

Input on CleanUp script

Source: <http://www.tech-archive.net/Archive/Scripting/microsoft.public.scripting.wsh/2004-10/0280.html>

From: Egil Hansen (*egil_at_dailyrush.dk*)

Date: 10/15/04

Date: Fri, 15 Oct 2004 12:39:22 +0200

Hi all

I'm quite inexperienced with WSH, but I've managed to write a script that seem to work as its suppose ĩ Š. But I would really like some feedback on it, from some of you guys with lots of experience. I've made quite a few comments in the code, so it should be understandable.

The scripts job is to search through a folder, and find all files that are older then a XX days (e.g. haven't been modified/created for XX days), and delete them. It should also remove all empty subfolders.

Since I'm going to be using this script in a production environment, I'm going to need logging, which the script also supports, and I've included support for backup as well. If the script is used with a backup folder specified, files selected for deletion, will be moved to that folder instead.

I ran in to a few problems with the backup support. I need to recreate the directory structure from the original folder in the backup folder, which turned out to be more difficult to do then I initially thought. I ended up chickening out, and calling "XCOPY <target> <dest> /T /Y" from the script, which simply makes a copy of the directory structure, but I donâ€™t really like that solution (any good ideas anybody?).

In the first ~110 lines I mostly check the input given to the script. Rest should be more interesting.

Hereâ€™s the script (I have attached the script file to this mail as well, may be easiere to read then):

Best regards, Egil Hansen.

```
/*  
 * CleanUp  
 *  
 * Description: This script will search a specified folder for files
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
* that have not been modified/created for xx days and delete them
(with/without backup).
* It will also delete empty folders.
*
* Version: 0.03
* By: Egil Hansen
*
* Errorcodes returned by this script:
* 1 = The target folder does not exist.
* 2 = Invalid input.
* 3 = Could NOT create backup folder.
*
*
* Changelog:
*
* Version 0.03
* – Files and folders objects are now added to a array and deleted
after the iteration of
* target folder.
*
*
*
*/
```

```
// Declaring variables
```

```
var fso, targetFolder, backupFolder, deleteDays, cd, useBackup;
var strTargetFolder, strBackupFolder, args0, args1, args2, noArgs;
var fileCounter = 0, folderCounter = 0;
var arrayFiles = new Array();
var arrayFolders = new Array();
```

```
// Testing for required/valid arguments and assigning...
```

```
noArgs = new String(" CleanUp.js requires the following
arguments:\n\nTarget folder:\t\t/path:<folder>\nDelete
days:\t\t/days:<number of
days>\nLogfile:\t\t/logfile:<logfile>\nOptional – Backup
folder:\t/backup:<folder>\n\nExample:\n\tWithout backup:\tCleanUp.js
/path:c:\\public_share /days:14 /logfile:c:\\logfile.txt\n\tWith
backup:\tCleanUp.js /path:c:\\public_share /days:14
/logfile:c:\\logfile.txt /backup:c:\\backup\n\nTip:\n\tUse Cscript to
run the script instead of Wscript.\nExample:\n\tCscript CleanUp.js
/path:c:\\public_share /days:14 /logfile:c:\\logfile.txt
/backup:c:\\backup");
```

```
if(WScript.Arguments.length == 0){
    WScript.Echo(noArgs);
    WScript.Quit();
}
```

```
if(WScript.Arguments.Named.Exists("path")){
    args0 = WScript.Arguments.Named.Item("path"); // "d:\\test";
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
}else{
    WScript.Echo(" No folder specified for cleaning.");
    WScript.Quit(2);
}

if(WScript.Arguments.Named.Exists("days")){
    args2 = parseInt(new String(WScript.Arguments.Named.Item("days")));
    if(args2 <= 0){
        WScript.Echo("No such deleteday. Use a number above 0 (zero): " + args2);
        //WScript.Echo(noArgs);
        WScript.Quit(2);
    }
}else{
    WScript.Echo("No deleteday defined. Use /days:<number> to define a
deleteday.");
    //WScript.Echo(noArgs);
    WScript.Quit(2);
}

if(WScript.Arguments.Named.Exists("logfile")){
    logfile = WScript.Arguments.Named.Item("logfile");
}else{
    WScript.Echo("No log file specified...");
    WScript.Quit(2);
}

if(WScript.Arguments.Named.Exists("backup")){
    args1 = new String(WScript.Arguments.Named.Item("backup"));
    useBackup = true;
}else{
    useBackup = false;
}

// Instantiating objects and assigning values to variables
fso = new ActiveXObject("Scripting.FileSystemObject");
wsh = WScript.CreateObject("Wscript.Shell");

deleteDays = 1000*60*60*24*args2; // Deletedays in milli secs.

// Does the target folder exist?
if(fso.FolderExists(args0)){
    targetFolder = fso.GetFolder(args0);
}else{
    WScript.Echo("The target folder doesnt exist: "+args0);
    WScript.Quit(1);
}

// Backup active – connects to the backup folder / creates backup folder
and folder structure if nessesary...
if(useBackup == true){
    var tmp;
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
if(fso.FolderExists(args1)){
    backupFolder = fso.GetFolder(args1);

}else{
    try {
        backupFolder = fso.CreateFolder(args1);
    }catch(e){
        WScript.Echo("Could NOT create backup folder: "+args1);
        WScript.Quit(3); // >>>>> ABORTING SCRIPT >>>>>
    }
}
tmp = "xcopy " + targetFolder + " " + backupFolder + " /T /Y";
wsh.Run(tmp, 0, true);
strTargetFolder = new String(targetFolder);
strBackupFolder = new String(backupFolder);
}

// Starts logging...
var lf, logfile;
var ForReading = 1, ForWriting = 2, ForAppending = 8;
var TristateUseDefault = -2, TristateTrue = -1, TristateFalse = 0;

if(!fso.FileExists(logfile)){
    fso.CreateTextFile(logfile);
    logfile = fso.GetFile(logfile);
}
else{
    logfile = fso.GetFile(logfile);
}

lf = logfile.OpenAsTextStream(ForAppending, TristateUseDefault);
lf.WriteLine("-----");
lf.WriteLine("Starting CleanUp.js version 0.02:");
lf.WriteLine(new Date);
lf.WriteLine();
lf.WriteLine("Target path ....: "+args0);
lf.WriteLine("Deletedays ....: "+args2);
lf.WriteLine("Backup .....: "+useBackup);
lf.WriteLine("Backup folder .: "+args1);
lf.WriteLine("Logfile .....: "+logfile);
lf.WriteLine();
lf.WriteLine();

// Executes the cleanup!!!
ScanFolders(targetFolder);
DeleteFolders();

lf.WriteLine();
lf.WriteLine();
lf.WriteLine("Finished running CleanUp.js:");
lf.WriteLine("A total of "+fileCounter+" file(s) was deleted/moved.");
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
If.WriteLine("A total of "+folderCounter+" folders(s) was deleted.");
If.WriteLine(new Date);
If.WriteLine("-----");
If.Write("\n\n\n");
If.Close();

/*
 * Finds sub folders and recurse on them and executes CleanUpFolder...
 */
function ScanFolders(f){
    var fc, tmp;
    // Cleans up the current folder for old files...
    CleanUpFolder(f);
    // Deletes files marked for deletion
    DeleteFiles();
    // Is this folder empty? If not, look for subfolders...
    if(f.SubFolders.Count != 0){
        fc = new Enumerator(f.SubFolders);
        for (;!fc.atEnd(); fc.moveNext()){
            tmp = fso.GetFolder(fc.item());
            ScanFolders(tmp);
        }
    }
    // Test if folder should be deleted. If the size of the folder
    // is == 0, then the folder neither contains subfolders or files.
    if(f.Size == 0 && f != targetFolder){
        arrayFolders.push(f);
    }
}

/*
 * CleanUpFolder runs through every file in a specific folder (specified
 with an argument)
 * and either deletes or moves the files to the designated backup
 folder, that meet the deletedays criteria.
 */
function CleanUpFolder(fold){
    var f, dlm, dd, dc, tmp;
    f = new Enumerator(fold.Files);
    for (;!f.atEnd();f.moveNext())
    {
        tmp = f.item();
        // Date Last Modified
        dlm = new Date(tmp.DateLastModified);
        // Date Created
        dc = new Date(tmp.DateCreated);
        /*
         * Getting the current time at each run, to make sure that
         * files modified during script run doesnt end up being deleted
         * because of a negative comparison...
         */
    }
}
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
cd = new Date();
dd = (cd.getTime()-deleteDays);
if(dc.getTime()<dd && dlm.getTime()<dd){
    arrayFiles.push(tmp);
}
}
}

function DeleteFiles(){
    WScript.Echo("files");

    // Iterate through arrayFiles if > 0 and delete or move files.
    if(arrayFiles.length > 0){
        while(arrayFiles.length != 0){
            var tmp;
            tmp = arrayFiles.shift();
            // Either delete files or move according to usebackup setting...
            if(useBackup == true){
                var destTmp, strBackTarget;
                strBackTarget = new String(tmp);
                destTmp = strBackTarget.replace(strTargetFolder, strBackupFolder);
                // if the file already exists in the backup dir, then delete and
                move the new from the public share...
                if(fso.FileExists(destTmp)){
                    fso.DeleteFile(destTmp, true);
                    WScript.Echo(tmp+"\n\tReplaced file in backup folder.\n\tModified:
"+tmp.DateLastModified+"\n\tCreated: "+new Date(tmp.DateCreated));
                    If.WriteLine("Moving file (R): " + tmp);
                    fileCounter++;
                    fso.MoveFile(tmp, destTmp);
                }else{
                    WScript.Echo(tmp+"\n\tModified:
"+tmp.DateLastModified+"\n\tCreated: "+new Date(tmp.DateCreated));
                    If.WriteLine("Moving file .....: " + tmp);
                    fileCounter++;
                    fso.MoveFile(tmp, destTmp);
                }
            }
            else{
                If.WriteLine("Deleting file .....: " + tmp);
                fileCounter++;
                fso.DeleteFile(tmp, true);
            }
        }
    }
}

function DeleteFolders(){
    WScript.Echo("Folders");
    // Iterate through arrayFolders if > 0 and delete folders.
    if(arrayFolders.length > 0){
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
for(var i = 0; i < arrayFolders.length; i++){
    var f;
    f = arrayFolders[i];
    WScript.Echo("Folder deleted .....: " + f);
    If.WriteLine("Folder deleted .....: " + f);
    folderCounter++;
    f.Delete();
}
}
```

EOF

```
/*
 * CleanUp
 *
 * Description: This script will search a specified folder for files
 * that have not been modified/created for xx days and delete them (with/without backup).
 * It will also delete empty folders.
 *
 * Version: 0.03
 * By: Egil Hansen
 *
 * Errorcodes returned by this script:
 * 1 = The target folder does not exist.
 * 2 = Invalid input.
 * 3 = Could NOT create backup folder.
 *
 *
 * Changelog:
 *
 * Version 0.03
 * – Files and folders objects are now added to a array and deleted after the iteration of
 * target folder.
 *
 *
 */
```

```
// Declaring variables
var fso, targetFolder, backupFolder, deleteDays, cd, useBackup;
var strTargetFolder, strBackupFolder, args0, args1, args2, noArgs;
var fileCounter = 0, folderCounter = 0;
var arrayFiles = new Array();
var arrayFolders = new Array();
```

```
// Testing for required/valid arguments and assigning...
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
noArgs = new String(" CleanUp.js requires the following arguments:\n\nTarget
folder:\t\t\t/path:<folder>\nDelete days:\t\t\t/days:<number of days>\nLogfile:\t\t\t/logfile:<logfile>\nOptional
– Backup folder:\t/backup:<folder>\n\nExample:\n\tWithout backup:\tCleanUp.js /path:c:\\public_share
/days:14 /logfile:c:\\logfile.txt\n\tWith backup:\tCleanUp.js /path:c:\\public_share /days:14
/logfile:c:\\logfile.txt /backup:c:\\backup\n\nTip:\n\tUse Cscript to run the script instead of
Wscript.\nExample:\n\tCscript CleanUp.js /path:c:\\public_share /days:14 /logfile:c:\\logfile.txt
/backup:c:\\backup");

if(WScript.Arguments.length == 0){
    WScript.Echo(noArgs);
    WScript.Quit();
}

if(WScript.Arguments.Named.Exists("path")){
    args0 = WScript.Arguments.Named.Item("path"); // "d:\\test";
}else{
    WScript.Echo(" No folder specified for cleaning.");
    WScript.Quit(2);
}

if(WScript.Arguments.Named.Exists("days")){
    args2 = parseInt(new String(WScript.Arguments.Named.Item("days")));
    if(args2 <= 0){
        WScript.Echo("No such deleteday. Use a number above 0 (zero): " + args2);
        //WScript.Echo(noArgs);
        WScript.Quit(2);
    }
}else{
    WScript.Echo("No deleteday defined. Use /days:<number> to define a deleteday.");
    //WScript.Echo(noArgs);
    WScript.Quit(2);
}

if(WScript.Arguments.Named.Exists("logfile")){
    logfile = WScript.Arguments.Named.Item("logfile");
}else{
    WScript.Echo("No log file specified...");
    WScript.Quit(2);
}

if(WScript.Arguments.Named.Exists("backup")){
    args1 = new String(WScript.Arguments.Named.Item("backup"));
    useBackup = true;
}else{
    useBackup = false;
}

// Instantiating objects and assigning values to variables
fso = new ActiveXObject("Scripting.FileSystemObject");
wsh = WScript.CreateObject("Wscript.Shell");
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
deleteDays = 1000*60*60*24*args2; // Deletedays in milli secs.

// Does the target folder exist?
if(fso.FolderExists(args0)){
    targetFolder = fso.GetFolder(args0);
}else{
    WScript.Echo("The target folder doesnt exist: "+args0);
    WScript.Quit(1);
}

// Backup active – connects to the backup folder / creates backup folder and folder structure if nessesary...
if(useBackup == true){
    var tmp;
    if(fso.FolderExists(args1)){
        backupFolder = fso.GetFolder(args1);

    }else{
        try {
            backupFolder = fso.CreateFolder(args1);
        }catch(e){
            WScript.Echo("Could NOT create backup folder: "+args1);
            WScript.Quit(3); // >>>>> ABORTING SCRIPT >>>>>
        }
    }
    tmp = "xcopy " + targetFolder + " " + backupFolder + " /T /Y";
    wsh.Run(tmp, 0, true);
    strTargetFolder = new String(targetFolder);
    strBackupFolder = new String(backupFolder);
}

// Starts logging...
var lf, logfile;
var ForReading = 1, ForWriting = 2, ForAppending = 8;
var TristateUseDefault = -2, TristateTrue = -1, TristateFalse = 0;

if(!fso.FileExists(logfile)){
    fso.CreateTextFile(logfile);
    logfile = fso.GetFile(logfile);
}
else{
    logfile = fso.GetFile(logfile);
}

lf = logfile.OpenAsTextStream(ForAppending, TristateUseDefault);
lf.WriteLine("-----");
lf.WriteLine("Starting CleanUp.js version 0.02:");
lf.WriteLine(new Date);
lf.WriteLine();
lf.WriteLine("Target path ....: "+args0);
lf.WriteLine("Deletedays ....: "+args2);
lf.WriteLine("Backup .....: "+useBackup);
```

microsoft.public.scripting.wsh: Input on CleanUp script

```
If.WriteLine("Backup folder .: "+args1);
If.WriteLine("Logfile .....: "+logfile);
If.WriteLine();
If.WriteLine();

// Executes the cleanup!!!
ScanFolders(targetFolder);
DeleteFolders();

If.WriteLine();
If.WriteLine();
If.WriteLine("Finished running CleanUp.js:");
If.WriteLine("A total of "+fileCounter+" file(s) was deleted/moved.");
If.WriteLine("A total of "+folderCounter+" folders(s) was deleted.");
If.WriteLine(new Date);
If.WriteLine("-----");
If.WriteLine("\n\n\n");
If.Close();

/*
 * Finds sub folders and recurse on them and executes CleanUpFolder...
 */
function ScanFolders(f){
    var fc, tmp;
    // Cleans up the current folder for old files...
    CleanUpFolder(f);
    // Deletes files marked for deletion
    DeleteFiles();
    // Is this folder empty? If not, look for subfolders...
    if(f.SubFolders.Count != 0){
        fc = new Enumerator(f.SubFolders);
        for (;!fc.atEnd(); fc.moveNext()){
            tmp = fso.GetFolder(fc.item());
            ScanFolders(tmp);
        }
    }
    // Test if folder should be deleted. If the size of the folder
    // is == 0, then the folder neither contains subfolders or files.
    if(f.Size == 0 && f != targetFolder){
        arrayFolders.push(f);
    }
}

/*
 * CleanUpFolder runs through every file in a specific folder (specified with an argument)
 * and either deletes or moves the files to the designated backup folder, that meet the deletedays criteria.
 */
function CleanUpFolder(fold){
    var f, dlm, dd, dc, tmp;
    f = new Enumerator(fold.Files);
    for (;!f.atEnd();f.moveNext())
```

microsoft.public.scripting.wsh: Input on CleanUp script

```

{
    tmp = f.item();
    // Date Last Modified
    dlm = new Date(tmp.DateLastModified);
    // Date Created
    dc = new Date(tmp.DateCreated);
    /*
    * Getting the current time at each run, to make sure that
    * files modified during script run doesnt end up being deleted
    * because of a negative comparison...
    */
    cd = new Date();
    dd = (cd.getTime()-deleteDays);
    if(dc.getTime()<dd && dlm.getTime()<dd){
        arrayFiles.push(tmp);
    }
}
}

function DeleteFiles(){
    WScript.Echo("files");

    // Iterate through arrayFiles if > 0 and delete or move files.
    if(arrayFiles.length > 0){
        while(arrayFiles.length != 0){
            var tmp;
            tmp = arrayFiles.shift();
            // Either delete files or move according to usebackup setting...
            if(useBackup == true){
                var destTmp, strBackTarget;
                strBackTarget = new String(tmp);
                destTmp = strBackTarget.replace(strTargetFolder, strBackupFolder);
                // if the file already exists in the backup dir, then delete and move the new from the
                public share...
                if(fso.FileExists(destTmp)){
                    fso.DeleteFile(destTmp, true);
                    WScript.Echo(tmp+" Replaced file in backup folder.\n\tModified:
"+tmp.DateLastModified+"\n\tCreated: "+new Date(tmp.DateCreated));
                    If.WriteLine("Moving file (R): " + tmp);
                    fileCounter++;
                    fso.MoveFile(tmp, destTmp);
                }else{
                    WScript.Echo(tmp+"\n\tModified: "+tmp.DateLastModified+"\n\tCreated: "+new
Date(tmp.DateCreated));
                    If.WriteLine("Moving file .....: " + tmp);
                    fileCounter++;
                    fso.MoveFile(tmp, destTmp);
                }
            }
            else{
                If.WriteLine("Deleting file .....: " + tmp);
            }
        }
    }
}

```

microsoft.public.scripting.wsh: Input on CleanUp script

```
        fileCounter++;
        fso.DeleteFile(tmp, true);
    }
}
}

function DeleteFolders(){
    WScript.Echo("Folders");
    // Iterate through arrayFolders if > 0 and delete folders.
    if(arrayFolders.length > 0){
        for(var i = 0; i < arrayFolders.length; i++){
            var f;
            f = arrayFolders[i];
            WScript.Echo("Folder deleted .....: " + f);
            If.WriteLine("Folder deleted .....: " + f);
            folderCounter++;
            f.Delete();
        }
    }
}
```