

322385

# AutoCAD Scripting Extreme with VBA, Excel, and AutoCAD Core Console

Michael Best  
Principal Automation Specialist  
The Lee Company

## Learning Objectives

- Learn how to write and run script files from the desktop and within AutoCAD
- Learn how to use AutoCAD VBA to run multiple scripts on multiple files and directories
- Learn how to use Excel to process listed files and directories utilizing script files
- Learn how to use Core Console to process thousands of drawings using script files

## Description

**Definition:** A script or scripting - To write an executable section of code that automates a task. "Automating a task" is exactly how AutoCAD scripting works. Putting together several script commands or scripts, any user can set up multiple drawing standards to change hundreds to thousands of drawings in a snap using AutoCAD with Microsoft Visual Basic for Applications (VBA), Microsoft Excel with VBA, and, most importantly, Core Console. In this class, you'll learn how to use all of these tools to change, adjust, and process individual files, directories, and whole projects efficiently and effectively with little effort. All attendees will leave armed with the necessary tools to take the drudgery out of mind-numbing tasks, and filled with the potential of being carried like a hero around the office for saving hundreds of hours processing thousands of project drawings. This class will feature AutoCAD software, AutoCAD with VBA, Excel with VBA, and Core Console. Core Console is included in all industry-specific, AutoCAD-based applications.

## **Speaker**

Michael Best is the Principal Automation Engineer at The Lee Company in Essex, Connecticut. He is responsible for developing design concepts by applying sound machine design practices and is the primary technical contact ensuring design goals and objectives are robust and accomplished accurately. Michael has attended AU 12 times out of the last 19 years while also sitting on discussion panels and focus groups as well as presenting 5 classes and assisting in other classes as an aid.

He has over 30 years' experience using AutoCAD and its many renditions. In his limited spare time also programs for AutoCAD, Inventor and Windows products.

Michael can be reached at: [BestM@TheLeeCo.com](mailto:BestM@TheLeeCo.com) or [MikeBest05@comcast.net](mailto:MikeBest05@comcast.net)

## Introduction

Every release of AutoCAD comes out with new and exciting features. Some releases enhance existing features. But remaining through the releases has always been scripting. It's easy, fast and very robust. Virtually every command in AutoCAD can be utilized via scripting. Do you want to set up a drawing to a standard? Scripting! Want to create some fast shortcut commands to run at once? Scripting!! Want to simply amaze your coworkers by dragging a file into an open drawing and watch all the layers change, rename, colors adjust and update the border? Scripting!!! How about learning how to update drawings without repetitive tasks and button clicks? Scripting!!!

What if you have 10's to 1000's of drawings to update to a customer's standards? Or you have 100's of drawing from a customer that you need to bring into standard your staff can understand better? Scripting!!! Core Console is the answer to help with all those files.

AutoCAD scripting and Core Console to the rescue! Scripting uses the same commands you do over and over via picks and clicks but from basically a text file that can be inserted from the menu or "drag and drop" after a little typing trial and error first. The best part, there's no programming language to learn! **Scripting is free.** It's always been part of AutoCAD. If you know how to type any commands in AutoCAD, then you already know how to script in AutoCAD. We will cover the basics and a little more in this class on just that.

Core Console (CC) is a stripped-down version of AutoCAD without the real need for the peripherals that we use in AutoCAD. All you need is to create a small .bat file and a script file or series of script files to process all those files in a single pass. A caveat, most all computers these days run on multiple cores CC can be setup to run a series of .bat files on separate folders on separate cores.

If you know the commands, you already know scripting. This class will help you realize your hidden potential in 60 minutes. We will start with some scripting then move into some basic AutoCAD VBA. Adding Excel and incorporating Core Console to monitor and maintain the progress of files processed.

## **Learn how to reduce mouse clicks, keystrokes and all strokes in general**

I'm sure most in this class can admit the second strongest muscle in their body is their index finger.... followed by the middle finger. For right clicks and rolls of course.

Using a mouse is the most repetitive work we do on any design project or the internet. You can automate just about any process in AutoCAD. Programming in any shape or form is always challenging. It's a Love/Hate relationship.....You love it when it works and hate it when it doesn't! When it doesn't, a different approach needs to be taken or just walk away to clear one's head before trying again, reading tea leaves or sacrificing a chicken to look for answers works sometimes too. Eating a bucket of chicken would be my choice or a couple Popeyes Chicken sandwiches.

Just a few methods are listed below:

### ***Visual Basic for Applications (VBA)***

AutoCAD VBA became available in Release 14. In its initial release it was powerful but a little hard to follow if you were used to creating VBA applications for Windows Office products. It's no longer included in the AutoCAD installation. It must be downloaded from the Autodesk website and installed separately.

### ***AutoLISP/Visual LISP***

AutoLISP is an interpretive programming language that doesn't require compiling before using it in AutoCAD. It's been in AutoCAD since version 2.18 (circa 1986) and is still in use today with millions of lines of code available to download and use. Very powerful programming language in AutoCAD for manipulating and processing .dwg files.

### ***Macro's and Ribbon***

Assigning macros to a button on the ribbon is very easy but depending on the complexity can be very hard to debug. Understanding the syntax is the key to creating easy to complex button macros. Scripting can also be added to the ribbon with a custom button.

### ***Scripting***

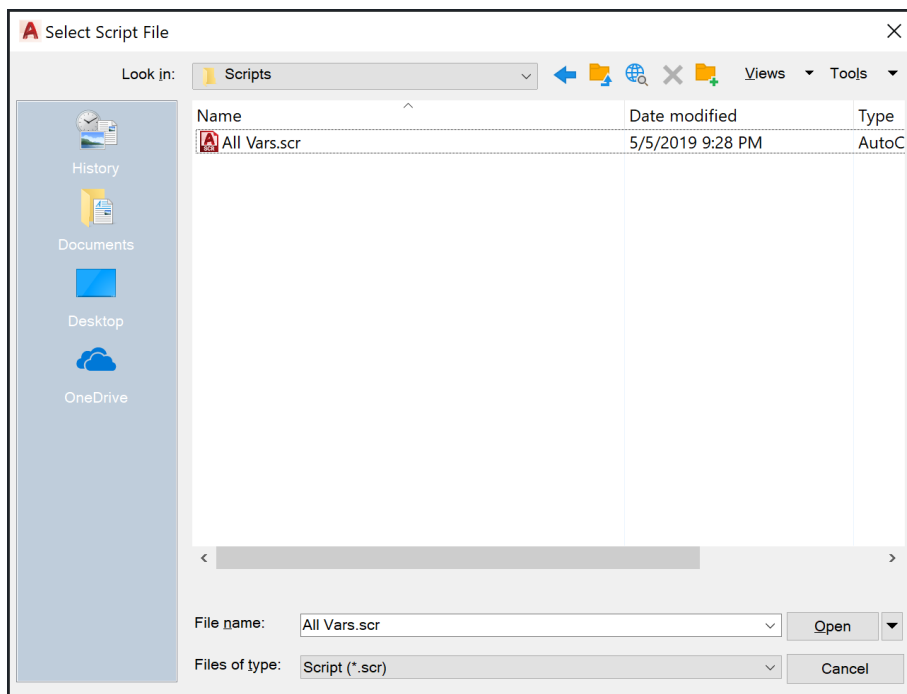
Let the command prompt be your friend! That's right, if you know the command you want to use then that little palette at the bottom of the screen that shows up all the time (annoying isn't it?) will be your best friend when scripting. What's your second best friend? The AutoCAD Text Window (or F2) of course!

## What is a script file?

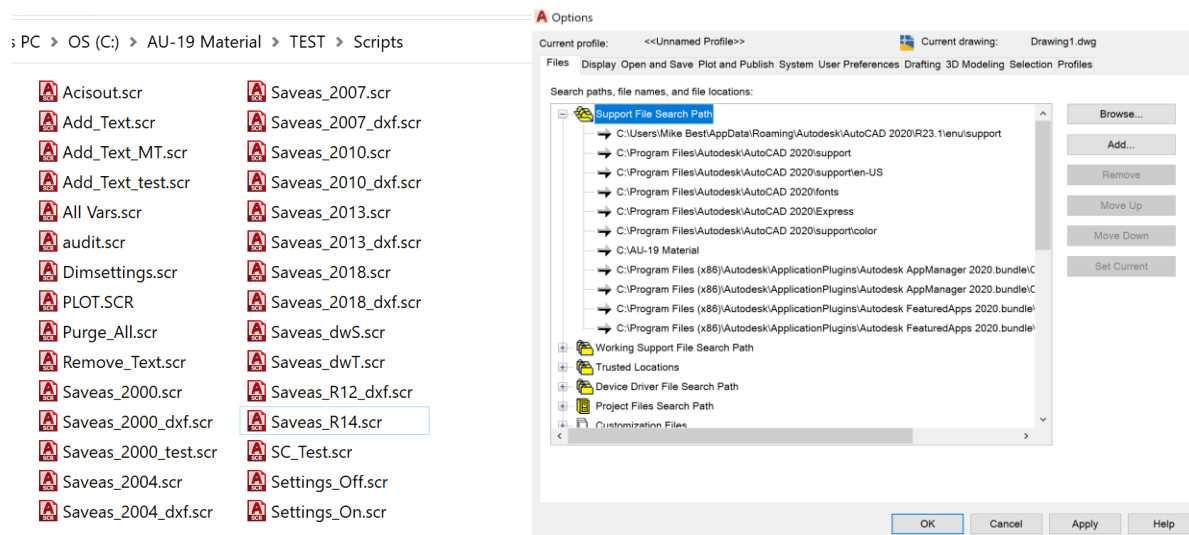
A script file is a text file with a set of commands that tells AutoCAD a series of commands to run when invoked. Script files are created and saved as .scr extension files. AutoCAD knows how to run those files either from the menu or through typing “scr” or “script” into AutoCAD. You can type it on the command line or on the screen and the context window.



When typing it in on the screen the popup appears so you can search for the script file you want to use.



But there's a much faster way to run a script file.....Much faster! Keep your favorite scripts in an accessible folder and open on your computer if you use them a lot (recommended). Also a good idea to add the location either on your hard drive or on a shared network drive to the search paths in the AutoCAD Options.



With the folder open you can drag and drop the script right into AutoCAD. Yup, it's that easy. Just make sure you're not in another command when running a script.

## Creating Script files

The best way to create a script is from typing it out first in AutoCAD. As you are typing in the commands to run, use the AutoCAD Text Window (F2) to read the results of each command and subsequent command till it's done.

The Text Window reports AutoCAD's command prompt history. Every command in the current drawing can be scrolled to and viewed. The history includes all the AutoCAD prompts, commands and system variables, default values and settings, error messages, reports, and of course everything you have literally typed or clicked from the menu or tool bar.

There are a few tools to use to create scrips. Windows Notepad or TextEdit on Mac OS, Notepad ++ (<https://notepad-plus-plus.org>) and Sublime Text (<https://www.sublimetext.com>) all work really well. All are free. For this class we will view and edit all commands in Windows Notepad.

### Headache Reducing Rules

- A space in the script is the same as hitting “Enter” except when a command allows spaces. “Enter” is evaluated at the end of each command series.

Example: *Zoom E* (the space between the “m” and “E” equates to an “enter” command.

- A semi-colon placed at the begging of a text line is not evaluated by AutoCAD. Think of adding a bit of a note to the next command just in case someone reads it and doesn’t understand the context.

Example: ;The next command will zoom to extents  
*Zoom E*

- A script file must always end with a blank line.

Example: *Zoom E*  
 (Cursor needs to be on this line. It indicates that enter was completed to allow the “Zoom Extents” to complete)

- Add a hyphen or dash (-) in front of a command name to keep the dialog box from opening. Anytime a dialog box opens, it kills the script! If unsure whether to use the hyphen or dash, type “-insert”, AutoCAD will process the entry on the command line only.

```
PURGE
Command: -PU
-PURGE
```

- When debugging a script and some changes were made, use the “Undo” command. It will undo everything the script had done in one pass. When a script is executed, the process is considered as one lump sum command.
- Turn on the variables changed at the beginning! If variables were turned off, you’ll do yourself a favor by turning them back on once the script has completed. It’s not a bad decision to do when processing a multitude of drawings either. If one happens to crash AutoCAD or stall you’ll remember it when trying to open a drawings and the dialog box doesn’t show up.....

### Scripting Commands

- **DELAY** - Provides a timed pause within a script. Specifies the duration of a pause. Entering **delay 1000** in your script delays the start of execution of the next command for about one second. The longest delay available is 32767, which is slightly less than 33 seconds.

Example: *Delay 3000* This will delay the script from continuing for 3 seconds. Good when running slides for a demo.

- **RESUME** - Continues an interrupted script. You can interrupt a script that is running by pressing Esc or Backspace. Any error encountered while processing input from a script file causes the script to be suspended. If a script is suspended while the program is active, you can use RESUME to continue the script.
- **RSCRIPT** - Repeats a script file. RSCRIPT is useful for demonstrations that repeat a script; for example, a script that must run over and over during a trade show or in a showroom.

If RSCRIPT is the last line in a script file, the file runs continuously until interrupted by Esc. This command will also work well when combined with the Delay command when running slides for a demo.


- **SCRIPT**- Executes a sequence of commands from a script file. A script is a text file with an .scr file extension. Each line of the script file contains a command that can be completed at the Command prompt.
- **SCRIPTCALL**- Executes a sequence of commands and nested scripts from a script file. With the SCRIPTCALL command, scripts can execute nested scripts as well as commands. A script is a text file with an .scr file extension. Each line of the script file contains a command that can be completed at the Command prompt, or a reference to another script file.
  - Example: Scriptcall "c:\scripts\I\_do\_it\_All.scr"

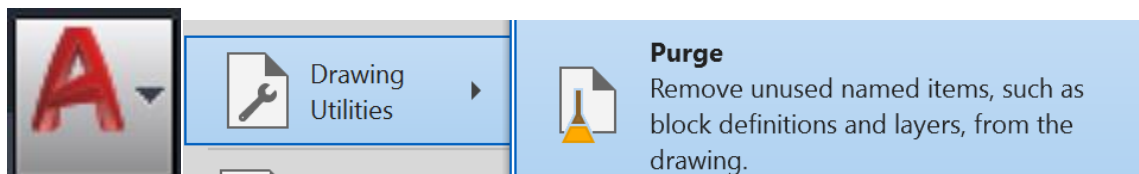
This works well when combining several scripts into one script without copying all the commands from several scripts into a large one.

## How to run script files

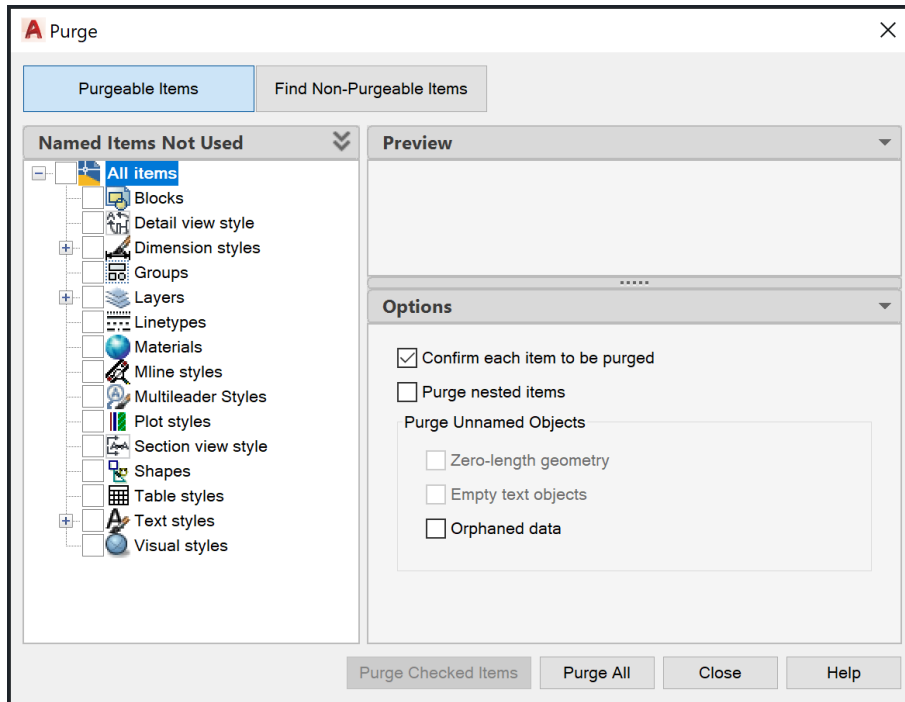
What is a script? It does one thing; it mimics what you can type in on the screen or command prompt (that thing that used to be anchored at the bottom of the screen). Type in the command, follow the prompts till the action is completed and you've basically written your first script. You want to run a command on a drawing? Instead of picking from the menu you can type the command on the AutoCAD command line or in the context window and the command with automagically begin to appear ..... or one similar. Make good choices on the one you want to run.

I like to use the Purge command as a great example when running a script after doing all the changes. You can access through the menu or typing it in at the command line.

Clicking on  AutoCAD menu and drilling down through the Drawing Utilities menu you get to the "Purge" command button.



After clicking on the button the popup menu below appears for all your choices.



Typically everyone wants to purge all items not used after finishing all the editing or just wrapping up the project and you want to put the drawing on a mini diet. Grazing down the buffet of options you finally decide to just hit the **Purge All** button and remove all the unneeded items in the drawing.

But, if you usually purge everything there is an easier way.....Using a script file.

Below are 2 illustrations using the Purge command. The first one is from typing *-purge* at the command line. The dash in front just tells AutoCAD not to use the dialog box. "All" tells the Purge command to "purge" all unused blocks, dimstyles, textstyles and various other styles that can be included in some standard drawings. The "\*" tells the command to purge only items to purge and leave the rest in the drawing. The next option will let the user verify each item purged, enter "n" so you can get a drink of coffee while the command clears the debris from your drawing.

The second illustration on the right below is the same command that's been put into a script file. Same answers just no questions. Notice the semicolon in front of the line allows AutoCAD to disregard the line. Adding these into your script allows you to put comments or notes in the script so others can understand what's going on.

```
-PURGE
Enter type of unused objects to purge [Blocks/DEta
data/All]: all
Enter name(s) to purge <*>:
Verify each name to be purged? [Yes/No] <Y>: n
No unreferenced blocks found.
No unreferenced layers found.
No unreferenced linetypes found.
Deleting text style "Annotative".
```

\*Purge\_All.SCR - Notepad

File Edit Format View Help

```
;begin purge command
-purge
ALL
*
NO
```

Spaces make a difference in scripting. The illustration below to the left shows spaces after each command request. They also mean just what happens when the spacebar or the enter button does, it acts as an enter to the command. Be careful when typing in the commands and requests. By the way, it's easier to maintain your scripting when you place each command on it's own line and follow the steps taken from the command prompt history.

\*Purge\_All.SCR - Notepad

File Edit Format View Help

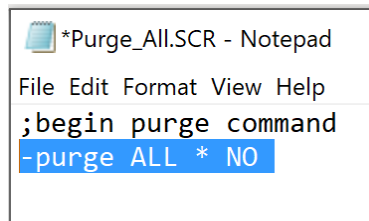
```
;begin purge command
-purge
ALL
*
NO
```

\*Purge\_All.SCR - Notepad

File Edit Format View Help

```
;begin purge command
-purge
ALL
*
NO
```

The illustration below is the same as above except it's all in the same line. It's more compact but easy to get lost in if you get lost of distractions. (Notice the space at the end in the highlight. It's the last enter to complete the Purge command.



```
*Purge_All.SCR - Notepad
File Edit Format View Help
;begin purge command
-purge ALL * NO
```

## Script Prepping

Thoughts and preparations are always the best way to begin your scripting processes. Some guess work will also get you there as well.

Before starting a script, AutoCAD dialog boxes need to be dealt with.....they need to be turned off! If they aren't they will stop any command that requires a dialog box will kill the script dead!

To begin, we need to change some variables before begin scripting. The 3 main variables I always turn off are the Attribute dialog box, File dialog box and the Command dialog box. They don't keep all dialog boxes from opening, but they do cover most.

### ***Attribute Dialog Box***

Controls whether the INSERT command uses a dialog box for attribute value entry.

### ***Command Dialog Box***

Controls the display of the In-Place Text Editor for the DIMEDIT and QLEADER commands, and the display of certain dialog boxes in AutoCAD-based products.

### ***File Dialog Box***

Suppresses display of file navigation dialog boxes such as SAVEAS and OPEN.

The following script turns off those dialog boxes. It's good to do this for good measure. Anytime a dialog box opens during a script it will cancel the script. (0 = off, 1 = on)

attdia 0

cmddia 0

filedia 0

**Quick Tip:** DWG Check – Autodesk’s Definition: Checks drawings for potential problems when opening them.

It’s just a good idea to make sure this dialog box doesn’t open up. Especially when opening drawings (.dwg’s) created from another CAD software. AutoCAD will do a quick assessment of the drawing by reading the header and inform the user that it’s not from a dwg native software.

dwgcheck 0

The following script will, turn off the dialog boxes, turn off DWG check, Audit the drawing and repair it, run Overkill, Zoom all, Purge all unused items, then turn the DWG check back on as well as the dialog boxes. It will finish up by saving the drawing then closing it.

attdia 0

cmddia 0

filedia 0

dwgcheck 0

audit y

-overkill all D

zoom all

-purge ALL \* NO

dwgcheck 1

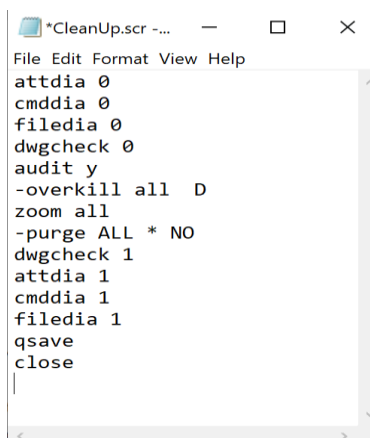
attdia 1

cmddia 1

filedia 1

qsave

close

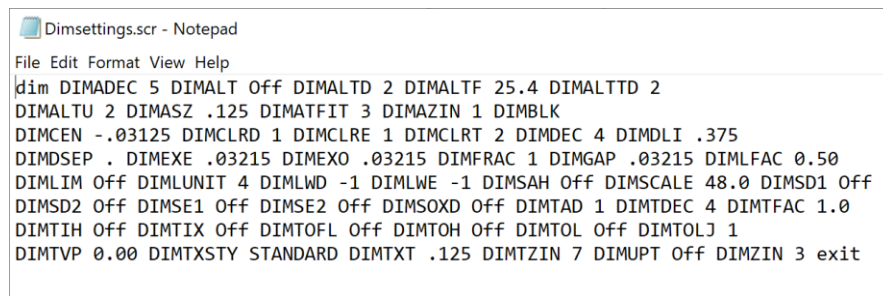


**Quick Tip:** Overkill – Autodesk’s Definition: Removes duplicate or overlapping lines, arcs, and polylines. Also, combines those that are partially overlapping or contiguous

Some 3D software’s when exporting out to 2D don’t always hide or remove lines and blocks that are on top of each other. I have seen some 2D drawings when that have duplicate lines on top each other when copied out to create detail drawings. Overkill will remove the shortest of those lines leaving the longest edge to edge, end to end or corner to corner.

Do you have dimension standards? No problem, you can use scripting to setup entire drawings in a flash.

The illustration below shows setting up dimension standards from a script file.



```

Dimsettings.scr - Notepad
File Edit Format View Help
dim DIMADEC 5 DIMALT Off DIMALTD 2 DIMALTF 25.4 DIMALTDD 2
DIMALTU 2 DIMASZ .125 DIMATFIT 3 DIMAZIN 1 DIMBLK
DIMCEN -.03125 DIMCLRD 1 DIMCLRE 1 DIMCLRT 2 DIMDEC 4 DIMDLI .375
DIMDSEP . DIMEXE .03215 DIMEXO .03215 DIMFRAC 1 DIMGAP .03215 DIMLFAC 0.50
DIMLIM Off DIMLUNIT 4 DIMLWD -1 DIMLWE -1 DIMSAH Off DIMSCALE 48.0 DIMSD1 Off
DIMSD2 Off DIMSE1 Off DIMSE2 Off DIMSOXD Off DIMTAD 1 DIMTDEC 4 DIMTFAC 1.0
DIMITH Off DIMITX Off DIMITOFL Off DIMITOH Off DIMITOL Off DIMITOLJ 1
DIMITVP 0.00 DIMITXSTY STANDARD DIMITXT .125 DIMITZIN 7 DIMUPT Off DIMZIN 3 exit
  
```

Scripting is just doing what you already know how to do in your head and place it in a “list” file. The more you use AutoCAD and pay attention to the command line the faster you can work. Creating scripts to simplify the work and speed up the process just makes you look that much smarter in the end.

## AutoCAD VBA and Scripting

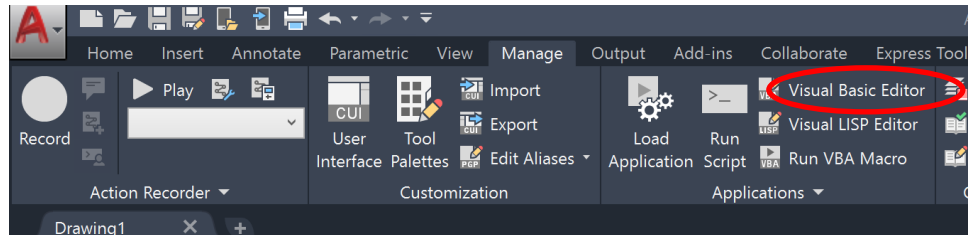
AutoCAD VBA was added in R14. It became more robust in AutoCAD 2000. Since then it’s been enhanced and sculpted into what it is today; a viable tool to manipulate AutoCAD. For this class you will see limited coding to do what we need for it to run scripts for us.

Those who are new to VBA will need to learn a few basics:

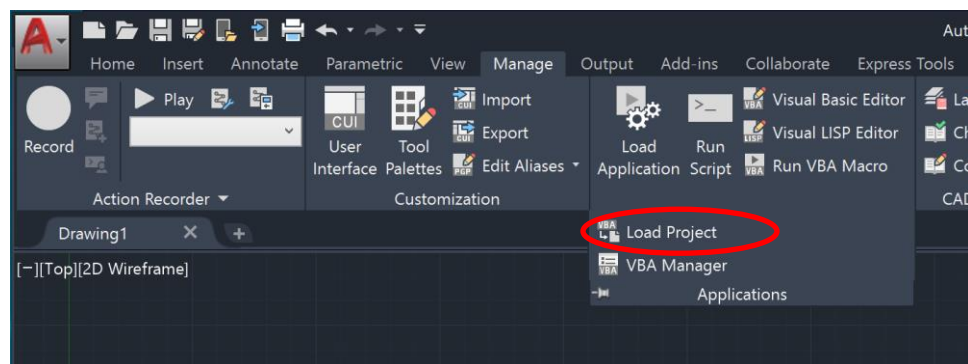
1. Setting up the interface
2. Figuring out the steps to use on the interface, i.e. the sequence
3. Then putting the code together to make it work

This isn’t going to be a coding class for VBA but what will be shown is how to use minimal coding to create an interface to run a script on a directory of files. Short and simple. From there the imagination is the users.

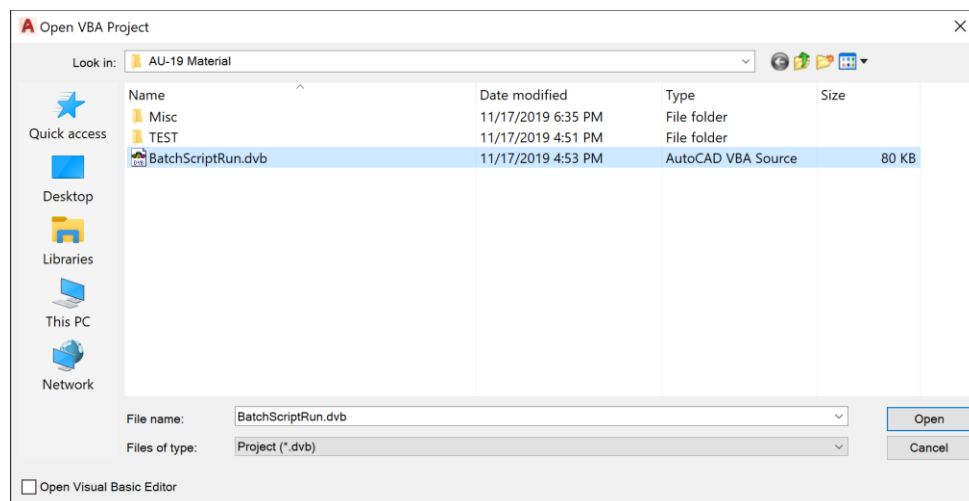
Typically, you would need to go to the “Manage” ribbon then select “Visual Basic Editor” to begin creating the interface and starting the coding for your new project to change the cadding world. For this class I’ve already done that for you.



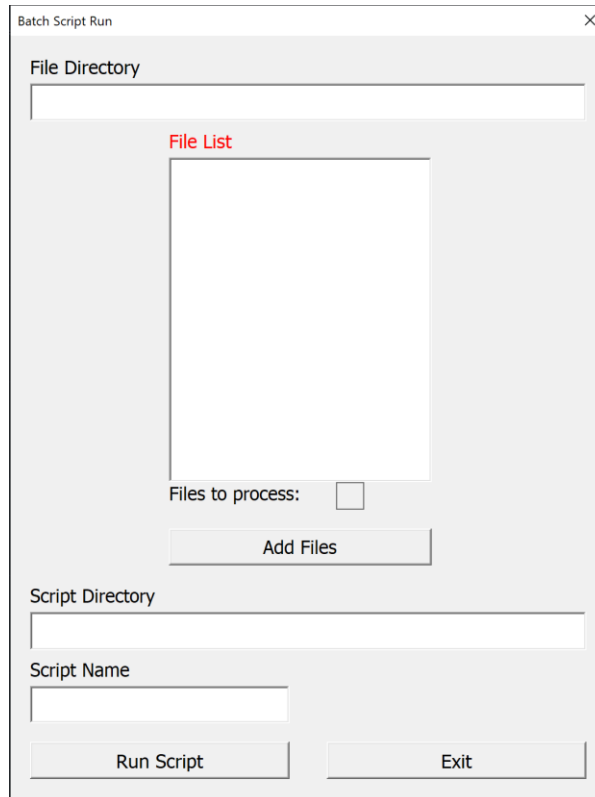
For this class I’ve created a VBA file (for AutoCAD the extension is .dwb) called BatchScriptRun.dwb. In that file it contains the interface (Userform) and modules containing the code to run the project. But to get there you need to load the .dwb first with the “Load Project” command shown in the illustration below.



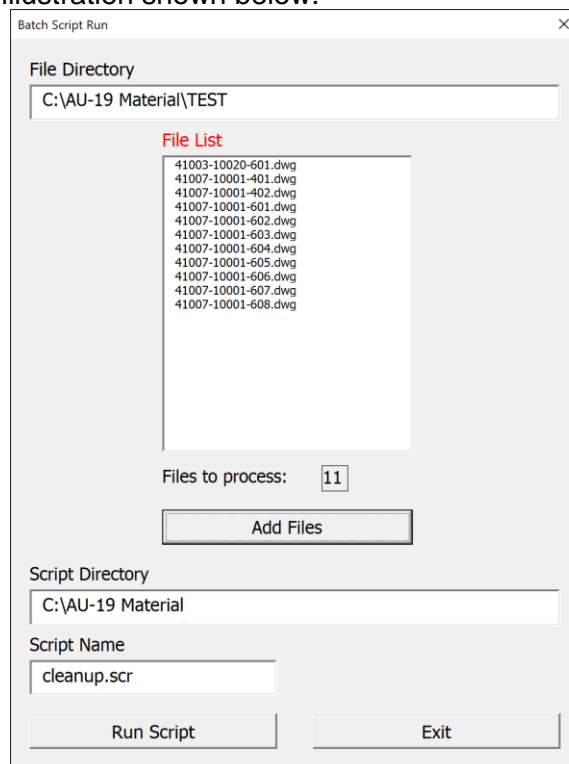
The dialog box below will open, then drill down to the directory where you stored the file uploaded with this handout and the presentation. But, we will walk through all the aspects that went into the creation of this simple VBA project.



The UserForm shown below (Batch Script Run) is what was created for this class. It's simple, not a lot of flash but it gets the job done. The only controls added to this form are: 3 text boxes (File Directory, Script Directory and Script Name), 1 Filelist box and 3 Command Buttons (Add Files, Run Script and Exit)

A screenshot of a Windows UserForm titled 'Batch Script Run'. The form has a light gray background and a standard Windows window border with a close button (X) in the top right corner. It contains several controls: a text box labeled 'File Directory' at the top; a large empty rectangular area labeled 'File List' in red text; a checkbox labeled 'Files to process:' below the file list; a button labeled 'Add Files' below the checkbox; a text box labeled 'Script Directory' below the 'Add Files' button; a text box labeled 'Script Name' below the 'Script Directory' box; a button labeled 'Run Script' at the bottom left; and a button labeled 'Exit' at the bottom right.

The File Directory text box just tells the program where to get the AutoCAD files to load into the File List box to run. The File List box is just that, a list of all the files in the directory for AutoCAD to process. The Script Directory is a text box to tell the program where to find the script to run and the Script Name text box will be the script that will run on the files. Once filled out it should look like the illustration shown below.



Batch Script Run

File Directory  
C:\AU-19 Material\TEST

File List

- 41003-10020-601.dwg
- 41007-10001-401.dwg
- 41007-10001-402.dwg
- 41007-10001-601.dwg
- 41007-10001-602.dwg
- 41007-10001-603.dwg
- 41007-10001-604.dwg
- 41007-10001-605.dwg
- 41007-10001-606.dwg
- 41007-10001-607.dwg
- 41007-10001-608.dwg

Files to process: 11

Add Files

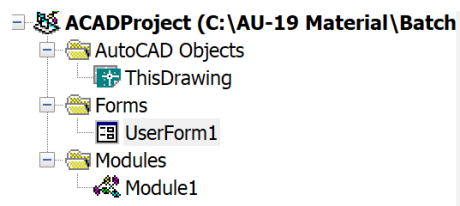
Script Directory  
C:\AU-19 Material

Script Name  
cleanup.scr

Run Script Exit

I added a file count label below the File List box just so I know how many files I was loading.

Below is the simple structure I used for the project. Just a User form and a Module.



Below is all the coding that went into this program to batch run 1 script on a whole directory of AutoCAD files.

This module simply loads the user form shown previously

```

C:\AU-19 Material\BatchScriptRun.dvb - Module1 (Code)
(General)
BatchScript
Sub BatchScript()
  UserForm1.show
End Sub

```

This is all the coding needed to run the buttons on the user form and fill in the File List Box. Notice I commented (in green) before each command and what was happening before and during each sub command was running. This is a good technique to maintain in future programming to know where you were at when you stopped and if you're forgetful, to remember what you were trying to do in the first place!

```

C:\AU-19 Material\BatchScriptRun.dvb - UserForm1 (Code)
(General)
Option Explicit
'Add File directory  DONE
'Open single file  DONE
'Add script directory  DONE
'Add script to run  DONE
'Process file  DONE
'Misc.
'Add counter?  DONE
'Global Variables
Dim ScriptName As String
Dim ScriptDir As String
Dim ScrAddress As String
Dim FileLoc As String

Private Sub CmdAddFiles_Click()
  'declare variables
  Dim fileList() As String
  Dim fName As String
  Dim dirname As String
  Dim ScriptName As String
  Dim i As Integer

  dirname = TBFile_Loc.Text
  'if root directory do nothing

  If TBFile_Loc.Text = "" Then
    MsgBox "No file directory listed"
    Exit Sub
  End If

  'build a list of the files
  fName = Dir(dirname & "\*.*")
  While fName <> ""
    'add fName to the list
    i = i + 1
    ReDim Preserve fileList(1 To i)
    fileList(i) = fName
    'get next filename
    fName = Dir()
  Wend
  'list file count in label
  Label5.Caption = i

  'see if any files were found
  If i = 0 Then
    MsgBox "No files found"
    Exit Sub
  End If

  FileBox1.Clear
  'cycle through the list and add to listbox
  For i = 1 To UBound(fileList)
    FileBox1.AddItem fileList(i)
  Next
End Sub

```

```

C:\AU-19 Material\BatchScriptRun.dvb - UserForm1 (Code)
(General)
Private Sub CmdRun_Script_Click()
  Dim adoc As AcadDocument
  Dim D
  Dim T

  'turn off the file open dialog box
  ThisDrawing.SetVariable "FILEDIA", 0
  'set AutoCAD to only open one drawing at a time
  ThisDrawing.SetVariable "SDI", 0
  For D = 0 To FileBox1.ListCount - 1
    If FileBox1.List(D, 0) <> "" Then
      T = TBFile_Loc.Text & "\" & FileBox1.List(D, 0)
      Set adoc = Application.Documents.Open(T, False)
      adoc.Activate
      'once the next file opens we need to make sure the dialog box doesn't open again.
      ThisDrawing.SetVariable "FILEDIA", 0
      'turn off saving backups
      ThisDrawing.SetVariable "isavebak", 0

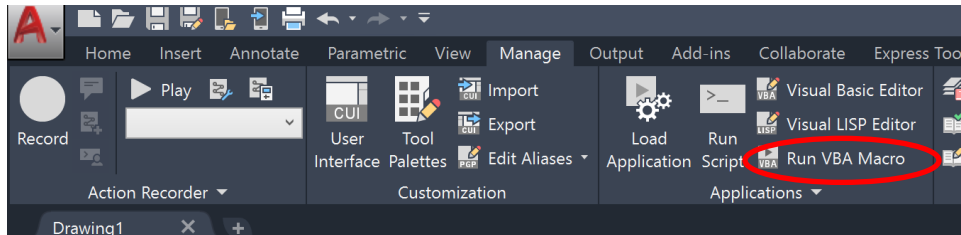
      'Get the name of the file to work on
      ScriptDir = TBScript_Loc.Text
      ScriptName = TBScript.Text
      ScrAddress = TBScript_Loc & "\" & TBScript.Text
      'do some script run/run here
      ThisDrawing.SendCommand "script" & vbCrLf & ScrAddress & vbCrLf

      'save file
      adoc.Save
      'close doc
      adoc.Close
    End If
  Next D

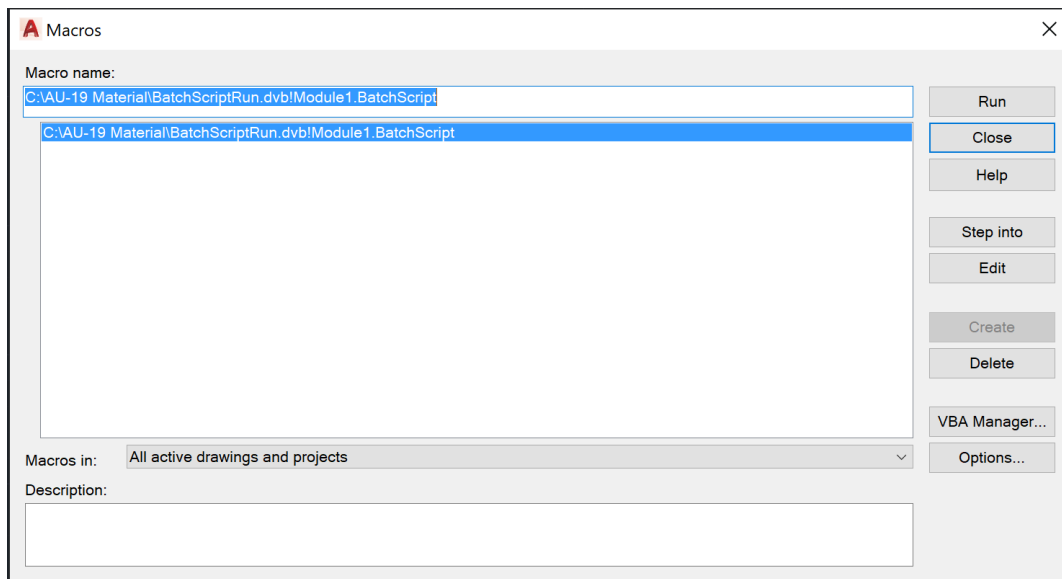
  'turn on the file open dialog box
  ThisDrawing.SetVariable "FILEDIA", 1
  'set AutoCAD to open multiple drawings again
  ThisDrawing.SetVariable "SDI", 1
  'turn off saving backups
  ThisDrawing.SetVariable "isavebak", 0
End Sub

```

To run the VBA file you need to click on the “Run VBA Macro” shown below. Depending on what you named your VBA project after it’s been created and loaded shown in the illustration above.



The macro run dialog box below will open so you can run the macro.



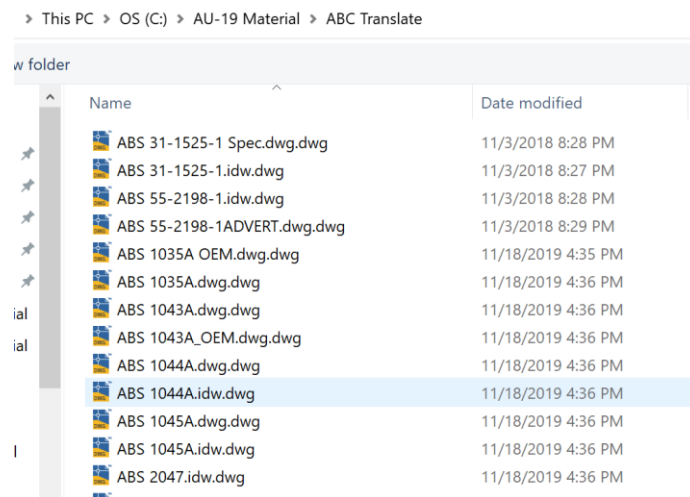
## MS Excel is Good!

Microsoft Excel has been one of those programs that just seems to morph into any need with any application. AutoCAD is no different when it comes to interacting with MS Excel. From Excel you can push or force AutoCAD to run certain commands and record the return info into Excel.

This program will work a lot like what we did in AutoCAD; gather the files selected via the “Process Files” button, select the files and list them in Excel. As the files run through Core Console, Excel will look for when each file has finished updating then change the status from “Pending” to “Complete”.

	A	B	C	D	E	F	G	H
9			Process Files					
10	FILENAME	STATUS						
1	ABS 1035A OEM.dwg.dwg	Pending						
2	ABS 1035A.dwg.dwg	Pending						
3	ABS 31-1525-1 Spec.dwg.dwg	Pending		Scripts to run				
4	ABS 31-1525-1.idw.dwg	Pending		cleanup_inside.scr				
5	ABS 55-2198-1.idw.dwg	Pending		zoom_all.scr				
6	ABS 55-2198-1ADVERT.dwg.dwg	Pending		ChangeLayers.scr				
7				PurgeAll.scr				
8								

File open and select dialog box shown below.



Starting out simple in Excel is a little bit of a test of endurance and patience! Especially if you are not an expert to begin with. But once you get going it does get much easier to handle. The power of Excel can allow users to greatly expand the use of AutoCAD translations to more options.

Options examples just to name a few:

1. Attributes mapping
2. Layer mapping
3. Color mapping
4. Block mapping and altering
5. BOM extracting/creation
6. Script writing

## **Core Console = AWESOMENESS!**

### Introduction

Who: Who can use it

When: When can I use it

What: What is Core Console?

Where: Where is it?

Who: Anyone can use Core Console (CC) to batch run drawings. All you need is 1 seat of AutoCAD to muscle through thousands of drawings at an amazing speed. Designers/engineers spend weeks, sometimes months completing drawings either in 2D or 3D design project then have to groan over converting them to a customer's standard at delivery. It's the painful end to an otherwise, hopefully successful project. And when converting those drawings they are never to the exact standard you or your customer were hoping for!



You could spend weeks opening and changing 1000's of drawings exported to any type of standard one at a time. I've done it before while working at another company some 20 years ago. I was given the job of taking nearly 1000 files exported from another 2D software into our AutoCAD standard.

When: Any CAD junkie or flunky who knows how to type in the commands can write scripts to do pretty much the same thing as those custom programs using some ingenuity.

When to use Core Console is best used when "a bunch" of drawings all need to be processed. That can include just updating previous projects to current status with new borders or the new company logo. The limits are a little endless in the processes that can be used with CC.

What: What is Core Console (accorreconsole.exe)? Think of it as AutoCAD in silent mode. Nothing on the monitor, no need for a mouse, keyboard and since the screen will be empty, no need for a monitor either! When I was explaining what CC was to a long time Autodesk associate and how it worked, his response was "It's headless!" Indeed, that is one way of looking at it. But with scripting, it still has intelligence to do what is required in a short period of time.

Where: How headless is it? Just look at the executable sizes of the programs below!  
(They are found here: C:\Program Files\Autodesk\AutoCAD 2020)

 acad.exe	7/2/2019 1:16 AM	Application	5,670 KB
 accoreconsole.exe	7/2/2019 1:06 AM	Application	877 KB

Processing batches of drawings without the overhead just makes things so much faster! Now, to add to the fun factor, you can do Core Console batches on separate CPU cores. That's right! Each batch run on separate folders starts Core Console on a separate CPU core as well. We'll go through that another time.

```
FOR %%f IN ("%~dp0*.dwg") DO "C:\Program Files\Autodesk\AutoCAD
2020\accoreconsole.exe" /i "%%f" /s "%~dp0cleanup.scr" /l en-US
```

%%f = Parameter for processing a file

IN = In the current folder loop

("%~dp0\*.dwg") = Will open every file one at a time with the extension .dwg

DO "C:\Program Files\Autodesk\AutoCAD 2020\accoreconsole.exe" = This command in the batch file will open Core Console to open the file.

/i "%%f" = Mutes status messages and opens files one at a time in the currently directory.

"%~dp0ProcessAll.scr" = This command tells AutoCAD what program to call once it opens.

/l en-US = This part, the last part, tells AutoCAD which language to use when open.

Double clicking the AcCoreConsole.exe shows the commands necessary to run the program from a batch file.

In the usage nomenclature in the illustration below it tells you a few options that can be run with Core Console: You'll notice that it follows what is shown above as far as opening the drawings and which script to run on each drawing.

```
Usage:
AcCoreConsole.exe [/i <input dwg>] /s <script>[/product <product>] [/l <language>] [/isolate <userid> <userDataFolder>]
[/readonly] [/p[rofile] <profile>]
```

Below is what the DOS window looks like when Core Console is running on your computer. This particular run is opening .dxf files and saving them as .dwg's in 1.4 seconds.

```

C:\WINDOWS\system32\cmd.exe
Command:
Command:
Command:
Command: +saveas
Input save format [dwg/dwt/dws/dxf/Other]: other
Input save format in an integer value: 0
Enter file format [R14(LT98&LT97)/2000(LT2000)/2004(LT2004)/2007(LT2007)/2010(LT2010)/2013(LT2013)/2018(LT2018)/Standard
s/DXF/Template] <2018>: 2018(lt2018)
Save drawing as <C:\AU-19 Material\Folder-8_20k\AD76727_S02.dwg>:
Command: ..quit

C:\AU-19 Material\Folder-8_20k>"C:\Program Files\Autodesk\AutoCAD 2020\accoreconsole.exe" /i "C:\AU-19 Material\Folder-8
_20k\AD76731_S01.dxf" /s "C:\AU-19 Material\Folder-8_20k\Saveas_2018.scr" /l en-US
Redirect stdout (file: C:\Temp2\acc406410).
AcCoreConsole: StdOutConsoleMode: processed-output: enabled,auto
AutoCAD Core Engine Console - Copyright 2019 Autodesk, Inc. All rights reserved. (Q.104.0.0)

Execution Path:
C:\Program Files\Autodesk\AutoCAD 2020\accoreconsole.exe

Version Number: Q.104.0.0 (UNICODE)
Regenerating model.

**** System Variable Changed ****
2 of the monitored system variables have changed from the preferred value. Use SYSVARMONITOR command to view changes.
Command:
Command:
Command:
Command: +saveas
Input save format [dwg/dwt/dws/dxf/Other]: other
Input save format in an integer value: 0
Enter file format [R14(LT98&LT97)/2000(LT2000)/2004(LT2004)/2007(LT2007)/2010(LT2010)/2013(LT2013)/2018(LT2018)/Standard
s/DXF/Template] <2018>: 2018(lt2018)
Save drawing as <C:\AU-19 Material\Folder-8_20k\AD76731_S01.dwg>:
Command: ..quit

C:\AU-19 Material\Folder-8_20k>"C:\Program Files\Autodesk\AutoCAD 2020\accoreconsole.exe" /i "C:\AU-19 Material\Folder-8
_20k\AD76731_S02.dxf" /s "C:\AU-19 Material\Folder-8_20k\Saveas_2018.scr" /l en-US
Redirect stdout (file: C:\Temp2\acc41922421).

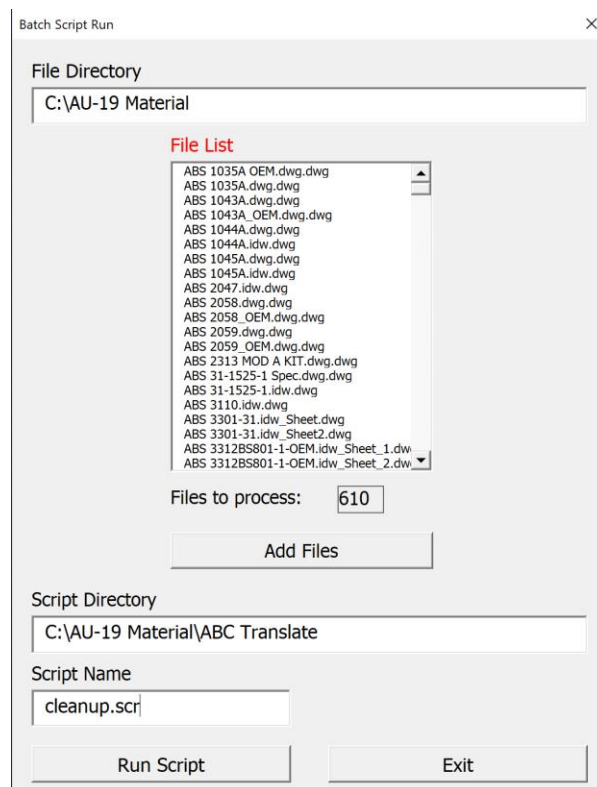
```

**Important Note:** Don't close the DOS window unless you intend to stop the file processing!

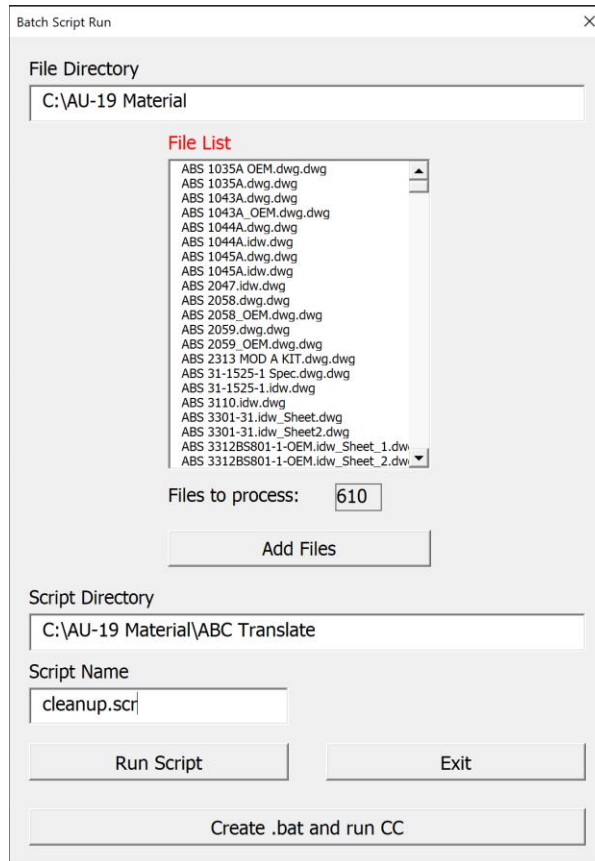
## Remember the AutoCAD VBA Program earlier?

With some additional coding I turned it into interface to write the .bat file and tell it where to find the script file to run on a whole directory. It also starts the .bat file and closes the VBA program so you can continue to work in AutoCAD while the files are being processed.

Below is the program we started with.



Below are some subtle changes to the form simply by double clicking on the form.



Batch Script Run

File Directory  
C:\AU-19 Material

File List

- ABS 1035A OEM.dwg.dwg
- ABS 1035A.dwg.dwg
- ABS 1043A.dwg.dwg
- ABS 1043A\_OEM.dwg.dwg
- ABS 1044A.dwg.dwg
- ABS 1044A.idw.dwg
- ABS 1045A.dwg.dwg
- ABS 1045A.idw.dwg
- ABS 2047.idw.dwg
- ABS 2058.dwg.dwg
- ABS 2058\_OEM.dwg.dwg
- ABS 2059.dwg.dwg
- ABS 2059\_OEM.dwg.dwg
- ABS 2313 MOD A KIT.dwg.dwg
- ABS 31-1525-1 Spec.dwg.dwg
- ABS 31-1525-1.idw.dwg
- ABS 3110.idw.dwg
- ABS 3301-31.idw\_Sheet.dwg
- ABS 3301-31.idw\_Sheet2.dwg
- ABS 3312BS801-1-OEM.idw\_Sheet\_1.dwg
- ABS 3312BS801-1-OEM.idw\_Sheet\_2.dwg

Files to process: 610

Add Files

Script Directory  
C:\AU-19 Material\ABC Translate

Script Name  
cleanup.scr

Run Script Exit

Create .bat and run CC

Notice the “Create .bat and run CC” button added to the button. Clicking this button will create a .bat file called “File\_Processor.bat” and place it in the folder where the files originate. It will insert the location identified where the script file is from the screen inputs as well.

Once the program is started with the “Create .bar.....” button it will close the VBA program because there is no longer a handshake with the .bat file.

Below is what the new code looks like added to the program

```
Private Sub CmdCC_Click()
CreatBatFile
End Sub

Sub CreatBatFile()
ScriptDir = TBScript_Loc.Text
ScriptName = TBScript.Text
ScrAddress = TBScript_Loc & "\" & TBScript.Text
Dim X As String
Dim Txtstring As String
Dim BatFile As String

'Create location to place the .bat file
BatFile = TBScript_Loc & "\File_Processor.bat"
'create the " for the .bat call routine
X = Chr(34)
'create the .bat file for the file location
Open BatFile For Output As #1
'write to place in the .bat file to run
Txtstring = "FOR %%f IN ("%~dp0*.dwg") DO ""C:\Program Files\Autodesk\AutoCAD 2020\accoreconsole.exe"" /i ""%%f"" /s " & X & ScrAddress & X & " /l en-US"
'print the string in the .bat file
Print #1, Txtstring
'close the bat file
Close #1

'shell out to run the bat file on the files
Shell BatFile, vbNormalFocus
End Sub

Private Sub UserForm_Click()
If UserForm1.Height <= 516 Then
UserForm1.Height = 560

ElseIf UserForm1.Height <= 560 Then
UserForm1.Height = 516
End If
End Sub
```

The first section starts with the “Create .bat....” button then jumps to the sub to get the file locations and set them to strings to handle in the .bat file. “Open BatFile ....” Creates and new file in the directory and saves it as “File\_Processor.bat” file. “Txtstring” assembles all the information to put in the file noting which type of files to open, start accoreconsole, then which script to run from the program screen and which language to use with AutoCAD.

Then it will close and run on its own in the background only noticeable by viewing the DOS popup screen.

## Conclusion

Depending on your skill level with AutoCAD, the knowledge of the tools within or the desire to learn more about what you can do to enhance your skills. This presentation will help you get started in that direction. At the core of this class is scripting and Core Console. I joke that only the old timers still know about and how to use scripting. Maybe true, maybe not but always increasing your knowledge and speed is never bad.

As mentioned, please don't hesitate to reach out to me at [BestM@TheLeeCo.com](mailto:BestM@TheLeeCo.com) or [MikeBest05@comcast.net](mailto:MikeBest05@comcast.net).