

How to save money on Oracle Java Client Licenses

Oracle has multiple software for Java Clients. You can use the Enterprise version (JEE), but also the Runtime Engine (JRE). There is one big difference: you have to pay for the commercial use of the JEE. But I will give a work around that will save you money!

1. On the [Licensing and Distribution FAQs](#) you find additional license information about the Java Runtime Engine:

I am an administrator. Can I install Java on all the computers at my company?

Yes, you may make unlimited copies for internal use for the purpose of running applications on Java-enabled general purpose desktop computers and servers.

Yes, you may make unlimited copies for internal use for the purpose of running applications on Java-enabled general purpose desktop computers and servers.

Do I need to purchase a license if I need the Java Runtime Environment for multiple users in my company?

No, you may make unlimited copies for internal use for the purpose of running applications on Java-enabled general purpose desktop computers and servers.

No, you may make unlimited copies for internal use for the purpose of running applications on Java-enabled general purpose desktop computers and servers.

2. However, the JEE has different license conditions. It is mentioned on the [Java Platform, Standard Edition MSI Enterprise JRE Installer Guide](#):

Note:

The MSI Enterprise JRE Installer requires a commercial license for use in production. To learn more about commercial features and how to enable them, see [Oracle Java SE Advanced & Suite Products](#).

The MSI Enterprise JRE Installer requires a commercial license for use in production.

However, there is a work around. Use the regular JRE installation instead.

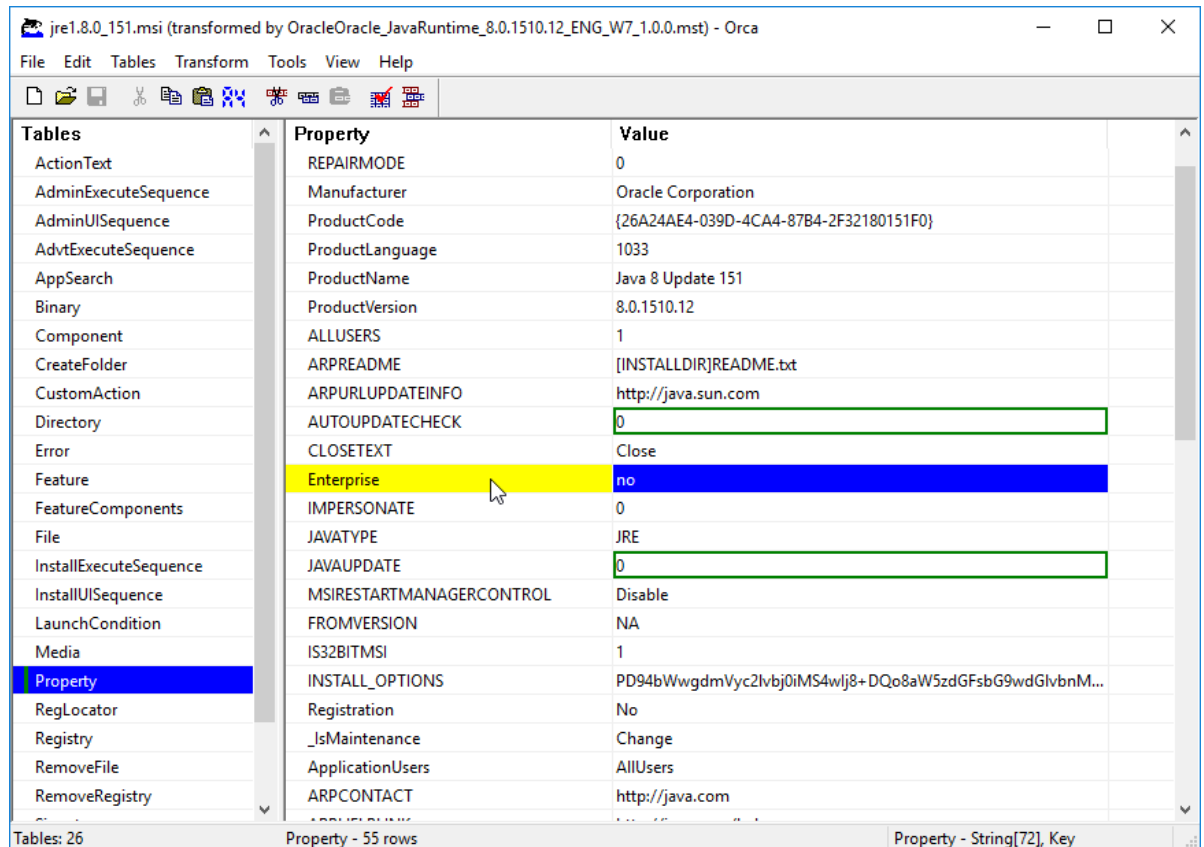
1. Download the latest JRE from [Java Downloads for All Operating Systems](#).
2. Run the installer, but stop when this screen appears:



Installation screen

3. The MSI is extracted from the installer in one of the subfolders of `C:\Users\%UserName%\AppData\LocalLow\Oracle\Java`.
4. Quit the installer.
5. Use Orca to create a transform file, based on the MSI. In this case it is `jre1.8.0_151.msi`.

6. And the enterprise version is not used:



The screenshot shows the Orca MSI editor window titled "jre1.8.0_151.msi (transformed by OracleOracle_JavaRuntime_8.0.1510.12_ENG_W7_1.0.0.mst) - Orca". The interface includes a menu bar (File, Edit, Tables, Transform, Tools, View, Help) and a toolbar. On the left, a "Tables" pane lists various MSI tables, with "Property" selected. The main area displays a table with three columns: "Property", "Value", and an empty column. The "Enterprise" property is highlighted in yellow, and its value "no" is highlighted in blue. Other properties include REPAIRMODE (0), Manufacturer (Oracle Corporation), ProductCode, ProductLanguage (1033), ProductName (Java 8 Update 151), ProductVersion (8.0.1510.12), ALLUSERS (1), ARPREADME ([INSTALLDIR]README.txt), ARPURLUPDATEINFO (http://java.sun.com), AUTOUPDATECHECK (0), CLOSETEXT (Close), IMPERSONATE (0), JAVATYPE (JRE), JAVAUPDATE (0), MSIRESTARTMANAGERCONTROL (Disable), FROMVERSION (NA), IS32BITMSI (1), INSTALL_OPTIONS (PD94bWwgdmVyc2lvbj0iMS4wLjE5MTAuMTI=...), Registration (No), _IsMaintenance (Change), ApplicationUsers (AllUsers), and ARPCONTACT (http://java.com). The status bar at the bottom indicates "Tables: 26", "Property - 55 rows", and "Property - String[72], Key".

Property	Value
REPAIRMODE	0
Manufacturer	Oracle Corporation
ProductCode	{26A24AE4-039D-4CA4-87B4-2F32180151F0}
ProductLanguage	1033
ProductName	Java 8 Update 151
ProductVersion	8.0.1510.12
ALLUSERS	1
ARPREADME	[INSTALLDIR]README.txt
ARPURLUPDATEINFO	http://java.sun.com
AUTOUPDATECHECK	0
CLOSETEXT	Close
Enterprise	no
IMPERSONATE	0
JAVATYPE	JRE
JAVAUPDATE	0
MSIRESTARTMANAGERCONTROL	Disable
FROMVERSION	NA
IS32BITMSI	1
INSTALL_OPTIONS	PD94bWwgdmVyc2lvbj0iMS4wLjE5MTAuMTI=...
Registration	No
_IsMaintenance	Change
ApplicationUsers	AllUsers
ARPCONTACT	http://java.com

Orca – enterprise no.

A vbscript to uninstall previous versions and install the latest version

```
'  
  
=====
```

```
' Installs Java Runtime 8 Update 151  
' Created by Willem-Jan Vroom  
' Version history:  
'  
' 1.0.0  
'   Initial version  
'  
=====
```

```
'  
-----
```

```
-----  
' Declare the most variables.  
' -----  
-----
```

```
Option Explicit
```

```
Dim objShell : set objShell  
= WScript.CreateObject("WScript.Shell")  
Dim objNetWork : set objNetWork  
= Wscript.CreateObject("WScript.Network")  
Dim objLogFileFSO : Set  
objLogFileFSO =  
CreateObject("Scripting.FileSystemObject")  
Dim objFSO : Set objFSO  
= CreateObject("Scripting.FileSystemObject")  
Dim objProcessEnv : Set  
objProcessEnv =  
objShell.Environment("PROCESS")  
Dim objFolder  
Dim objFile  
Dim colFiles  
Dim objWMIService : Set  
objWMIService =  
GetObject("winmgmts:{impersonationLevel=impersonate}!\\.  
\root\cimv2")  
Dim fn_objWMIService : Set  
fn_objWMIService =  
GetObject("winmgmts:{impersonationLevel=impersonate}!\\.  
\root\cimv2")  
Dim objReg : Set objReg  
=  
GetObject("winmgmts:{impersonationLevel=impersonate}!\\.  
\root\default:StdRegProv")  
Dim objLogFile  
Dim colListOfServices  
Dim objService  
Dim strCurrentDir : strCurrentDir  
=  
Left(Wscript.ScriptFullname,  
InstrRev(Wscript.ScriptFullname, "\"))  
strCurrentDir
```

```

= Left(strCurrentDir, len(strCurrentDir) - 1)
    Dim strcomputerName :
strcomputerName =
objProcessEnv("COMPUTERNAME")
    Dim strLogLocation : strLogLocation
= "C:\WINDOWS\system32\Logfiles"
    Dim strOutputFile : strOutputFile
= strLogLocation & "\" & strcomputerName &
"_JavaRuntime1.8.0_151_Install_" &
Replace(FormatDateTime(Now(), 2), "/", "-") & ".log"
    Dim strCurrentUser : strCurrentUser
= ""
    Dim strCurrentUserSID :
strCurrentUserSID = ""
    Dim strOS : strOS
= ""
    Dim strValueName : strValueName
= ""
    Dim strValue : strValue
= ""
    Dim strArchitecture :
strArchitecture = ""
    Dim strMSI : strMSI
= ""
    Dim strMST : strMST
= ""
    Dim strMSILogFile : strMSILogFile
= ""
    Dim strCommand : strCommand
= ""
    Dim strSoftwareName :
strSoftwareName = "%Java 6 %, %Java 7
%, %Java 8 %"
    Dim arrSoftwareToUninstall :
arrSoftwareToUninstall =
split(strSoftwareName, ",")
    Dim strFileNamesToKill :
strFileNamesToKill =
"iexplore.exe, chrome.exe, firefox.exe, javaw.exe, webstart.
exe"
    Dim arrFileNamesToKill :

```

```

arrFileNamesToKill                                     =
split(strFileNamesToKill, ",")
  Dim objSoftwareToUninstall
  Dim arrValues, arrSubKeys, arrArguments, strFileName,
strCommonPrograms

  Dim colProcess, objProcess
  Dim valResult, valOSBuildNumber, valCounter
  Dim objSubfolder
  Dim objItem, colItems
  Dim strKeyPath

  Const ForWriting          = 2
  Const ForReading          = 1
  Const ForAppending        = 8
  Const OverWriteFiles     = True
  const HKEY_CURRENT_USER  = &H80000001
  const HKEY_LOCAL_MACHINE = &H80000002
  Const HKEY_USERS          = &H80000003

' -----
' Create the log location (if not exists)
' Open the logfile.
' -----
-----

CreateFolderStructure(strLogLocation)
OpenLogFile()
WriteToLog "- ACTION: script started."

' -----
' Detect the current OS.
' -----
-----

Set colItems = objWMIService.ExecQuery("Select
Caption, BuildNumber from Win32_OperatingSystem")
For Each objItem in colItems
  strOS          = objItem.Caption

```

```

        valOSBuildNumber = objItem.BuildNumber
    next

' -----
-----
' Detect the current processor architecture.
' -----
-----

                                                                    if
objFS0.FolderExists(objShell.ExpandEnvironmentStrings("%
windir%") & "\SysWOW64\Config") then
    strArchitecture = "x64"
    else
    strArchitecture = "x86"
end if

    WriteToLog("##### Details regarding operating
system and logged on user          #####")
    WriteToLog("Found Operating System:      " & strOS)
    WriteToLog("Found architecture:          " &
strArchitecture)

' -----
-----
' Find pending reboots and write it to the log file.
' -----
-----

if PendingRebootRequired Then
    WriteToLog("Pending restarts:           Yes")
    else
    WriteToLog("Pending restarts:           No")
End if
    WriteToLog("##### End details regarding operating
system and logged on user          #####")
    WriteToLog(" ")

' -----
-----
' Sets SEE_MASK_NOZONECHECKS to 1.

```



```
' That will avoid a screen that asks for permission to
run an application from
' a share or mapped drive.
```

```
' -----
-----
```

```
objProcessEnv("SEE_MASK_NOZONECHECKS") = 1
```

```
' -----
-----
```

```
' Check if the parameter /uninstall is given.
' In that case: remove the product and quit.
```

```
' -----
-----
```

```
if wscript.arguments.count > 0 Then
    Set arrArguments = wscript.arguments
    for valCounter = 0 to wscript.arguments.count - 1
        if
instr(lcase(arrArguments(valCounter)), "/uninstall") > 0
or instr(lcase(arrArguments(valCounter)), "/remove") > 0
Then
            UninstallJavaAllVersions
            CloseLogFile()
            wscript.Quit(0)
        end if
    next
end if
```

```
KillProcess()
```

```
UninstallJavaAllVersions
```

```
' -----
-----
```

```
' Install Java 8 (x86 version only)
```

```
' -----
-----
```

```
strMSI=FindFileInTheFolder(strCurrentDir & "\x86",
"msi")
```

```

    strMST=FindFileInTheFolder(strCurrentDir & "\x86",
"mst")

    KillProcess()
    strMSILogFile = left(strMST,len(strMST)-4) & "
('x86').log"

    WriteToLog("##### Start installing Java 8.0.151.
#####")
    WriteToLog(" ")
    strCommand = "msiexec /i " & chr(34) & strCurrentDir &
"\x86\" & strMSI & chr(34)
    strCommand = strCommand & " TRANSFORMS=" & chr(34) &
strCurrentDir & "\x86\" & strMST & chr(34)
    strCommand = strCommand & " /qn /l*v " & chr(34) &
strLogLocation & "\" & strMSILogFile & chr(34)
    WriteToLog ("Running command: " & strCommand)
    valResult = objShell.Run(strCommand,6,True)
    WriteToLog ("Result: " & valResult)
    WriteToLog(" ")
    WriteToLog("##### End installing Java 8.0.151.
#####")
    WriteToLog(" ")

' -----
-----
' Uninstall the Automatic Update function (msi)
' -----
-----

    WriteToLog("##### Start uninstalling the Java
Update Manager. #####")
    WriteToLog(" ")

    strMSI          = "au.msi"
        strMST
                                =
"Oracle_JavaRuntime_DisableAutomaticUpdate_8.0.1510.1_EN
G_W7_1.0.0.mst"
    strMSILogFile = left(strMST,len(strMST)-4) & ".log"

```

```

    strCommand = "msiexec /x " & chr(34) & strCurrentDir &
"\AutoUpdater\" & strMSI & chr(34)
    strCommand = strCommand & " /qn /l*v " & chr(34) &
strLogLocation & "\" & strMSILogFile & chr(34)
    WriteToLog ("Running command: " & strCommand)
    valResult = objShell.Run(strCommand,6,True)

    if valResult = 1605 then
        valResult = 0 ' 1605 -> This action is only valid
for products that are currently installed.
    end if

    WriteToLog ("Result: " & valResult)
    WriteToLog(" ")
    WriteToLog("##### End uninstalling the Java
Update Manager. #####")
    WriteToLog(" ")

    ModifyIcons()

    CloseLogFile()

'
=====
=====
' All the subroutines and functions.
'
=====
=====

Function fnKillProcess(strProcessName)

' -----
-----
' Function: fnKillProcess(strProcessName)
' Terminates the given processname.
' -----
-----

    Set colProcess = fn_objWMIService.ExecQuery ("Select *
From Win32_Process")

```

```

    For Each objProcess In colProcess
                                                If
Instr(LCase(objProcess.Name),LCase(strProcessName)) > 0
Then
        objShell.Run "TASKKILL /F /T /IM " &
objProcess.Name, 0, False
        objProcess.Terminate()
        WriteToLog("Terminating application: " &
objProcess.Name)
    End If
Next

```

```
End Function
```

```
Sub OpenLogFile()
```

```

' -----
' -----
' Subroutine: OpenLogFile()
' The name of the logfile is mentinoed in the variabele
strOutputFile.
' -----
' -----

```

```

    If objLogFileFS0.FileExists(strOutputFile) Then
                                                Set      objLogFile      =
objLogFileFS0.OpenTextFile(strOutputFile, ForAppending)
    Else
                                                Set      objLogFile      =
objLogFileFS0.CreateTextFile(strOutputFile)
    End If

```

```
End Sub
```

```
Sub CloseLogFile()
```

```

' -----
' -----
' Subroutine: CloseLogFile()
' Close the log file.
' -----
' -----

```

```
-----  
    WriteToLog("##### End installation of Java 8.  
#####")  
    WriteToLog("- ACTION: script ended.")  
    WriteToLog(" ")  
    objLogFile.Close  
    Set objLogFileFSO = Nothing
```

End Sub

Function WriteToLog(sLogMessage)

```
' -----  
-----  
' Function: WriteToLog(sLogMessage)  
' Writes an entry 'sLogMessage' in the logfile.  
' -----  
-----
```

```
    if instr(sLogMessage, "- ACTION: ") = 0 then  
sLogMessage = "          " & sLogMessage  
    objLogFile.WriteLine("Time: " & now & "    " &  
sLogMessage)
```

End Function

Function PendingRebootRequired()

```
' -----  
-----  
' Function: PendingRebootRequired()  
' Detects if there are pending reboots by querying the  
registry:  
'                                                                 *  
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVer  
sion\Component Based Servicing  
'   Key: RebootPending    (Only on Vista and above)  
'                                                                 *  
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVer  
sion\WindowsUpdate\Auto Update  
'   Key: RebootRequired
```

```

'                                                                 *
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session
Manager
'   Value: PendingFileRenameOperations
'
' (C) by Peter Fonk
' -----
-----

    PendingRebootRequired = False

    If valOSBuildNumber >= 6001 Then
        PendingRebootRequired = KeyExists("RebootPending",
"SOFTWARE\Microsoft\Windows\CurrentVersion\Component
Based Servicing\")
        if PendingRebootRequired Then
            Exit Function
        end if
    End If

    PendingRebootRequired = KeyExists("RebootRequired",
"SOFTWARE\Microsoft\Windows\CurrentVersion\WindowsUpdate
\Auto Update\")
    if PendingRebootRequired Then
        Exit Function
    end if

                                PendingRebootRequired           =
ValueExists("PendingFileRenameOperations",
"SYSTEM\CurrentControlSet\Control\Session Manager\")

End Function

Function KeyExists(Key, KeyPath)
' -----
-----
' Function: KeyExists(Key, KeyPath)
' Checks to see if the 'Key' in 'KeyPath' exists.
' -----
-----

```

```

-----

    Dim arrSubKeys
        Dim oReg: Set oReg =
GetObject("winmgmts:!root/default:StdRegProv")
        Dim regReadResult : regReadResult =
oReg.EnumKey(HKEY_LOCAL_MACHINE, KeyPath, arrSubKeys)
    Dim i
    KeyExists = False
    If regReadResult = 0 And IsArray(arrSubKeys) Then
        For i = 0 To UBound(arrSubKeys)
            If InStr(arrSubKeys(i), Key) <> 0 Then
                KeyExists = True
                i = UBound(arrSubKeys)
            End If
        Next
    End If
    Set oReg = Nothing

```

End Function

Function ValueExists(Key, KeyPath)

```

' -----
' -----
' Function: ValueExists(Key, KeyPath)
' Checks to see if the value 'Key' in 'KeyPath' exists.
' -----
' -----

```

```

    Dim arrSubValues
        Dim oReg: Set oReg =
GetObject("winmgmts:!root/default:StdRegProv")
        Dim regReadResult : regReadResult =
oReg.EnumValues(HKEY_LOCAL_MACHINE, KeyPath,
arrSubValues)
    Dim i
    ValueExists = False
    If regReadResult = 0 And IsArray(arrSubValues) Then
        For i = 0 To UBound(arrSubValues)
            If InStr(arrSubValues(i), Key) <> 0 Then

```

```

        ValueExists = True
        i = UBound(arrSubValues)
    End If
Next
End If
Set oReg = Nothing

End Function

Sub CreateFolderStructure(strFolderNameToBeCreated)

' -----
'
'                               Subroutine:
CreateFolderStructure(strFolderNameToBeCreated)
'   Creates the map as mentioned in
strFolderNameToBeCreated.
' -----
' -----

    Dim arrFoldersTMP : arrFoldersTMP = split
(strFolderNameToBeCreated, "\")
    Dim strFolder : strFolder = ""
    Dim objFolderTMP

    For Each objFolderTMP in arrFoldersTMP
        strFolder = strFolder & objFolderTMP
        If NOT objFS0.FolderExists(strFolder) Then
            objFS0.CreateFolder(strFolder)
        end If
        strFolder = strFolder & "\"
    Next

End Sub

Function GetSIDFromUser(UserName)

' -----
' -----
' Function: GetSIDFromUser(UserName)
' Gets the SID from the give username.

```


' We use the registry to avoid an empty result if run under the SYSTEM account.

' -----

```
Dim DomainName, Result, WMIUser

If InStr(Username, "\") > 0 Then
    DomainName = Mid(Username, 1, InStr(Username, "\") - 1)
    Username = Mid(Username, InStr(Username, "\") + 1)
Else
    DomainName =
CreateObject("ADSystemInfo").DomainShortName
End If

On Error Resume Next
    Set WMIUser =
GetObject("winmgmts:{impersonationlevel=impersonate}!" &
    & "/root/cimv2:Win32_UserAccount.Domain='" &
DomainName & "' _
    & ",Name='" & Username & "'")
If Err.Number = 0 Then
    Result = WMIUser.SID
Else
    Result = ""
End If
On Error GoTo 0

GetSIDFromUser = Result

End Function
```

```
Function UninstallJavaAllVersions()
    Dim result
    Dim objSoftware
    Dim colSoftware

    WriteToLog("##### Start uninstalling all
previous Java versions. #####")
```

```

        For Each objSoftwareToUninstall in
arrSoftwareToUninstall
            Set colSoftware =
objWMIService.ExecQuery("Select * from Win32_Product
Where Name LIKE '" & objSoftwareToUninstall & "'")
            For Each objSoftware in colSoftware
                WriteToLog "- ACTION: Uninstalling " &
objSoftware.Name & "."
                result = objShell.Run("msiexec.exe /x "
& objSoftware.IdentifyingNumber & "
REBOOT=ReallySuppress /qn /l*v " & chr(34) &
"c:\windows\system32\logfiles\Uninstalling " &
objSoftware.Name & ".log" & chr(34),0,True)
                WriteToLog "- ACTION: " &
objSoftware.Name & " removed." & vbCrLf
            Next
        Next
    Next

        WriteToLog("##### End uninstalling all
previous Java versions. #####")
        WriteToLog(" ")

        Set objSoftware = Nothing
        Set colSoftware = Nothing
    End Function

Function KillProcess()
    Dim objProcess
        Dim colProcess : Set colProcess =
objWMIService.ExecQuery ("Select * from Win32_Process")
        For Each objProcess in colProcess
            Dim strFileName, strTemp
                For Each strFileName In
arrFileNamesToKill
                    If LCase(objProcess.Name) =
LCase(strFileName) Then
                        On Error Resume Next
                            objWshShell.Run
"TASKKILL /F /T /IM " & objProcess.Name, 0, False
                            objProcess.Terminate()
                            WriteToLog "- ACTION: "

```

```

& objProcess.Name & " terminated"
                                On Error Goto 0
                                End If
                            Next
                        Next
                    Set objProcess = Nothing
                    Set colProcess = Nothing
End Function

Sub ModifyIcons()

' -----
' -----
' Subroutine: ModifyIcons()
' Removes all the Java icons from both the Desktop and
Start Menu and replace it by a modified
' version
' -----
' -----

    WriteToLog(" ")
    WriteToLog("----- Modify the Start Menu and
remove unneeded icons. -----")
    WriteToLog(" ")

        strKeyPath =
"SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\Shell
Folders"
        strValueName = "Common Programs"
                                objReg.GetStringValue
HKEY_LOCAL_MACHINE, strKeyPath, strValueName, strCommonProg
rams

        if objFS0.FolderExists(strCommonPrograms & "\Java")
Then
            objFS0.DeleteFolder(strCommonPrograms & "\Java")
            WriteToLog("The folder '" & strCommonPrograms &
"\Java' has been deleted.")
        else
            WriteToLog("The folder '" & strCommonPrograms &
"\Java' does not exists thus not deleted.")

```

```

End if

WriteToLog(" ")
WriteToLog("----- End modifying Start Menu.
-----")
WriteToLog(" ")

End Sub

Function FindFileInTheFolder(strFolderName,
strExtension)

' -----
-----
-----
' Function: FindFileInTheFolder(strFolderName,
strExtension)
' Returns the full file name in the folder
strFolderName and the extension strExtension
' -----
-----
-----

Set objFolder = objFSO.GetFolder(strFolderName)
Set colFiles = objFolder.Files
FindFileInTheFolder = ""

For Each objFile in colFiles
    if lcase(right(objFile.Name,3)) =
lcase(strExtension) Then
        FindFileInTheFolder = objFile.Name
    end if
Next
End Function

```

You can [download the script – excluding the MSI – here.](#)