
» message.
00199      '
00200      If INIT_RunOnce = 1 Then
00201
00202          '-- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00203          Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00204          If Err <> 0 Then
00205          » Err.Description
00206              strWarning = "Error getting Last StatusLog. Err(" & Err.Number & ") = " &
00207              DEBUG(strWarning)
00208              Initialize = ERROR_Generic
00209              Exit Function
00210          End If
00211
00212          '-- Retrieve the last StatusLog (TaskLog.Status).
00213          If Not oTaskLog Is Nothing Then
00214              nStatus = oTaskLog.Status
00215              If nStatus = TASKSTATUS_Success Then
00216                  strWarning = "Warning: '" & strScriptName & "' already run."
00217                  SetStatusLog strScriptName, TASKSTATUS_Success, ERROR_NoError, strWarning
00218                  DEBUG(strWarning)
00219              End If
00220          Else
00221              nStatus = TASKSTATUS_NotStarted
00222          End If
00223          Set oTaskLog = Nothing
00224      End If

```

```
00224
00225     Set oStatusLog = Nothing
00226
00227     ' Return what we've found.
00228
00229     Initialize = nStatus
00230
00231 End Function
00232
00233
00234
00235 ' *****
00236 ' *
00237 ' * Get the parameters
00238 ' *
00239 ' *****
00240 Function GetParms (strScriptName)
00241
00242     '-- Ignore all errors.
00243     On Error Resume Next
00244
00245     Dim strEventName ' Name of the associated event for this batch
00246     Dim objEventDetail ' The associated event tdetail for the above event
00247     Dim objParms ' Object containing parameters
00248     Dim nCount ' General count
00249     Dim nLoopCount ' Loop-control count
00250     Dim strParameters ' Parameters
00251     Dim strWarning ' Warning message
00252
00253     strEventName = GSBOEvent.IGSBOEvent_EventName
00254     If Err.Number <> 0 Then
00255         strWarning = "Getting Event Name : Err(" & Err.Number & ") = " &
00256     Err.Description
00257     End If
00258
00259     '-- Create a reference to the EventDetail object.
00260     If strWarning = "" Then
00261         Set objEventDetail = GSBOEvent.IGSBOEvent_EventDetail
00262         If Err.Number <> 0 Then
00263             strWarning = "Getting Event Detail : Err(" & Err.Number & ") = " &
00264         Err.Description
00265         End If
00266     End If
00267
00268     '-- Create a reference to the AdditionalParameters collection.
00269     If strWarning = "" Then
00270         Set objParms = objEventDetail.AdditionalParameters
00271         If Err.Number <> 0 Then
00272             strWarning = "Get Additional Parameters : Err(" & Err.Number & ") = " &
00273         Err.Description
00274         End If
00275     End If
00276
00277     '-- Count the number of the elements in the collection.
00278     If strWarning = "" Then
00279         nCount = objParms.Count
00280         If Err.Number <> 0 Then
00281             strWarning = "Get parameter count : Err(" & Err.Number & ") = " &
00282         Err.Description
00283         End If
00284     End If
00285
00286     '-- Retrieve the parameters.
00287     If strWarning = "" Then
00288         If nCount = 0 Then
00289             strWarning = "No parameters"
00290         Else
00291             strParameters = ""
00292             For nLoopCount = 1 to nCount
00293                 strParameters = strParameters & "," & objParms.Item(1)
00294                 If Err.Number <> 0 Then
00295                     strWarning = "Error Getting Parameter " & nLoopCount & " : Err(" &
00296                 Err.Number & ") = " & Err.Description
00297                 End If
00298             Next
00299         End If
00300     End If
00301 End Function
```

```
00297     '-- Log the error.
00298     If strWarning <> "" Then
00299         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00300         DEBUG (strWarning)
00301         strParameters = ""
00302     Else
00303         DEBUG ("Parameters = " & strParameters)
00304     End If
00305
00306     GetParms = strParameters
00307
00308 End Function
00309
00310
00311
00312
00313     ' *****
00314     ' *
00315     ' * Update "TaskLog" table
00316     ' *
00317     ' *****
00318 Sub SetStatusLog (strTaskID, nStatus, nResultCode, strNote)
00319
00320     '-- Ignore all errors.
00321     On Error Resume Next
00322
00323     Dim oStatusLog ' Status log object
00324     Dim strMsg ' The message to log
00325
00326     DEBUG("Logging Status : " & strNote)
00327
00328     '-- Retrieve a reference to StatusLog service.
00329     Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00330     If oStatusLog Is Nothing Then
00331         strMsg = "Error creating StatusLog object"
00332         DEBUG (strMsg)
00333         Exit Sub
00334     End If
00335
00336     '-- Insert/Update "TaskLog" table.
00337     If strNote = "" Then
00338         oStatusLog.Log strTaskID, nStatus, nResultCode
00339     Else
00340         oStatusLog.Log strTaskID, nStatus, nResultCode, , , strNote
00341     End If
00342
00343     Set oStatusLog = Nothing
00344
00345 End Sub
00346
00347
00348
00349     ' *****
00350     ' *
00351     ' * Run the Model Batch Program
00352     ' *
00353     ' *****
00354 Function RunComboCOAutoPay (strScriptName, strParameters)
00355
00356     On Error Resume Next
00357
00358     Dim oStatusLog ' Status Log Object
00359     Dim strMsg ' General messages
00360     Dim oModel ' Model Batch Application
00361     Dim nStatus ' Status code
00362     Dim nCopies
00363     Dim strWarning ' Warning message
00364     Dim lngStatus ' Status code (long)
00365
00366     nStatus = ERROR_NoError
00367
00368     DEBUG("Running ICL [COMBO] Auto Pay Tenders")
00369
00370     ' *****
00371     ' * Create object
00372     ' *****
00373     If (nStatus = ERROR_NoError) Then
00374         DEBUG("Creating ICL [COMBO] Auto Pay Tenders")

```

```

00375     Set oModel = Createobject("ComboCOAutoPay.Application")
00376     If oModel Is Nothing Then
00377         DEBUG("Error creating ICL [COMBO] Auto Pay Tenders")
00378         strWarning = "Error creating ComboCOAutoPay.Application object"
00379         nStatus = ERROR_ObjectCreation
00380     Else
00381         DEBUG("Created ICL [COMBO] Auto Pay Tenders OK")
00382     End If
00383 End If
00384
00385     '*****
00386     '* DO whatever it does, passing the parameter.
00387     '*****
00388     If (nStatus = ERROR_NoError) Then
00389         DEBUG ("Running ICL [COMBO] Auto Pay Tenders")
00390         Err.Clear
00391         nStatus = oModel.PerformTask (strParameters)
00392         If Err <> 0 Then
00393             strWarning = Err.Description
00394             nStatus = ERROR_Generic
00395         End If
00396     End If
00397
00398
00399     '*****
00400     '* Log error
00401     '*****
00402     If (nStatus <> ERROR_NoError) And (strWarning <> "") Then
00403         SetStatusLog strScriptName, TASKSTATUS_Error, nStatus, strWarning
00404     End If
00405
00406     RunComboCOAutoPay = nStatus
00407
00408 End Function
00409
00410
00411
00412     '*****
00413     '*
00414     '* Async Event
00415     '*
00416     '*****
00417     Function AsyncEvent (strScriptName, strEventName, strParameters)
00418
00419         '-- Ignore all errors.
00420         On Error Resume Next
00421
00422         Dim objLauncher
00423         Dim strWarning
00424         Dim lngStatus
00425
00426         DEBUG("About to Fire Asynchronous Event : " & strEventName)
00427
00428
00429         '*****
00430         '* Create the launcher event
00431         '*****
00432         Set objLauncher = Createobject("ICLSKLaunch.LaunchEXE")
00433         If err.number <> 0 Then
00434             strWarning = "Error Creating Launcher Object - Err(" & Err.Number & ") = " &
» Err.Description
00435             SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00436             DEBUG(strWarning)
00437             AsyncEvent = ERROR_ObjectCreation
00438             Exit Function
00439         End If
00440
00441         '*****
00442         '* Set up parameters and run the exe !
00443         '*****
00444         objLauncher.ApplicationName = PATH_EventActivator
00445         objLauncher.ApplicationArgs = strEventName & " " & strParameters
00446
00447         objLauncher.Execute
00448         If err.number <> 0 Then
00449             strWarning = "Error Starting Event Activator. Err(" & Err.Number & ") = " &
» Err.Description
00450             SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning

```

```
00451     DEBUG(strWarning)
00452     AsyncEvent = ERROR_ObjectCreation
00453     Exit Function
00454 End If
00455
00456     Set objLauncher = Nothing
00457
00458     AsyncEvent = ERROR_NoError
00459
00460 End Function
00461
00462
00463
00464
00465 ' *****
00466 ' *
00467 ' * DEBUG messages (if required)
00468 ' *
00469 ' *****
00470 Sub DEBUG (strMessage)
00471
00472     If (DEBUG_Required = 1) Then
00473         MsgBox strMessage
00474     End If
00475
00476     If (DEBUG_StatusLog_Required = 1) Then
00477         lngDebugCtr = lngDebugCtr + 1
00478         SetStatusLog strDebugName & lngDebugCtr, TASKSTATUS_Success, 0, strMessage
00479     End If
00480
00481
00482 End Sub
00483
00484
00485
```

```
00001 '*****
00002 '
00003 '           © International Computers Limited 1999
00004 '
00005 '*****
00006 '
00007 ' Description
00008 '
00009 '     This batch calls ComboCORecordSpotCheck.
00010 '
00011 ' Authors (first name is last person to make any change)
00012 '
00013 '     R.Parkinson
00014 '
00015 '*****
00016 » *****
00017 Option Explicit
00018
00019 Private Const DEBUG_Required = 0 '* 1/0 turns on/off message boxes
00020 Private Const DEBUG_StatusLog_Required = 0 '* 1/0 turns on/off tasklog debug
00021
00022 Private Const INIT_ErrorRerun = 0 '* 1/0 turns on/off rerun on error
00023 Private Const INIT_RunOnce = 0 '* 1/0 turns on/off rerun checking
00024
00025 Private Const CONF_SignalBatch = 0 ' 1/0 turns on/off batch signalling
00026
00027 Private Const ERROR_NoError = 0
00028 Private Const ERROR_Generic = -10
00029 Private Const ERROR_ObjectCreation = -20
00030
00031 Private Const TASKSTATUS_NotStarted = -2
00032 Private Const TASKSTATUS_Started = -1
00033 Private Const TASKSTATUS_Success = 0
00034 Private Const TASKSTATUS_Warnings = 1
00035 Private Const TASKSTATUS_Interrrupted = 2
00036 Private Const TASKSTATUS_Error = 3
00037
00038 Private Const STORESTATUS_Open = 1
00039 Private Const STORESTATUS_Closing = 2
00040 Private Const STORESTATUS_Closed = 3
00041
00042 Private Const PATH_EventActivator =
00043 » "D:\Gstr\ApplicationServices\Tools\Bin\ICLSKEventActivator.exe"
00044
00045 Dim lngDebugCtr
00046 Dim strDebugName
00047 Dim strParms
00048
00049 ' *****
00050 ' *
00051 ' *
00052 ' *
00053 ' *****
00054 Sub Main
00055
00056 ' -- Ignore all errors.
00057 On Error Resume Next
00058
00059 Dim nTaskStatus ' The status of this particular task
00060 Dim nStoreStatus ' Store status (trading, not trading etc.)
00061 Dim nReturnCode ' General return code constant
00062
00063 Dim strScriptName ' The name of this particular script
00064 Dim nNewStoreStatus ' The new store status
00065 Dim pMachine ' Machine role
00066
00067 strScriptName = "Spot Check"
00068 strDebugName = "Spot Check " & formatdatetime(Now, 4) & "-"
00069 lngDebugCtr = 0
00070
00071 DEBUG("Start " & strScriptName)
00072
00073
```

1

```
00074 '*****
00075 '* Initialize
00076 '*****
00077 nTaskStatus = Initialize (strScriptName)
00078 DEBUG( "Initialize() returns nTaskStatus = " & nTaskStatus)
00079 If (nTaskStatus = ERROR_Generic) Or (nTaskStatus = TASKSTATUS_Success) Then
00080     Exit Sub
00081 End If
00082
00083
00084 '*****
00085 '* Log the beginning of the task.
00086 '*****
00087 SetStatusLog strScriptName, TASKSTATUS_Started, 0, ""
00088
00089
00090 '*****
00091 '* Retrieve any parameters - bypass non to retrieve.
00092 '*****
00093 ' strParms = GetParms (strScriptName)
00094
00095 strParms = ""
00096
00097 '*****
00098 '* Run ComboCORecordSpotCheck
00099 '*****
00100 nReturnCode = RunComboCORecordSpotCheck (strScriptName, strParms)
00101 If nReturnCode <> ERROR_NoError Then
00102     DEBUG("Error running ComboCORecordSpotCheck = " & nReturnCode)
00103     Exit Sub
00104 End If
00105
00106
00107 '*****
00108 '* Kick off another batch (if configured)
00109 '*****
00110 If CONF_SignalBatch = 1 Then
00111     ' Note that by default we just pass the parameters we were given.
00112     nReturnCode = AsyncEvent (strScriptName, "Unspecified", strParms)
00113     If nReturnCode <> ERROR_NoError Then
00114         DEBUG("Error signalling Unspecified = " & nReturnCode)
00115         Exit Sub
00116     End If
00117 End If
00118
00119 '*****
00120 '* Log the end of the task.
00121 '*****
00122 SetStatusLog strScriptName, TASKSTATUS_Success, 0, ""
00123
00124 '*****
00125 '* Remove the message from the queue
00126 '*****
00127 RemoveMessage
00128
00129 DEBUG("ComboCORecordSpotCheck terminated OK...")
00130
00131 End Sub
00132
00133
00134
00135 '*****
00136 '*
00137 '* Initialize
00138 '*
00139 '*****
00140 Function Initialize (strScriptName)
00141
00142     '-- Ignore all errors.
00143     On Error Resume Next
00144
00145     Dim oStatusLog
00146     Dim oTaskLog
00147     Dim strWarning
00148     Dim nStatus
00149
00150     ' Set the initial value for the function. This may get overwritten.
00151     nStatus = TASKSTATUS_NotStarted
```

```

00152
00153 ' -- Retrieve a reference to StatusLog service.
00154 Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00155 If oStatusLog Is Nothing Then
00156     strWarning = "Error creating StatusLog object. Err(" & Err.Number & ") = " &
» Err.Description
00157     DEBUG(strWarning)
00158     Initialize = ERROR_Generic
00159     Exit Function
00160 End If
00161
00162 ' Note: the next blocks can be bypassed by setting INIT_<whatever> at the head
» of this file.
00163 ' These variables are initially set by the batch app wizard.
00164
00165 ' First block: check if we've been run before *with errors* and if so bomb out
» with an error message.
00166 '
00167 If INIT_ErrorRerun = 1 Then
00168
00169     ' -- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00170     Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00171     If Err <> 0 Then
00172         If Err.Number - vbObjectError <> 100 Then ' Ignore log record does not exit
00173
00174             strWarning = "Error getting Last StatusLog. Err(" & Err.Number & ") = " &
» Err.Description
00175             'msgbox strWarning
00176             Initialize = ERROR_Generic
00177             Exit Function
00178         End If
00179
00180         Err.Clear
00181         Set oTaskLog = Nothing
00182     End If
00183
00184     ' -- Retrieve the last StatusLog (TaskLog.Status).
00185     If Not oTaskLog Is Nothing Then
00186         nStatus = oTaskLog.Status
00187         If nStatus <> TASKSTATUS_Success Then
00188             strWarning = "Warning: '" & strScriptName & "' already run with errors."
00189             SetStatusLog strScriptName, oTaskLog.Status, ERROR_NoError, strWarning
00190             DEBUG(strWarning)
00191         End If
00192     Else
00193         nStatus = TASKSTATUS_NotStarted
00194     End If
00195     Set oTaskLog = Nothing
00196 End If
00197
00198 ' Second block: check if we've been run before and if so bomb out with an error
» message.
00199 '
00200 If INIT_RunOnce = 1 Then
00201
00202     ' -- Create a reference to ICLSKBDSTaskLog.BOTaskLog object.
00203     Set oTaskLog = oStatusLog.GetLastStatusLog (strScriptName)
00204     If Err <> 0 Then
00205         strWarning = "Error getting Last StatusLog. Err(" & Err.Number & ") = " &
» Err.Description
00206         DEBUG(strWarning)
00207         Initialize = ERROR_Generic
00208         Exit Function
00209     End If
00210
00211     ' -- Retrieve the last StatusLog (TaskLog.Status).
00212     If Not oTaskLog Is Nothing Then
00213         nStatus = oTaskLog.Status
00214         If nStatus = TASKSTATUS_Success Then
00215             strWarning = "Warning: '" & strScriptName & "' already run."
00216             SetStatusLog strScriptName, TASKSTATUS_Success, ERROR_NoError, strWarning
00217             DEBUG(strWarning)
00218         End If
00219     Else
00220         nStatus = TASKSTATUS_NotStarted
00221     End If
00222     Set oTaskLog = Nothing
00223 End If

```

```

00224
00225     Set oStatusLog = Nothing
00226
00227     ' Return what we've found.
00228
00229     Initialize = nStatus
00230
00231 End Function
00232
00233
00234
00235 ' *****
00236 ' *
00237 ' * Get the parameters
00238 ' *
00239 ' *****
00240 Function GetParms (strScriptName)
00241
00242     '-- Ignore all errors.
00243     On Error Resume Next
00244
00245     Dim strEventName ' Name of the associated event for this batch
00246     Dim objEventDetail ' The associated event tdetail for the above event
00247     Dim objParms ' Object containing parameters
00248     Dim nCount ' General count
00249     Dim nLoopCount ' Loop-control count
00250     Dim strParameters ' Parameters
00251     Dim strWarning ' Warning message
00252
00253     strEventName = GSBOEvent.IGSBOEvent_EventName
00254     If Err.Number <> 0 Then
00255         strWarning = "Getting Event Name : Err(" & Err.Number & ") = " &
    » Err.Description
00256     End If
00257
00258     '-- Create a reference to the EventDetail object.
00259     If strWarning = "" Then
00260         Set objEventDetail = GSBOEvent.IGSBOEvent_EventDetail
00261         If Err.Number <> 0 Then
00262             strWarning = "Getting Event Detail1 : Err(" & Err.Number & ") = " &
    » Err.Description
00263         End If
00264     End If
00265
00266     '-- Create a reference to the AdditionalParameters collection.
00267     If strWarning = "" Then
00268         Set objParms = objEventDetail.AdditionalParameters
00269         If Err.Number <> 0 Then
00270             strWarning = "Get Additional Parameters : Err(" & Err.Number & ") = " &
    » Err.Description
00271         End If
00272     End If
00273
00274     '-- Count the number of the elements in the collection.
00275     If strWarning = "" Then
00276         nCount = objParms.Count
00277         If Err.Number <> 0 Then
00278             strWarning = "Get parameter count : Err(" & Err.Number & ") = " &
    » Err.Description
00279         End If
00280     End If
00281
00282     '-- Retrieve the parameters.
00283     If strWarning = "" Then
00284         If nCount = 0 Then
00285             strWarning = "No parameters"
00286         Else
00287             strParameters = ""
00288             For nLoopCount = 1 to nCount
00289                 strParameters = strParameters & "," & objParms.Item(1)
00290                 If Err.Number <> 0 Then
00291                     strWarning = "Error Getting Parameter " & nLoopCount & " : Err(" &
    » Err.Number & ") = " & Err.Description
00292                 End If
00293             Next
00294         End If
00295     End If
00296

```

```

00297     '-- Log the error.
00298     If strWarning <> "" Then
00299         SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00300         DEBUG (strWarning)
00301         strParameters = ""
00302     Else
00303         DEBUG ("Parameters = " & strParameters)
00304     End If
00305
00306     GetParms = strParameters
00307
00308 End Function
00309
00310
00311
00312
00313     ' *****
00314     ' *
00315     ' * Update "TaskLog" table
00316     ' *
00317     ' *****
00318 Sub SetStatusLog (strTaskID, nStatus, nResultCode, strNote)
00319
00320     '-- Ignore all errors.
00321     On Error Resume Next
00322
00323     Dim oStatusLog ' Status log object
00324     Dim strMsg ' The message to log
00325
00326     '   DEBUG("Logging Status : " & strNote)
00327
00328     '-- Retrieve a reference to StatusLog service.
00329     Set oStatusLog = Createobject ("ICLSKStatusLog.clsStatusLog")
00330     If oStatusLog Is Nothing Then
00331         strMsg = "Error creating StatusLog object"
00332         DEBUG (strMsg)
00333         Exit Sub
00334     End If
00335
00336     '-- Insert/Update "TaskLog" table.
00337     If strNote = "" Then
00338         oStatusLog.Log strTaskID, nStatus, nResultCode
00339     Else
00340         oStatusLog.Log strTaskID, nStatus, nResultCode, , , strNote
00341     End If
00342
00343     Set oStatusLog = Nothing
00344
00345 End Sub
00346
00347
00348
00349     ' *****
00350     ' *
00351     ' * Run the Model Batch Program
00352     ' *
00353     ' *****
00354 Function RunComboCORecordSpotCheck (strScriptName, strParameters)
00355
00356     On Error Resume Next
00357
00358     Dim oStatusLog ' Status Log Object
00359     Dim strMsg ' General messages
00360     Dim oModel ' Model Batch Application
00361     Dim nStatus ' Status code
00362     Dim nCopies
00363     Dim strWarning ' Warning message
00364     Dim lngStatus ' Status code (long)
00365
00366     nStatus = ERROR_NoError
00367
00368     DEBUG ("Running ComboCORecordSpotCheck")
00369
00370     ' *****
00371     ' * Create object
00372     ' *****
00373     If (nStatus = ERROR_NoError) Then
00374         DEBUG ("Creating ComboCORecordSpotCheck")

```

```

00375     Set oModel = Createobject("ComboCORecordSpotCheck.Application")
00376     If oModel Is Nothing Then
00377         DEBUG("Error creating ComboCORecordSpotCheck")
00378         strWarning = "Error creating ComboCORecordSpotCheck.Application object"
00379         nStatus = ERROR_ObjectCreation
00380     Else
00381         DEBUG("Created ComboCORecordSpotCheck OK")
00382     End If
00383 End If
00384
00385     '*****
00386     '* DO whatever it does, passing the parameter.
00387     '*****
00388     If (nStatus = ERROR_NoError) Then
00389         DEBUG ("Running ComboCORecordSpotCheck")
00390         Err.Clear
00391         nStatus = oModel.BeginSpotCheck (strParameters)
00392         If Err <> 0 Then
00393             strWarning = Err.Description
00394             nStatus = ERROR_Generic
00395         End If
00396     End If
00397
00398
00399     '*****
00400     '* Log error
00401     '*****
00402     If (nStatus <> ERROR_NoError) And (strWarning <> "") Then
00403         SetStatusLog strScriptName, TASKSTATUS_Error, nStatus, strWarning
00404     End If
00405
00406     RunComboCORecordSpotCheck = nStatus
00407 End Function
00408
00409
00410
00411
00412     '*****
00413     '*
00414     '* Async Event
00415     '*
00416     '*****
00417     Function AsyncEvent (strScriptName, strEventName, strParameters)
00418
00419         '-- Ignore all errors.
00420         On Error Resume Next
00421
00422         Dim objLauncher
00423         Dim strWarning
00424         Dim lngStatus
00425
00426         DEBUG("About to Fire Asynchronous Event : " & strEventName)
00427
00428
00429         '*****
00430         '* Create the launcher event
00431         '*****
00432         Set objLauncher = Createobject("ICLSKLaunch.LaunchEXE")
00433         If err.number <> 0 Then
00434             strWarning = "Error Creating Launcher Object - Err(" & Err.Number & ") = " &
» Err.Description
00435             SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning
00436             DEBUG(strWarning)
00437             AsyncEvent = ERROR_ObjectCreation
00438             Exit Function
00439         End If
00440
00441         '*****
00442         '* Set up parameters and run the exe !
00443         '*****
00444         objLauncher.ApplicationName = PATH_EventActivator
00445         objLauncher.ApplicationArgs = strEventName & " " & strParameters
00446
00447         objLauncher.Execute
00448         If err.number <> 0 Then
00449             strWarning = "Error Starting Event Activator. Err(" & Err.Number & ") = " &
» Err.Description
00450             SetStatusLog strScriptName, TASKSTATUS_Error, ERROR_Generic, strWarning

```

```
00451     DEBUG(strWarning)
00452     AsyncEvent = ERROR_ObjectCreation
00453     Exit Function
00454 End If
00455
00456     Set objLauncher = Nothing
00457
00458     AsyncEvent = ERROR_NoError
00459
00460 End Function
00461
00462
00463
00464     '*****
00465     '*
00466     '* Remove the message from the queue
00467     '*
00468     '*****
00469 Function RemoveMessage
00470
00471     Dim iQueueInfo
00472     Dim iQueue
00473     Dim iMessageDest
00474
00475     Set iQueueInfo = Createobject("MSMQ.MSMQQueueInfo")
00476     iQueueInfo.PathName = ".\GSBOEventTargetQueue1"
00477     Set iQueue = iQueueInfo.Open(1,0)
00478
00479     Set iMessageDest = iQueue.Receive(,,1000)
00480
00481     If iMessageDest Is Nothing Then
00482         DEBUG ("There are no messages in the queue.")
00483     Else
00484         DEBUG ("The first message was removed from the queue.")
00485     End If
00486
00487 End Function
00488
00489
00490
00491     ' *****
00492     '*
00493     '* DEBUG messages (if required)
00494     '*
00495     ' *****
00496 Sub DEBUG (strMessage)
00497
00498     If (DEBUG_Required = 1) Then
00499         MsgBox strMessage
00500     End If
00501
00502     If (DEBUG_StatusLog_Required = 1) Then
00503         lngDebugCtr = lngDebugCtr + 1
00504         SetStatusLog strDebugName & lngDebugCtr, TASKSTATUS_Success, 0, strMessage
00505     End If
00506
00507
00508 End Sub
00509
00510
00511
```

```

00001 ' Windows Script Host Sample Script
00002 '
00003 ' -----
00004 '           Copyright (C) 1996 Microsoft Corporation
00005 '
00006 ' You have a royalty-free right to use, modify, reproduce and distribute
00007 ' the Sample Application Files (and/or any modified version) in any way
00008 ' you find useful, provided that you agree that Microsoft has no warranty,
00009 ' obligations or liability for any Sample Application Files.
00010 ' -----
00011 '
00012 'This script deletes users from the Windows NT DS
00013 'via ADSI. The script reads an EXCEL spreadsheet (DelUsers.xls) that contains a
» page
00014 'of users to delete.
00015 '
00016 'The sample uses the directory root "LDAP://DC=ArcadiaBay,DC=Com,O=Internet
00017 'Change the directory path in the EXCEL spreadsheet to match your DS
00018 'before running this sample.
00019 '
00020 ''To add users, run ADDUSERS.VBS with %windir%\Your Samples Directory
» here"\AddUsers.XLS.
00021 ''To Delete users, run DELUSERS.VBS with %windir%\Your Samples Directory
» here"\DelUsers.XLS.

00022
00023
00024 Dim oXL
00025 Dim u
00026 Dim c
00027 Dim root
00028 Dim ou
00029 Dim TextXL
00030 Dim CRLF
00031 Dim oArgs
00032
00033
00034 'Get the command line args
00035 Set oArgs=wscript.arguments
00036
00037 CRLF = Chr(13) & Chr(10)
00038
00039 'If no command line arguments provided, prompt for file containing users to
» add/delete
00040 If oArgs.Count = 0 Then
00041     TextXL = InputBox("This scripts reads an Excel spread sheet and deletes " & _
00042         "users from the Windows NT DS via ADSI." & CRLF & CRLF & _
00043         "Before starting, change the DS root in the EXCEL spreadsheet to match " & _
»
00044         "your DS."& CRLF & CRLF & _
00045         "Type in the path of a file containing users to add or delete" & CRLF & CRLF
» & _
00046         "Sample Add User file: ADDUSERS.XLS" & CRLF & _
00047         "Sample Delete User file: DELUSERS.XLS" & CRLF)
00048     'Else file containing users is the first argument
00049 Else
00050     TextXL = oArgs.item(0)
00051 End If
00052
00053 If TextXL = "" Then
00054     WScript.Echo "No input file provided.  stopping script now."
00055     WScript.Quit(1)
00056 End If
00057
00058 'We will use ou to control loop, so set initial value to null
00059 ou = ""
00060
00061 'Start EXCEL and display it to the user
00062 Set oXL = WScript.CreateObject("EXCEL.application")
00063 oXL.Visible = True
00064
00065 'Open the workbook passed in the command line
00066 oXL.workbooks.open TextXL
00067
00068
00069 'Now do deletes
00070 '
00071 'Activate the Delete page

```

```
00072 oXL.sheets("Delete").Activate
00073
00074 'Set the cell cursor to the DS root
00075 oXL.ActiveSheet.range("A2").Activate ' this cell has the DS root in it
00076
00077 'Show it to the user
00078 'WScript.Echo oXL.activecell.Value
00079
00080 root = oXL.activecell.Value
00081 oXL.activecell.offset(1, 0).Activate
00082
00083 'Until we run out of rows...
00084 Do While oXL.activecell.Value <> ""
00085
00086     'If the requested OU is different
00087     If oXL.activecell.Value <> ou Then
00088         ou = oXL.activecell.Value
00089
00090         'Compose the new ou path...
00091         s = "LDAP://" + OU + "," + root
00092
00093         'Show it to the user
00094         WScript.Echo s
00095
00096         'Get the new container
00097         Set c = Getobject(s)
00098     End If
00099
00100     'Compose the user name
00101     uname = "CN=" + oXL.activecell.offset(0, 1).Value + " " + oXL.activecell.offset(
» 0, 2).Value
00102     ' wscript.echo uname
00103     'Delete the user
00104     Call c.Delete("user", uname)
00105     oXL.activecell.offset(1, 0).Activate ' next row
00106 Loop
00107
00108 'Done. close excel spreadsheet
00109 oXL.application.quit
```

```
00001 ' Windows Script Host Sample Script
00002 '
00003 ' -----
00004 '           Copyright (C) 1996 Microsoft Corporation
00005 '
00006 ' You have a royalty-free right to use, modify, reproduce and distribute
00007 ' the Sample Application Files (and/or any modified version) in any way
00008 ' you find useful, provided that you agree that Microsoft has no warranty,
00009 ' obligations or liability for any Sample Application Files.
00010 ' -----
00011
00012 ' This sample will display Windows Scripting Host properties in Excel.
00013
00014 L_Welcome_MsgBox_Message_Text = "This script will display Windows Scripting Host
» properties in Excel."
00015 L_Welcome_MsgBox_Title_Text = "Windows Scripting Host Sample"
00016 Call Welcome()
00017
00018
00019 ' *****
»
00020 ' *
00021 ' * Excel Sample
00022 ' *
00023 Dim objXL
00024 Set objXL = WScript.CreateObject("Excel.Application")
00025
00026 objXL.Visible = True
00027
00028 objXL.WorkBooks.Add
00029
00030 objXL.Columns(1).ColumnWidth = 20
00031 objXL.Columns(2).ColumnWidth = 30
00032 objXL.Columns(3).ColumnWidth = 40
00033
00034 objXL.Cells(1, 1).Value = "Property Name"
00035 objXL.Cells(1, 2).Value = "Value"
00036 objXL.Cells(1, 3).Value = "Description"
00037
00038 objXL.Range("A1:C1").Select
00039 objXL.Selection.Font.Bold = True
00040 objXL.Selection.Interior.ColorIndex = 1
00041 objXL.Selection.Interior.Pattern = 1 'xlSolid
00042 objXL.Selection.Font.ColorIndex = 2
00043
00044 objXL.Columns("B:B").Select
00045 objXL.Selection.HorizontalAlignment = &hFFFFEFD ' xlLeft
00046
00047 Dim intIndex
00048 intIndex = 2
00049
00050 Sub Show(strName, strValue, strDesc)
00051     objXL.Cells(intIndex, 1).Value = strName
00052     objXL.Cells(intIndex, 2).Value = strValue
00053     objXL.Cells(intIndex, 3).Value = strDesc
00054     intIndex = intIndex + 1
00055     objXL.Cells(intIndex, 1).Select
00056 End Sub
00057
00058 '
00059 ' Show WScript properties
00060 '
00061 Call Show("Name", WScript.Name, "Application Friendly Name")
00062 Call Show("Version", WScript.Version, "Application Version")
00063 Call Show("FullName", WScript.FullName, "Application Context: Fully Qualified
» Name")
00064 Call Show("Path", WScript.Path, "Application Context: Path Only")
00065 Call Show("Interactive", WScript.Interactive, "State of Interactive Mode")
00066
00067
00068 '
00069 ' Show command line arguments.
00070 '
00071 Dim colArgs
00072 Set colArgs = WScript.Arguments
00073 Call Show("Arguments.Count", colArgs.Count, "Number of command line arguments")
00074
```

```
00075 { For i = 0 to colArgs.Count - 1
00076 {   objXL.Cells(intIndex, 1).Value = "Arguments(" & i & ")"
00077 {   objXL.Cells(intIndex, 2).Value = colArgs(i)
00078 {   intIndex = intIndex + 1
00079 {   objXL.Cells(intIndex, 1).Select
00080 { Next
00081
00082
00083
00084 { *****
>
00085 { *
00086 { * Welcome
00087 { *
00088 { Sub Welcome()
00089 {   Dim intDoIt
00090
00091 {   intDoIt = MsgBox(L_Welcome_MsgBox_Message_Text, _
00092 {     vbOKCancel + vbInformation, _
00093 {     L_Welcome_MsgBox_Title_Text )
00094 {   If intDoIt = vbCancel Then
00095 {     WScript.Quit
00096 {   End If
00097 { End Sub
00098
```