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

Michael Best

Principal Automation Specialist

AUTODESK. UNIVERSITY

About the speaker

Michael Best

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 or MikeBest05@comcast.net

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.

Just to Start

- AutoCAD has always had the ability to allow customization
- Slogging through
- In the meantime, we get bored
- AutoCAD scripting and Core Console to the rescue!
- Scripting is free.
- Using Core Console (CC) allows you to cruise through 100’s to 1000’s of drawings at speeds you can’t get by hand.
- Core Console 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 .bat file and a script file or series of script files to process all those files in a single pass.

Key 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

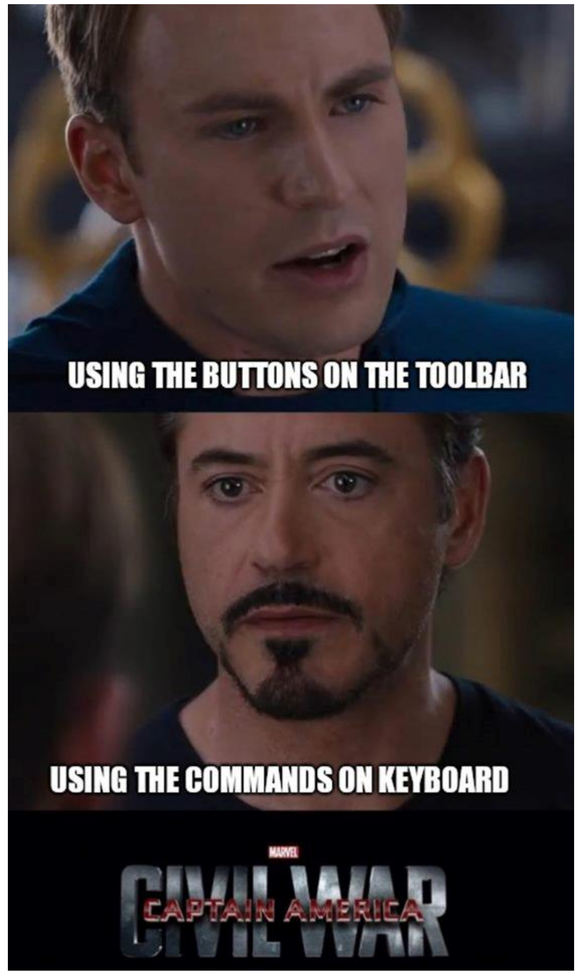
Introduction

Ways to mass process AutoCAD drawings

- Saveas 1 at a time
- Post processing
- .bat files
- VBA
- Scripting
- AutoLISP/VisualLisp

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

- Mouse clicks can be repetitive and tedious
- Automation can be a great tool
- Scripting and or VBA are awesome tools
- When they work
- When they don’t
- What next?
- Take a “walkabout”
- Consult a therapist
- Sacrifice a chicken…….or stand in line at Popeyes for a chicken sandwich



Methods to Reduce Clicks

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.

Methods to Reduce Clicks

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

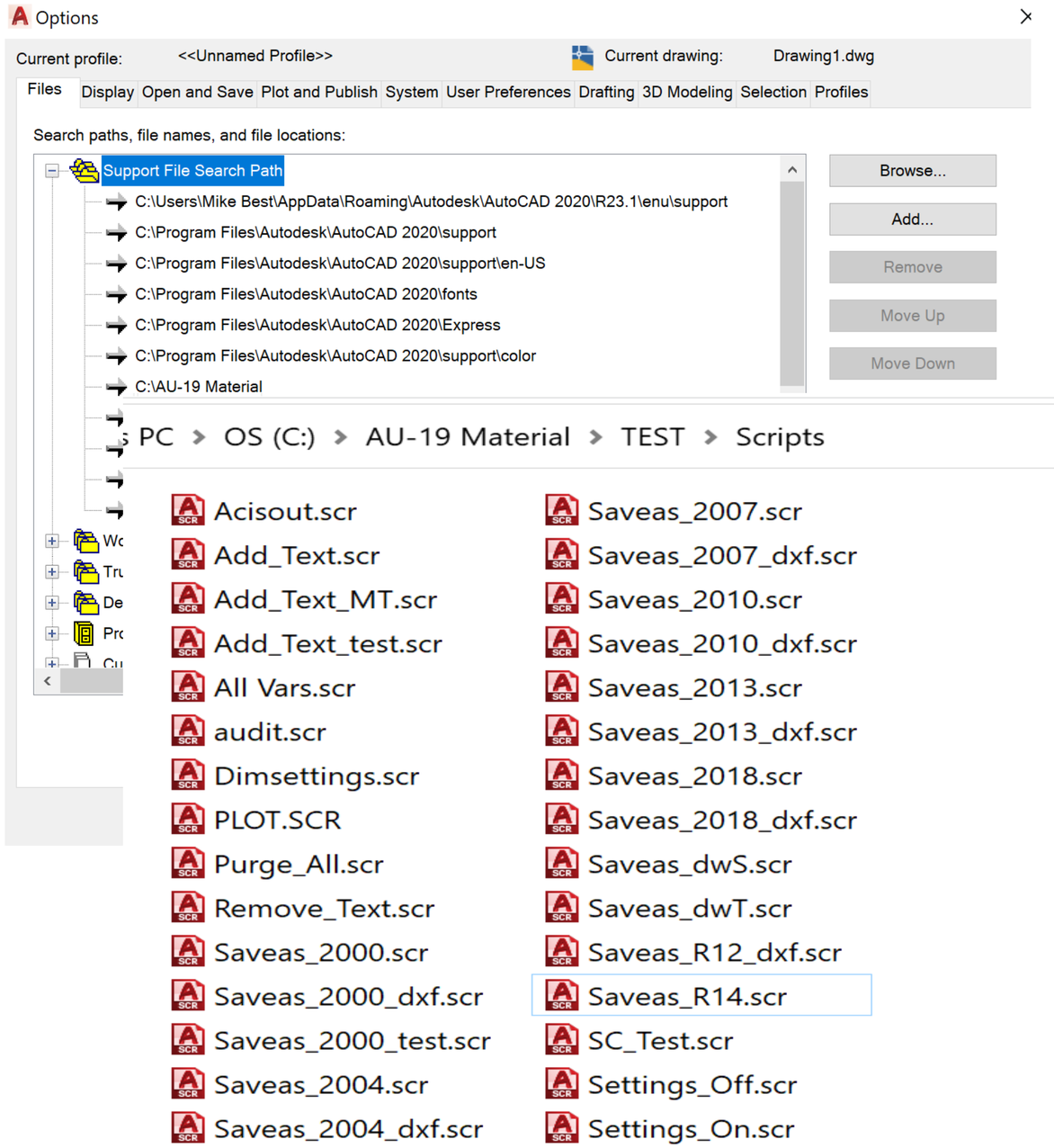
Let the command prompt be your friend! That's right, if you know the command you want to use then that little palette at the bottom of the screen that shows up all the time (annoying isn't it?) will be your best friend when scripting. What's your second best friend? The AutoCAD Text Window (or F2) of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

First Some Housekeeping

- Keep your script files in a central location on your hard drive, the network or a shared network drive for other users.
- Also, add the location in the search path in the options to make the directory more accessible.



Creating Script Files

- Sketch out what commands you want run
- Determine the order of commands to run
- Type the commands out in AutoCAD first
  - Understand the sequence of questions from AutoCAD for each command
  - Outline each answer and spacebar or enter command They Matter!!
- Use the Text Window (F2) to track your entries
- Type out the commands and entries in a text file

- Microsoft NotePad is a tool for this.

Creating Script Files

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

Creating Script Files

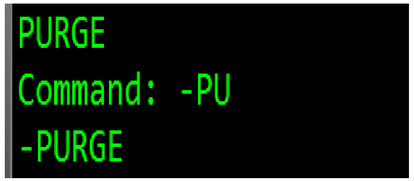
Headache Reducing Rules (cont.)

- 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.



Creating Script Files

Headache Reducing Rules (cont.)

- 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…..

Creating Script Files

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.

Creating Script Files

Scripting Commands (cont.)

- ***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

- A script does one thing, it mimics what you can type on the screen or on the command line

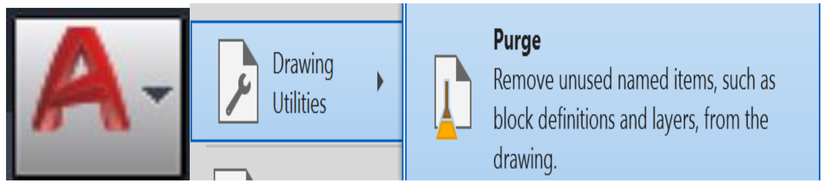
  (That thing that used to be anchored at the bottom of the screen)
- Type in the command and follow the prompts

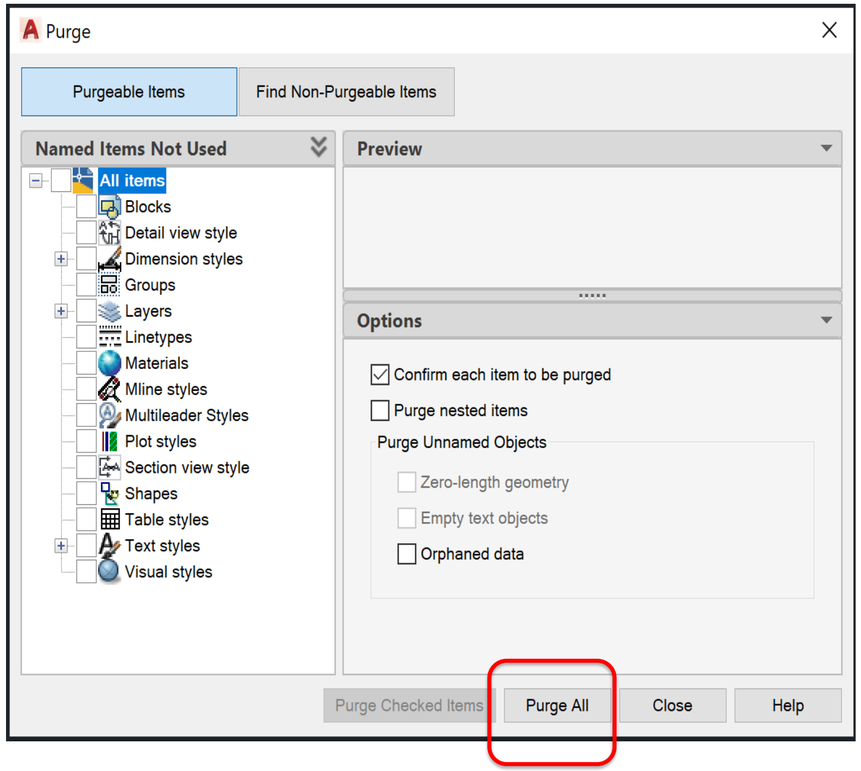
- I like to use the Purge command as a great example when running a script after doing multiple changes to a drawing.

How to run Script Files

- I like to use the “Purge” command as a great example when running a script after doing multiple changes to a drawing.

- Clicking on the  AutoCAD menu and drilling down through the Drawing Utilities menu you get the “Purge” command button.
- After clicking on the button the popup menu appears below for all your choices. (Many new choices in 2020)
- For this class, we will always use “Purge All”



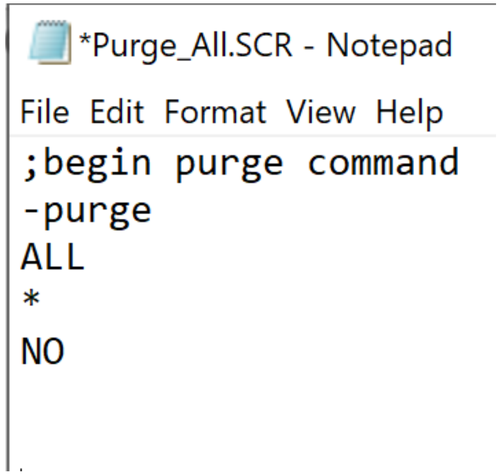


How to run Script Files

```
-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".
```

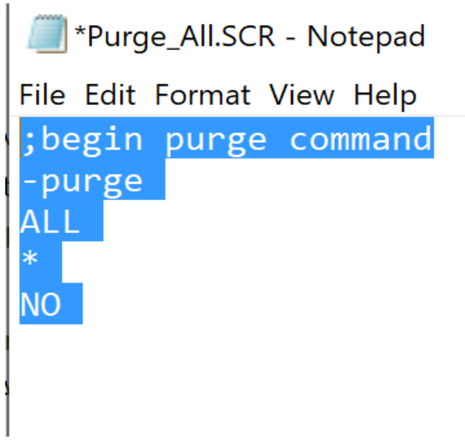
- But there’s an easier way to handle this…….Use a script file!
- But first we need to plan our use
- Open the Text Window (F2)
- Type “-PU” or “-Purge” on the screen or command line
  - Remember, the “-” dash in front disregards the dialog box.
- The first question wants to know what you want to purge “all” is my choice
- Enter names to purge. If looking to only purge on block, layer, linetype etc. then enter that name. We will use * for all.
- Do you want to verify names to purge? NO!
- Hit enter

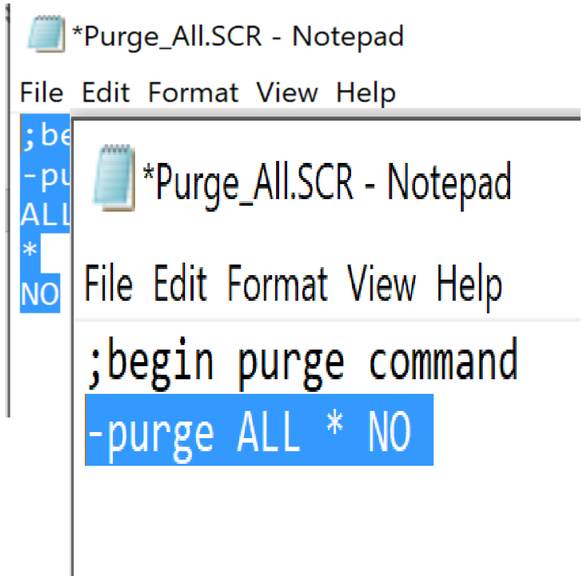
How to run Script Files



- A script file would look like this

  (The semi colon allows you to put notes in the script file





- Important to remember are the spaces used when creating the script file. I used Notepad to create and edit to create this file.
- The script file on the left won't work because there is a space after each command.
- The script file on the right does work because the next line acts as the return or enter for the next command.

Script Prep

- Knowing the commands you want to run is one thing, but what if there's a dialog box that pops up?
  - Dialog boxes Kill a script!
- The 3 main dialog boxes need to be turned off when running a script
  - Attribute Dialog Box
  - Command Dialog Box
  - File Dialog Box
- Through scripting we can turn them off before we start changing the world in the drawing. The following commands will shut them off: 0 = off, 1=on.
  - Attdia 0
  - Cmddia 0
  - Filedia 0

Script Prep

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

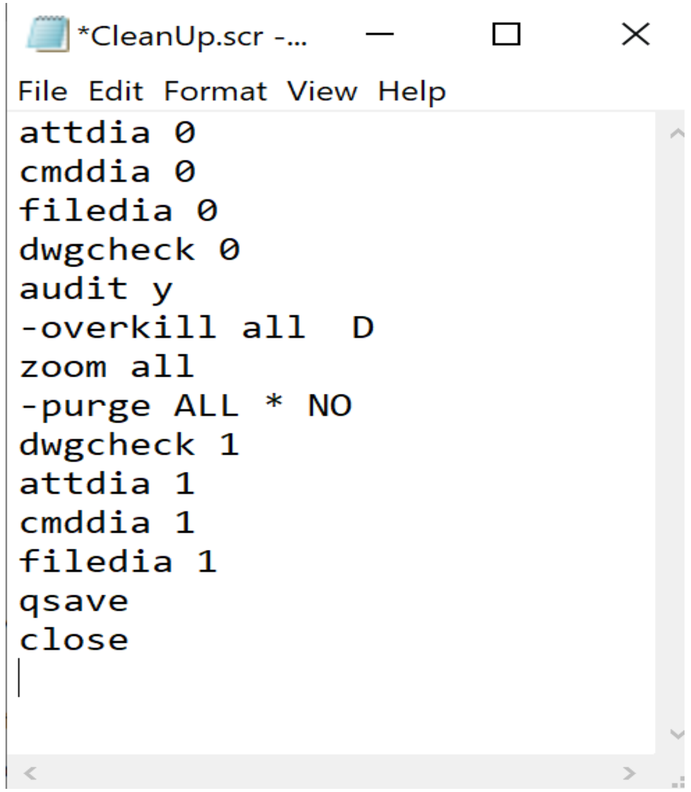
attdia 1

cmddia 1

filedia 1

qsave

close

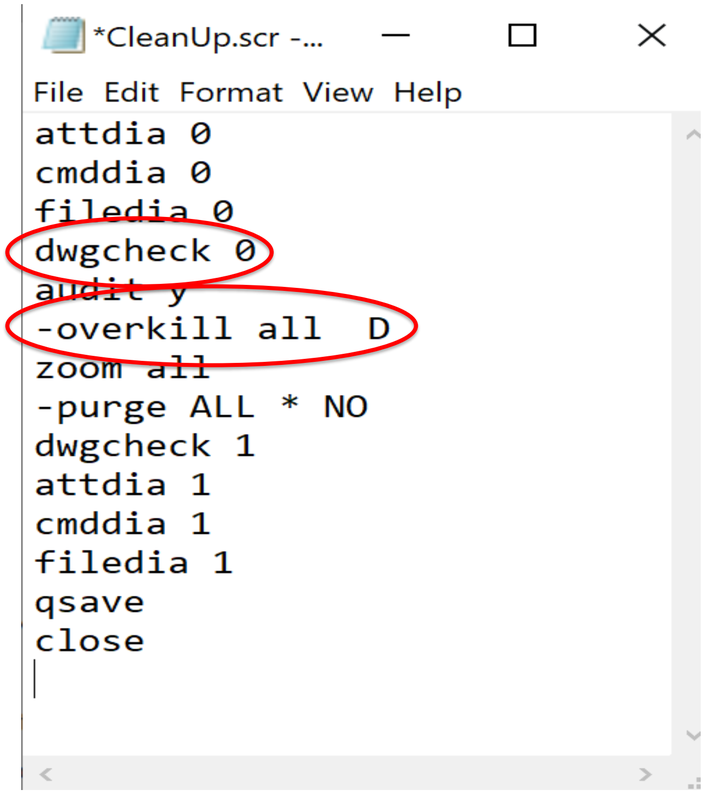


Script Prep

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.

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



Script Prep Wrap Up

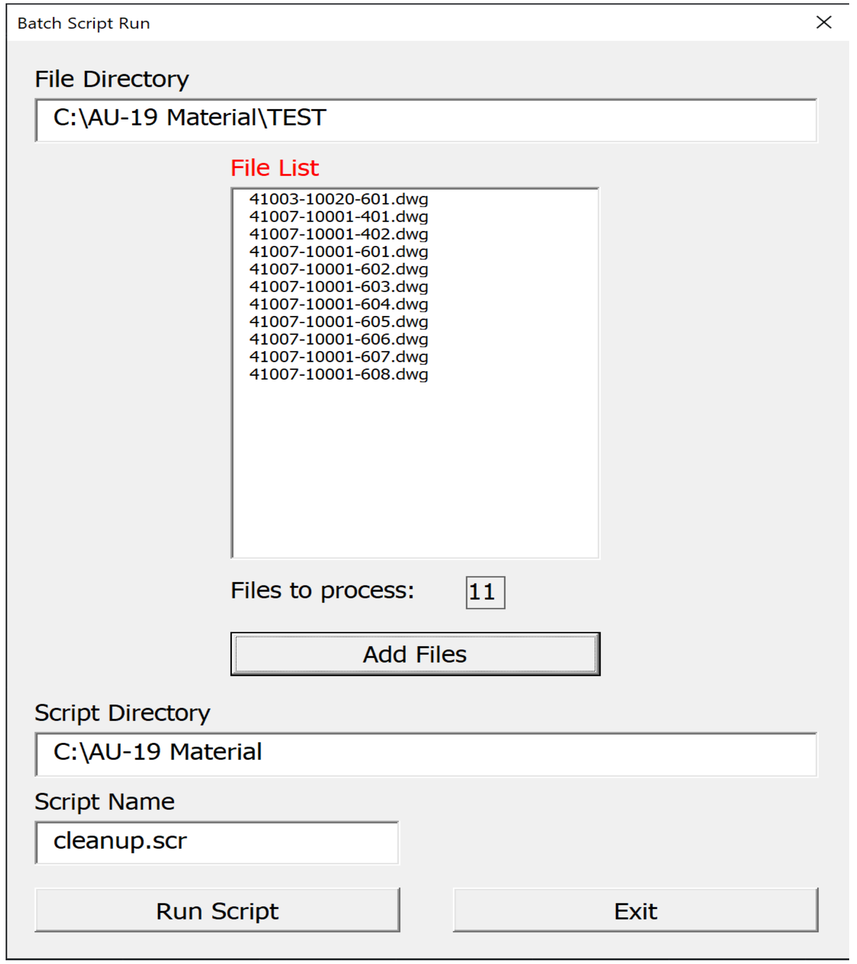
- Do you have dimension standards?
- Do you want to enforce drawing standards?
- Do you know the next command you’re going to do while doing the current one?
- Do you know the next 5 commands after the current one?
- Scripting can help you with that!
- It’s a list of things to do, a laundry list of things
- 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

- VBA was added to R14
- It became more robust in AutoCAD 2000
- Enhanced and sculpted into what it is today; a viable tool to manipulate AutoCAD
- For this class we will use limited coding for ease of use and to do what we need to run a script

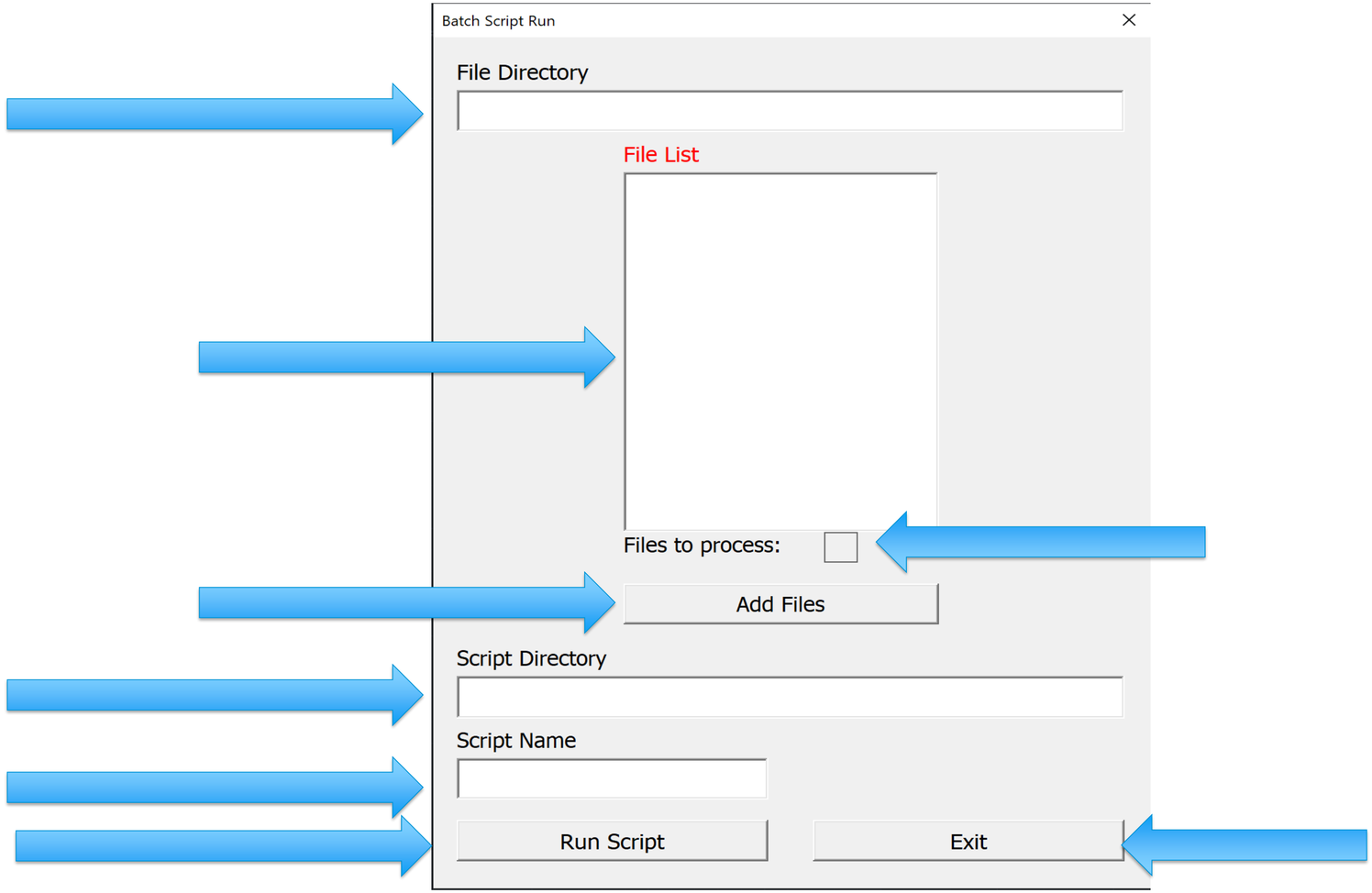
AutoCAD VBA and Scripting

- Using VBA to run scripting files is not new
- Its just not considered when programming VBA due to the power in VBA to run AutoCAD.
- For this class I created minimal coding to load a directory of dwgs to run a script on……or a series of scripts when using scriptcall inside the script to run other scripts as well.



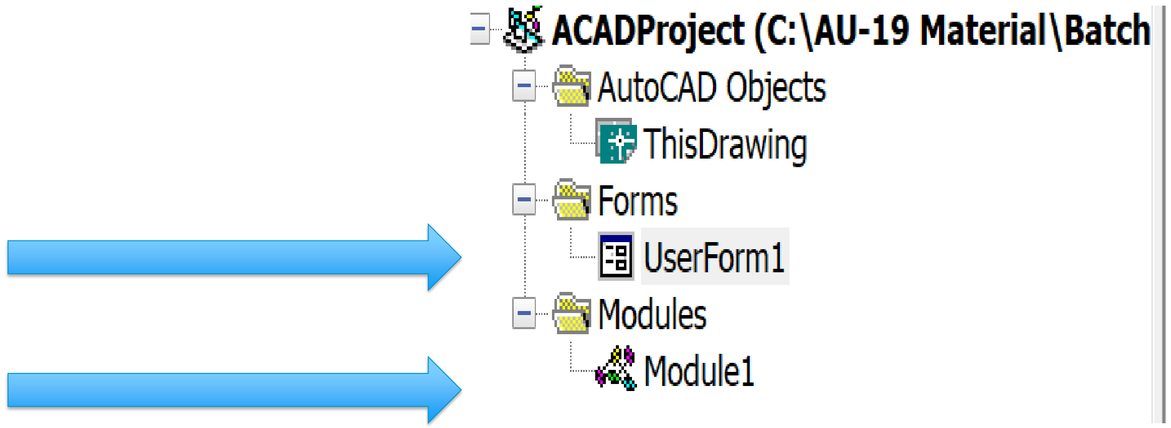
AutoCAD VBA and Scripting

- This uses 3 textboxes
  - File Directory
  - Script Directory
  - Script Name (to run)
- 1 filelist to show files to process
- 3 command buttons
  - Add Files
  - Run Script
  - Exit
- I threw in a file counter for fun

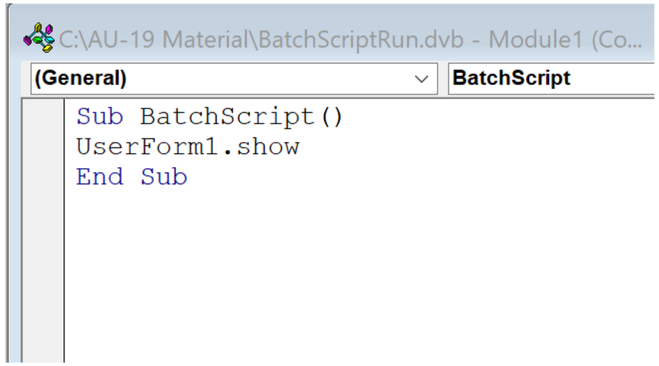


AutoCAD VBA and Scripting

- There is only 1 form
- And only 1 module



- This simple module loads the interface (UserForm1)



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 & "\" + "*.dwg")
    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) (Declarati

```
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/fun here
      ThisDrawing.SendCommand "script" & vbCr & ScrAddress & vbCr

      '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
```

I mentioned the coding was simple! This is all needed to run a script on a whole list of drawings!

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!

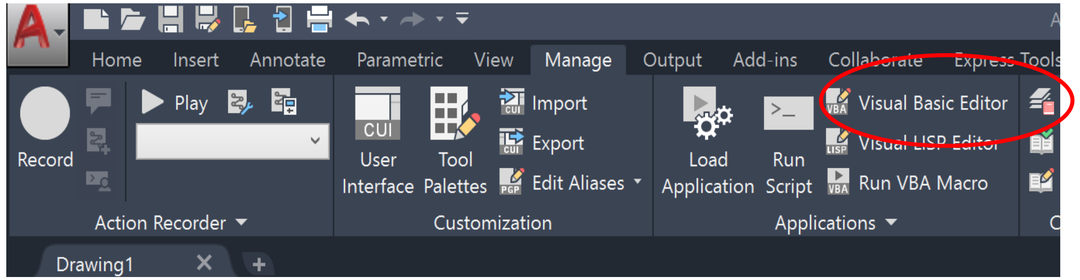
AutoCAD VBA and Scripting

To begin, go to the “Manage” menu then open the “Visual Basic Editor

From there you can begin to create the interface I showed in the previous slide……..or download the .dvb file I uploaded called BatchScriptRun.dvb. No additional references need to run the program.

If you do run into anything let me know at:

mikebest05@comcast.net

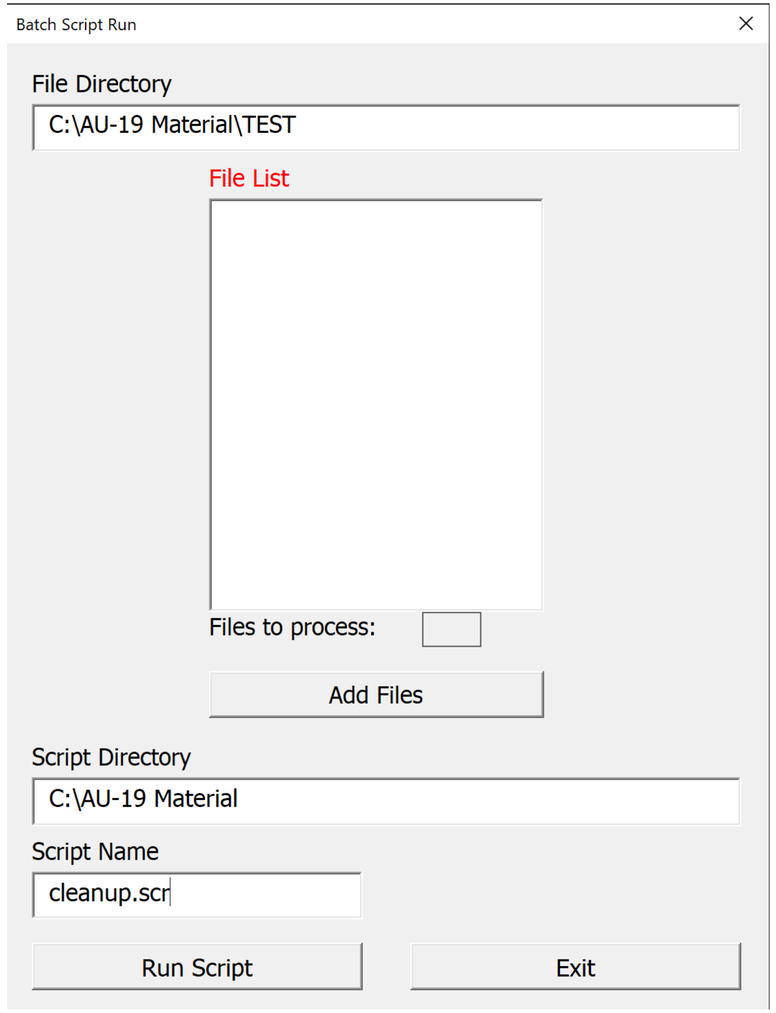


AutoCAD VBA and Scripting

When I mentioned the program was simple, just how simple is it?

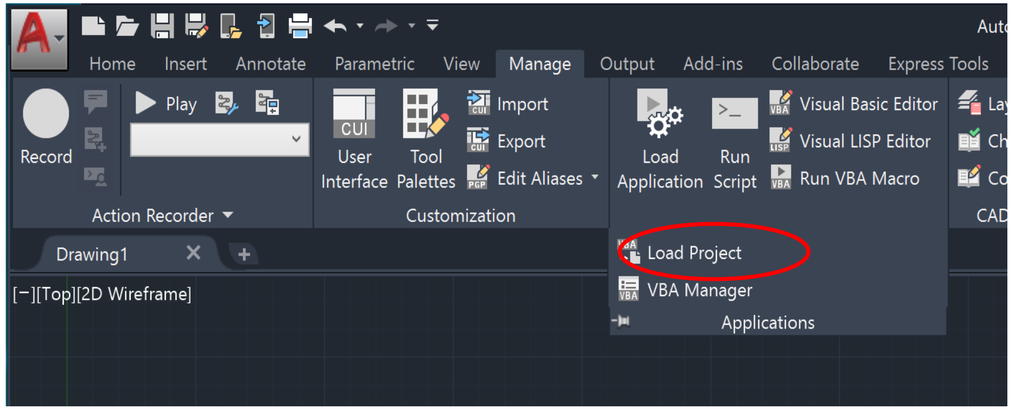
The textboxes are simply locations the program will use to know where the files are to run, the script file is to run and what the name of the script file is.

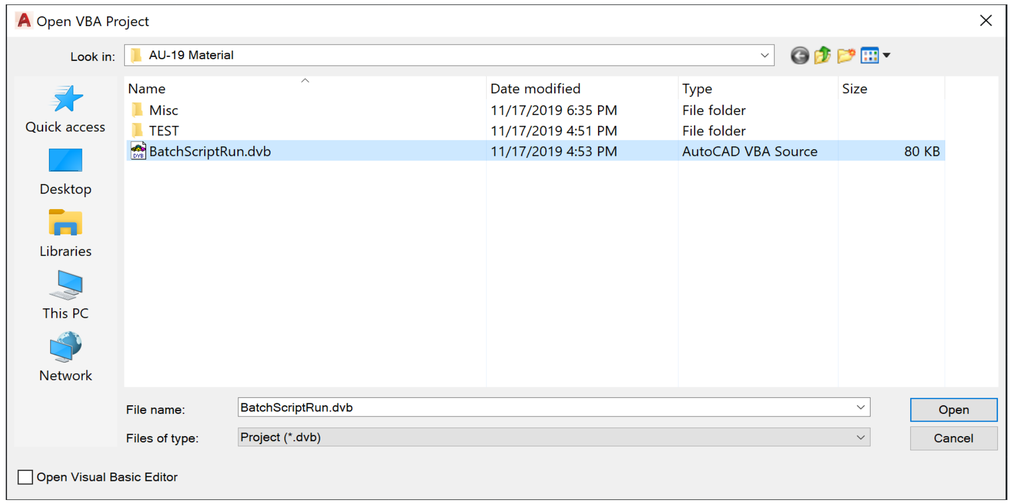
Type in or copy in the directory names and the script names. There is no search or dialog box to select the locations.



AutoCAD VBA and Scripting

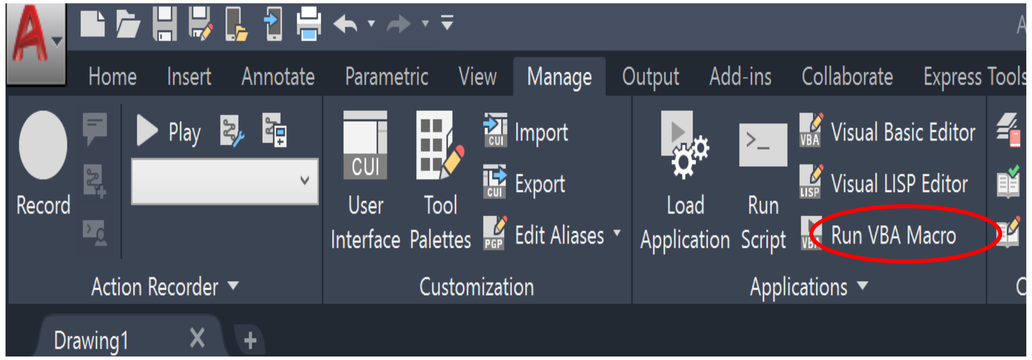
- After downloading the .dvb program created for the class we need to load it first into AutoCAD.
- Go to the “Manage” menu, click on the down arrow next to Applications, then click on “Load Project”
  - Go to the directory where you stored the .dvb and select it.

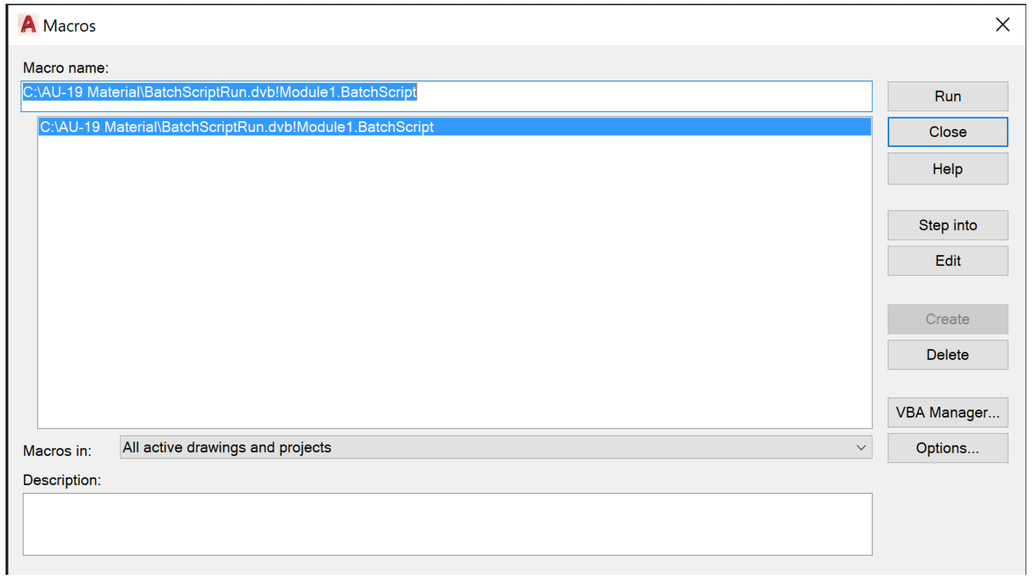




AutoCAD VBA and Scripting

- Now that the program is loaded into AutoCAD, let run the macro to check it out!
- Click on “Run VBA Macro” the select the macro in the dialog box to run. Click on “Run”
- Now you’re ready to run a script on a directory of files!



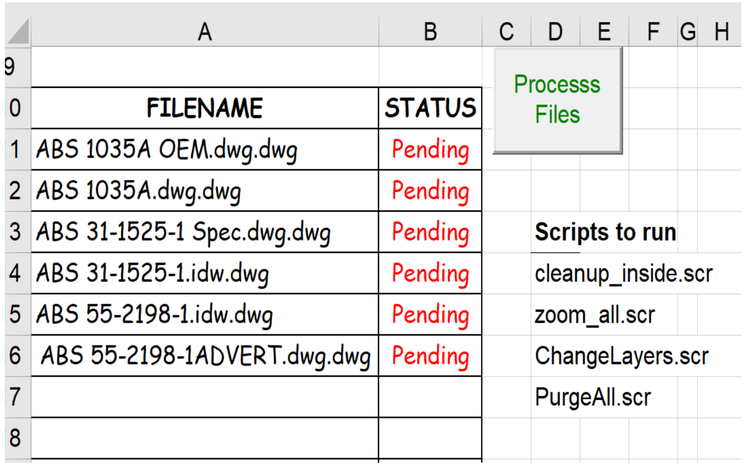


Excel and AutoCAD

- Excel is a dynamic spreadsheet program that runs with just about every distributed program out today
- AutoCAD works well with Excel
- Excel allows you to push or force AutoCAD to run certain commands and record the return info into Excel itself
- The force is strong with the 2 of them

Excel and AutoCAD

- 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:
- Attributes mapping
- Layer mapping
- Color mapping
- Block mapping and altering
- BOM extracting/creation
- Script writing



Core Console = AWESOMENESS

Introduction

- Core Console was introduced in AutoCAD 2013.
- It allows access to over 600 AutoCAD commands
- Access is via a .bat file
- Can process a drawing as fast 1.3 seconds
- Unitizes script files for all changes
- Any CAD Junky or flunky can run it

Core Console = AWESOMENESS

- Who: Who can use it
- When: When can I use it
- What: What is Core Console?
- Where: Where is it?

Core Console = AWESOMENESS

Who

- Anyone can use it.
- Only need 1 seat of AutoCAD

When

- When there’s a need to process 100’s to 1000’s of drawings
- Doesn’t tie up a seat license

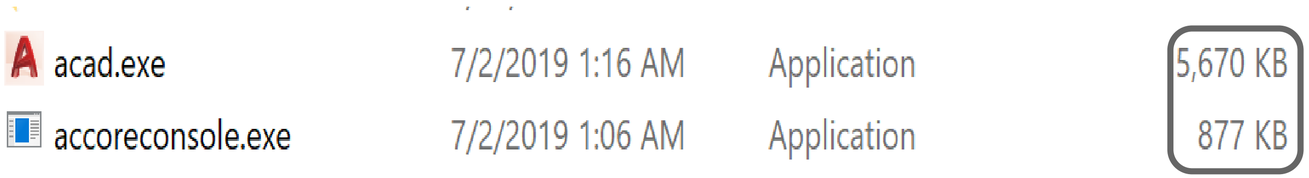
Core Console = AWESOMENESS

What

- What is Core Console?
  - 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!
  - It’s Headless!

Where

- How is it Headless?
- Let’s review the .exe program sizes



Core Console = AWESOMENESS

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.

Below is a typical batch file (.bat) to run Core Console

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

Core Console = AWESOMENESS

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

Definitions for above:

%%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.

Core Console = AWESOMENESS

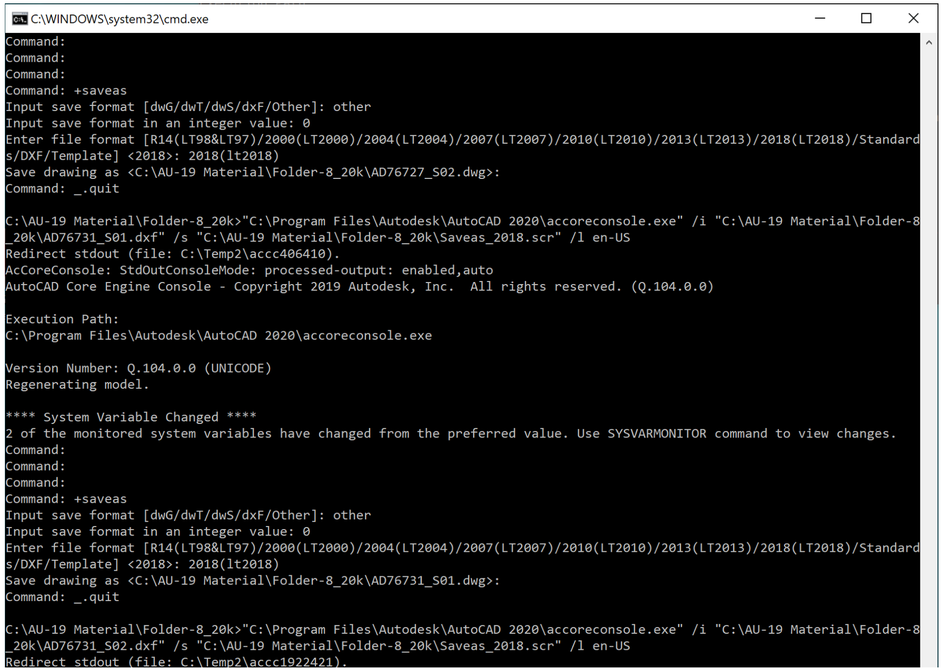
Double clicking the AcCoreConsole.exe shows the commands necessary to run the program from a batch file.

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>]
```

Core Console = AWESOMENESS

This 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.



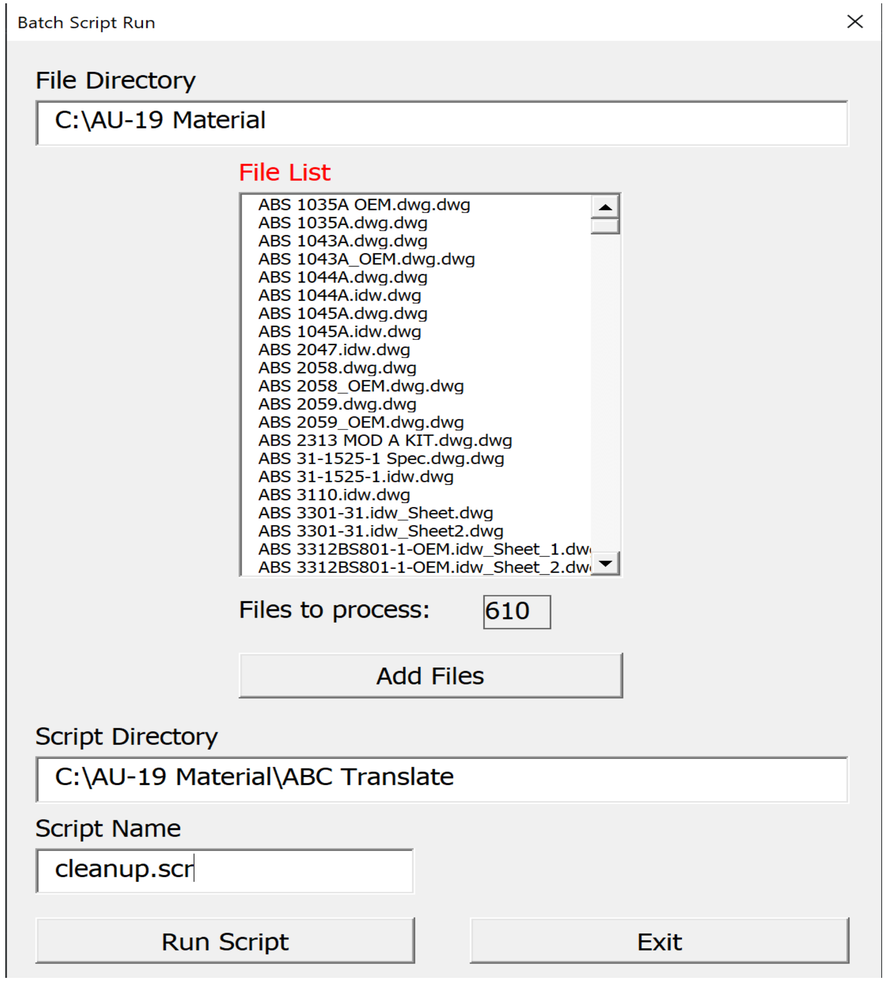
Important Notes:

- Don't close the DOS window unless you intend to stop the file processing!
- Keep spaces to a minimum in file names. The "~" in the processing string will work with most, but more than 1 space will cause Core Console to skip the file.
  - Example: ABC1234 T R2.dwg

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.

This is the program we started with.



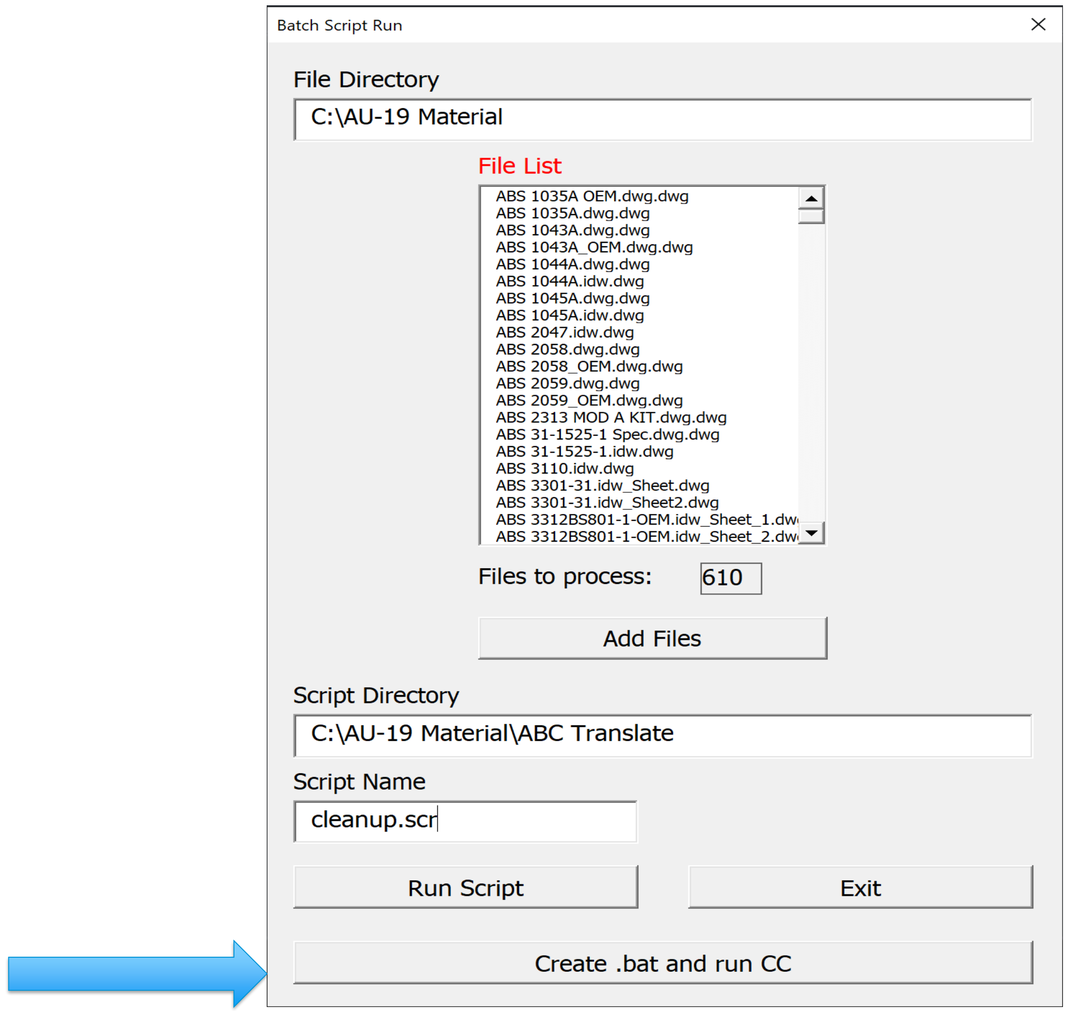
Remember the AutoCAD VBA Program Earlier?

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

I added the “Create .bat and run CC” button to the form.

After filling the “File Directory”, “Script Directory” and “Script Name” info; 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.



Remember the AutoCAD VBA Program Earlier?

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 = TBFile_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
        'string 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
```

Remember the AutoCAD VBA Program Earlier?

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 us with AutoCAD.

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

No matter what type of work you do, try to learn something every day. Learning is living

Michael Best

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

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 or MikeBest05@comcast.net.

AUTODESK®

Make anything™