



November 30 – December 3, 2004 ♦ Las Vegas, Nevada

## Changing Hundreds of AutoCAD® Drawings in a Hurry

Speaker: Daniel Abbott, Southern Maine Community College

### CP34-1L

Last year, this workshop was the top-rated 90-minute lab at AU. This year you can learn to make changes to thousands of AutoCAD drawings at once by making an AutoCAD script, a Windows batch file, and an AutoLISP® program work together. Maybe you need to redefine a block definition in every drawing file (the company Title Block changed again!), reduce file sizes as much as possible in all archived drawings, create individual drawing files from each block definition in a group of drawings, or take any other action that can be completed with a LISP program. In this hands-on lab, you will get a chance to sit back and watch the computer do all the work.

#### Who Should Attend

Experienced AutoCAD users who are willing to do some typing

#### Topics Covered

- \* How to make the same change to multiple drawings automatically
- \* How to make virtually any change to all the drawings in a folder
- \* Write a program and run it yourself during this workshop
- \* Take the results home and adapt them to your own update problem
- \* See how combining LISP, scripts, and batch files can save you hundreds of hours

#### About the Speaker:

Dan has been involved with technical education for more than 20 years, the past 15 of which have been as a member of the Architectural and Engineering Design Department at Southern Maine Community College. Dan trains industry professionals in AutoCAD® use and management, and in geometric dimensioning and tolerancing. He works with Knowledge Development Solutions to help create technical exams for Autodesk. Dan earned M.S. and B.S. degrees in Industrial Technology from the University of Southern Maine, and a B.A. degree in Psychology from Swarthmore College.

Email: [dabbott@smccME.edu](mailto:dabbott@smccME.edu)

## Changing Hundreds of Drawings in a Hurry

### INTRODUCTION

A number of years ago I was asked to develop a process that would automate the task of changing a large number of drawings. This particular company had a need to open .dxf files in AutoCAD, explode all blocks and polylines, and save the drawings in .dwg format so they could then be opened in a different CAD program. They estimated that they had between 2000 and 3000 drawings, and they wanted to do this to all of their drawings. The process I developed allowed them to start a batch file and go home for the weekend. On Monday all the drawings were done.

I haven't encountered anyone else who needed to explode all the blocks and polylines in a drawing, but since then I have developed similar processes to do other things. One company wanted to reduce the size of their archived drawings as much as possible to preserve drive space. Another wanted to extract specific attribute data from all of their drawings and place it in a text file. Another wanted to update the title block in all of their drawings to reflect a company name change and change in ownership. Another wanted to create a separate drawing file from each block definition in all of their symbol-library drawings. I will be presenting three of these here.

There are several different paths you can take to accomplish these tasks. At the end we will briefly review the ScriptPro migration utility. The system you will use today requires the following three elements:

1. a script file
2. a batch file
3. a lisp program

Two cautions are in order here:

1. Try out any programming you do along these lines on sample drawings to make sure it works before using it on your real drawings.
2. When first trying this out, write your programs in such a way that NEW drawing files are created and don't delete the existing files until you know that the changes worked.

Since this workshop is a lab, I will walk you through the creation of the files needed to make it work using a step-by-step approach. We will be working with files located in "C:\datafiles\CP34-1L."

### BACKGROUND

#### Script File

Scripts are often overlooked as customizing tools. They are simply text files saved with an .scr extension that contain a list of instructions to AutoCAD that could otherwise be typed at the command line. What makes them essential for this process is that a script can be specified to run at startup by using the switch /b and specifying a script name. You can also specify a drawing and have that drawing open automatically when AutoCAD starts. By combining these two functions, you can have AutoCAD start with a specific drawing, and then run a script, which is the basis for this system.

The following line shows the syntax to use at a command prompt, in the target window of a desktop icon, or in the Run window on the Windows Start button.

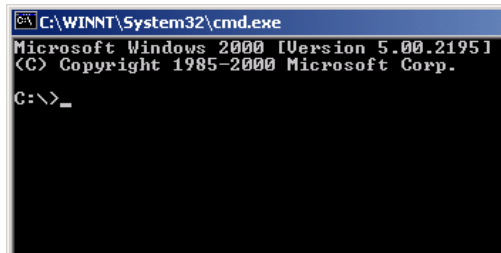
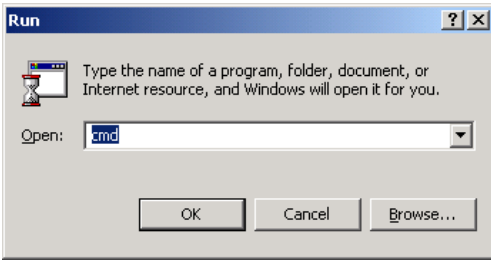
```
C:\Program Files\Autocad 2005\acad.exe c:\dwg\house.dwg /b c:\scripts\startup.scr
```

This line would start AutoCAD (**acad.exe**), open a specified drawing (**house.dwg**), and execute a named script (**c:\scripts\startup.scr.**) automatically.

#### Batch File

Like a script file, a batch file is a text file, but with a .bat extension. It contains operating system level commands. Those of you whose experience with personal computers goes back a few years will think of this as DOS. A command interpreter that allows batch files to run is still embedded in Windows 2000 and XP. You can get to it

directly by typing “cmd” in the Run Window from the Start menu. This opens an OS (operating system) window as shown. If you want, you can type DOS commands here. Or, you can place them in a batch file.



The reason I use a batch file to edit multiple drawings is because the batch file command “FOR” allows me to specify a directory and file names using wild cards, and have AutoCAD open each drawing file within that directory. It is even possible to open all drawing in the directory and in any subdirectories within a directory.

**Lisp File**

Like both script files and batch files, a lisp file is a text file, but with an .lsp extension. Lisp, however, is an actual programming language, and its tie to AutoCAD makes it possible to do nearly anything to a drawing. Every experienced AutoCAD user should at least be familiar with the programming language known as AutoLISP. In my system for updating drawings, it is Autolisp that generally does the work. If you don’t know it already, learning Autolisp is beyond the scope of this workshop, so I will provide you with the necessary lisp code for what we are doing. However, I do encourage you to explore this very useful tool, if you haven’t already.

Because Autolisp functions can be enclosed in parentheses and typed at the command line in AutoCAD, lisp code can actually be placed directly within a script file, or saved as its own file. Here I will have you save the Autolisp code in a separate file.

**PUTTING BATCH FILES, SCRIPTS, LISP CODE AND DRAWINGS ALL TOGETHER.**

*\*\*\*\*\*In each example, what you have to type is in text boxes\*\*\*\*\**

**Example one – reducing file sizes**

The first example we will use involves reducing the size of all drawing files within one or more directories to their minimum, using the WBLOCK command (you could, if you prefer, use the PURGE command). This would be useful for archived drawings, so you can eliminate unused layer names, block definitions, dimension styles, etc.

The following steps will allow you to use a script to automatically start AutoCAD, load the first drawing in a subdirectory, get the name of the drawing, use WBLOCK to save it with the same name in a different sub-directory, quit the drawing and go on to the next drawing in the directory to do the same thing.

**Step 1 – script file**

Use NOTEPAD (NOT a word processor) to create the following script file and save it as “wbout.scr” in a folder named “c:\datafiles\CP34-1L.” (I hope to have created this folder on each computer already. If not, create it, you must create it.) You should have five lines of text, with NO extra spaces or extra lines anywhere, especially at the ends of lines or the end of the file. You must, however, press ENTER at the end of the last line, but ONLY once.

```
(load "c:\\datafiles\\CP34-1L\\wbout.lsp")
ZOOM ALL
WBOUT
QUIT
Y
```

- Use two backslashes and include quotes.
- Leave a space between ZOOM and ALL.
- Don’t leave any spaces at the ends of lines.
- Don’t place any blank lines in file
- Press ENTER only once after typing “Y.”

## Changing Hundreds of Drawings in a Hurry

What does this all mean?

**(load "c:\datafiles\CP34-1L\wbout")** This uses the AutoLISP LOAD function to load the "wbout.lsp" file.

**ZOOM ALL** This changes something in the file to avoid having AutoCAD exit without saving during the use of the WBLOCK command in the Autolisp program.

**WBOUT** This issues the command WBOUT created by wbout.lsp.

**QUIT** This quits AutoCAD without saving the file again.

**Y** Response to the question "Quit without saving?".

### Step 2 -- batch file

Use Notepad to create the following batch file and save it as "wbout.bat" in the folder c:\datafiles\CP34-1L. Type each entry as a single line. You will have two lines of text when you are done.

```
md c:\datafiles\CP34-1L\dwg\wb
FOR %%f in (c:\datafiles\CP34-1L\dwg\*.dwg) do start /wait c:\program
files\Autocad 2005\acad.exe "%%f" /b c:\datafiles\CP34-1L\wbout.scr
```

Type the "FOR" statement on one line.

What does this all mean?

The first line creates a new folder (directory, sub-directory) name "wb" within an existing folder, so the new versions of the drawing files can be placed there until their integrity is confirmed.

The replaceable parameter %%f represents the name of each specified file within the folder named "c:\datafiles\CP34-1L\dwg." Each .dwg file in that location will be processed in alphabetical order as it is set to the replaceable parameter %%f. %%f will then be used in the second portion of the line as a variable, changing with each pass to the next drawing name. The **START /WAIT** option starts a windows application and returns control to the batch file when done. **Acad.exe** is the windows application that is being started.

AutoCAD will now start and open each drawing. Because of the /b switch, the script file named "wbout.scr" will run every time AutoCAD starts. When all files with a .dwg extension have been processed (made equal to variable %%f), the program will stop.

Note: the path given to locate the "acad.exe" file is not necessary unless there is more than one acad.exe file on the computer. This could happen if you have multiple releases of AutoCAD or if you have one or more vertical applications on your computer, as is the case on the computers in this lab. It is generally better to be precise in giving a path. The quotation marks within the batch file are required to compensate for spaces in the names of files or folders. Putting quotes around the second occurrence of %%f allows you to process drawing files that have spaces within their names as well. Place the quotations exactly as shown.

### Step 3 – Lisp Program

Use Notepad to create the following lisp file and save it as "wbout.lsp" in the folder c:\datafiles\CP34-1L. This file must be typed exactly as shown. It will get the current drawing name, issue the WBLOCK command, save the

existing drawing after purging everything possible (that's what the "\*" does), with the same name, but in a folder named "wb" within of the current specified directory. In this case, "c:\datafiles\CP34-1L\dwg."

```
(defun c:wbout(/ dn pa pawbdn)
  (setq dn (getvar "dwgname"))
  (setq pa (getvar "dwgprefix"))
  (setq pawbdn (strcat pa "wb\\" dn))
  (command "wblock" pawbdn "*")
)
```

Leave a space after the front-slash.

Type both quotes and both parentheses in lines two, three, and four.

Make sure that all the words are typed correctly.

What does this all mean?

<b>(defun c:wbout(/ dn pa pawbdn)</b>	defines the new command WBOUT
<b>(setq dn (getvar "dwgname"))</b>	get name of current drawing
<b>(setq pa (getvar "dwgprefix"))</b>	get path of the current drawing
<b>(setq pawbdn (strcat pa "wb\\" dn))</b>	creates new path by appending "wb"
<b>(command "wblock" pawbdn "*")</b>	saves entire drawing with same name in new folder
<b>)</b>	close first parenthesis

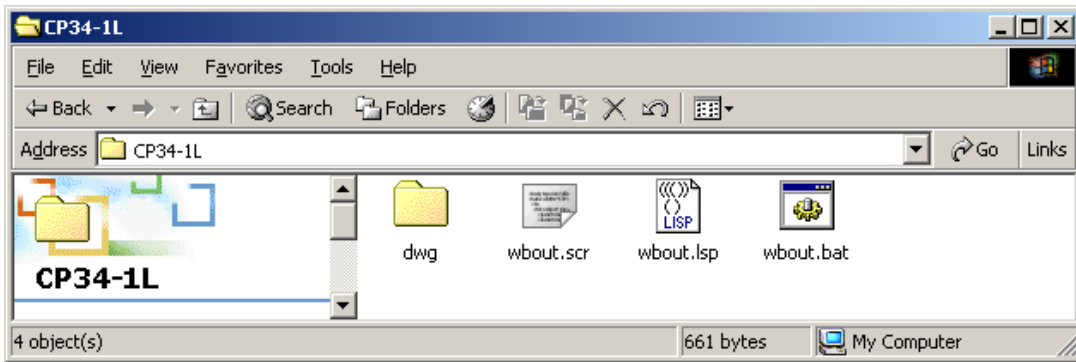
You might be wondering why I used a Lisp program here instead of a script to execute the WBLOCK command. Even though you can put lines of LISP code into a script, just like you can type such code at the command line, I prefer to write new lisp functions as separate files, using the VLIDE program. In this case, it is necessary to save the new drawing file created with the WBLOCK command to a specific location with a specific file name. By using a Lisp program I am able to get the current location and the current file name, modify them by appending a new folder name, and store them as variables, which can be combined into a single path and file name.

**Step 4 -- Run the program.**

There should be a group of drawings in the "c:\datafiles\CP34-1L\dwg" folder already. If not, just copy a few from the \Sample directory in AutoCAD. Once there are drawing files in that folder, you can start this process by running the batch file. The batch file is what gets this whole thing moving, because it selects each drawing file, and starts AutoCAD with the proper script.

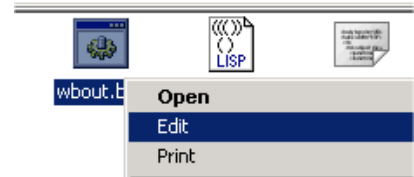
You can run this program from the Run window, a command prompt, from explorer, from My Computer, or from a desktop shortcut. For now, use the My Computer icon to find our folder on the C:\ drive, which will probably look like the following. The view of this dialog box can be changed, though, so it might look a little different.

## Changing Hundreds of Drawings in a Hurry



Double-click on the “wbout.bat” icon or select it and right-click. You should see AutoCAD opening and closing as it modifies each file.

NOTE. To run a batch file, you select OPEN from the right-click menu. To change the text in the file, you select EDIT. This is very different behavior than you might expect.



Using this collection of programs -- a batch file, a script, and a lisp routine -- can greatly reduce the space used on the hard drive for storage. However, as with any automated process, there are a number of things to consider -- namely, do you really want to eliminate all unused layers, blocks, styles, and dimstyles from the drawings? This kind of programming should be tested carefully on a limited number of expendable drawing files before using it widely. If you create a program like this for someone else, you must assure that they can use it safely. And it's not a bad idea to include a disclaimer at the beginning of the program, just in case something happens beyond your control.

Once you have run this program and are convinced that it did what you wanted, the old files can be deleted, and the new files placed where you want them. Or, you can alter the behavior of “wbout.lsp” so each drawing file is replaced automatically by the new smaller version. To do this, remove the reference to the \wb folder and set expert to 4 within your lisp program and reset it when the program is done, as follows:

<b>(defun c:wbout (/ dn pa padn f x)</b>	<b>;define a new command WBOUT</b>
<b>(setq x(getvar "expert"))</b>	<b>;get current expert value</b>
<b>(setq dn (getvar "dwgname"))</b>	<b>get name of current drawing</b>
<b>(setq pa (getvar "dwgprefix"))</b>	<b>get location of current drawing</b>
<b>(setq padn (strcat pa dn))</b>	<b>combine path and name of drawing</b>
<b>(setvar "expert" 4)</b>	<b>;set value of expert to 4 to suppress warnings</b>
<b>(command "wblock" padn "**")</b>	<b>replace existing drawing</b>
<b>(setvar "expert" x)</b>	<b>;reset value of expert</b>
<b>)</b>	<b>;closes the first parenthesis</b>

Note: It would be wise to add an error trapping function to this new program, to make sure that the value for EXPERT gets returned to its original setting, no matter what. See the end of this paper for an example.

To process drawings in sub-directories, you can add a switch to the FOR command in the batch file as follows. Note that the path now comes before the replaceable parameter, and the set (in parentheses) becomes just the wild card indicating the specific files. You can also use a text file to specify file names. See the help system for the FOR command for more details.

**FOR /R c:\datafiles\CP34-1L\dwg\ %%%f in (\*.dwg) do start /wait c:\program files\Autocad 2005\acad.exe "%%%f" /b c:\datafiles\CP34-1L\wbout.scr**

**Example two – redefining existing block definitions**

This second example would allow you to redefine any existing block definition within a group of drawings. The specific example will be of a title block that represents a company name change, or logo update, or other modification. Note that this is only feasible if every drawing contains a title block that has not been exploded, and if the replacement is fundamentally the same as the original.

**Step 1 – script file**

Open Notepad and type in the following script. Save it as “C:\datafiles\CP34-1L\border\_update.scr.”

```
INSERT border=c:\datafiles\CP34-1L\dwg\border
0,0 1 1 0
ERASE L

(load "c:\\datafiles\\CP34-1L\\border_update.lsp")
border_update
QUIT
Y
```

There are no spaces in the first line  
 There are three spaces in the second line  
 There is a space between ERASE and L  
 This is a single blank line  
 Make sure you have both quote marks.  
 This command name uses an underscore  
 Press ENTER only once after typing “Y.”

```
INSERT border=c:\datafiles\CP34-1L\dwg\border
0,0 1 1 0
ERASE L

(load "c:\\au03Abbott\\border_update.lsp")
border_update
QUIT
Y
```

this will redefine existing block “border” using a drawing insertion point, X,Y scales of 1 and rotation angle of 0  
 erases the block just inserted after the redefinition  
 A blank line = ENTER – stops selecting objects to erase  
 loads an Autolisp file to create a new command  
 issues the new command created by the LISP file  
 ends the drawing session  
 answers “yes” to question “quit without saving?”

**Step 2 – batch file.**

Open Notepad and type the following two lines. Save the file as “C:\datafiles\CP34-1L\border\_update.bat.”

```
md c:\datafiles\CP34-1L\dwg\NewBorder
for %%%f in (c:\datafiles\CP34-1L\dwg\D5*.dwg) do START /WAIT c:\program files\Autocad 2005\acad.exe "%%%f" /b c:\datafiles\CP34-1L\border_update.scr
```

The first line makes a new directory for the new drawings being created.

The second line must be typed as a single line, and is similar to the batch file in example one. This one is more selective about the files being selected. Only drawings beginning with the characters “D5” will be opened.

## Changing Hundreds of Drawings in a Hurry

### Step 3 – lisp program

Use Notepad to type the following lines and save the file as "C:\datafiles\CP34-1L\border\_update.lsp."

```
(defun c:border_update(/ dn pa panbdn)
  (setq dn (getvar "dwgname"))
  (setq pa (getvar "dwgprefix"))
  (setq panbdn (strcat pa "NewBorder\\" dn))
  (command "SAVE" panbdn )
)
```

Space after the frontslash

Two closing parentheses and two quotation marks in each of these three lines.

Don't forget this last parenthesis

What does this all mean?

```
(defun c:border_update(/ dn pa pawb md panbdn) ;defines the new command BORDER_UPDATE
  (setq dn (getvar "dwgname")) ;get name of current drawing file
  (setq pa (getvar "dwgprefix")) ;get path of the current drawing file
  (setq panbdn (strcat pa "NewBorder\\" dn)) ;create new path by appending "NewBorder"
  (command "SAVE" panbdn) ;save with same name in new folder
) ;closes first parenthesis
```

### Step 4 – run the program

Go to the folder "C:\datafiles\CP34-1L\" and open the .bat file named "border\_update.bat." Sit back and watch it work.

## Example three – creating drawings from block definitions

The purpose of this next example is to automatically open a group of drawings that contain symbols, and create a new drawing file from each of the block definitions in each drawing. It uses the same process as examples one and two, but with a different lisp program.

### Step 1 – script file

Open Notepad and type the following script. Save the file as "C:\datafiles\CP34-1L\blockout.scr."

```
(load "c:\\datafiles\\CP34-1L\\blockout.lsp")
blockout
QUIT
Y
```

No blank lines anywhere in file.

Press ENTER once after typing "Y."

### Step 2 – batch file

Open Notepad and type the following script. Save the file as "C:\datafiles\CP34-1L\blockout.bat."



```
md c:\datafiles\CP34-1L\dwg\NewBlocks
for %%f in (c:\datafiles\CP34-1L\dwg\sym*.dwg) do START /WAIT c:\program
files\Autocad 2005\acad.exe "%f" /b c:\datafiles\CP34-1L\blockout.scr
```

**Step 3 – lisp file**

Open Notepad and type the following script. Save the file as "C:\datafiles\CP34-1L\blockout.lsp."

```
(defun c:blockout (/ dn pa s1 blkdata blname fullname)
  (setq dn (getvar "dwgname"))
  (setq pa (getvar "dwgprefix"))
  (setq s1 (strcat pa "NewBlocks\\" dn))
  (setq blkdata (tblnext "BLOCK" T))
  (while blkdata
    (setq blname (cdr (assoc 2 BLKDATA)))
    (setq fullname (strcat s1 blname))
    (command "WBLOCK" fullname blname)
    (setq blkdata (tblnext "BLOCK")))
  )
)
```

Space after the front slash.

No spaces within quotes

Don't forget the T  
No closing parentheses YET

Three closing parentheses

What does all this mean?

```
(defun c:blockout (/ dn pa s1 blkdata blname fullname)
  (setq dn (getvar "dwgname"))
  (setq pa (getvar "dwgprefix"))
  (setq s1 (strcat pa "NewBlocks\\" dn))
  (setq BLKDATA (tblnext "BLOCK" t))
  (while BLKDATA
    (setq BLNAME (cdr (assoc 2 BLKDATA)))
    (setq fullname (strcat s1 blname))
    (command "wblock" fullname blname)
    (setq BLKDATA (tblnext "BLOCK")))
  )
)
```

```
;defines new command with 6 variables
;get name of current drawing and save as "dn"
;get path of current drawing and save as "pa"
;add new folder to path and save as s1
;set BLKDATA = to name of first block in list
;loop through steps if BLKDATA has a value
;set BLNAME equal to the name of next block
;create a variable with path and block name
use WBLOCK to create drawing file from block
;set BLKDATA = to name of next block in list
;end the WHILE function
;end the program
```

## Changing Hundreds of Drawings in a Hurry

### ScriptPRO

ScriptPro is a migration tool that provides a graphical user interface to apply some scripts to multiple drawings. You can download it for free. The file name is "CCTSETUP.EXE." Here is the site as of 10/04. If this link doesn't work, type "Autodesk Customization Conversion Tools" in the search window at [www.autodesk.com](http://www.autodesk.com).

<http://usa.autodesk.com/adsk/servlet/item?siteID=123112&id=4091678&linkID=2475161>

I don't use ScriptPro, because I have been unable to get it to run scripts that include AutoLISP code. If you can accomplish what you want using ONLY command line input without using AutoLISP, it may be useful to you. It does have some built-in functions for converting drawings to .dxf or earlier releases, and some additional script functions. You can get file names and paths from within a script by using the following ScriptPro functions: <acet:cFolderName> specifies the drawing file folder name (directory name); <acet:cBaseName> specifies the base file name without a directory or extension; <acet:cExtension> specifies the extension for the drawing file (.dwg, .dwt, or .dxf); <acet:cFileName> specifies the base name with the extension (DWGNAME system variable); <acet:cFullName> specifies the full file name with path and extension.

There are also some useful sample scripts supplied with ScriptPro for saving AutoCAD files in different formats. However, R14 is not one of the drawing formats supported. To convert 2004 drawings to Release 14 format, you will need the batch drawing converter. Download it as "BDCSETUP\_ENU.EXE" from the same link given above.

### Adding Error-Trapping to AutoLISP programs.

To reduce the typing needed for this workshop, I simplified the AutoLISP programs we used as much as possible. However, if you use an AutoLISP program in which variables are changed, it is a good idea to include error-trapping within the program. The following program is an example of how error-trapping could be added. For more information, I recommend the discussion at <http://www.afraisp.com/lispa/lisp6.htm>, an excellent site.

```
;;; Error Trapping function
(defun da_error (msg)
  (command)
  (setvar "aperture" ap) (setvar "osmode" os)
  (setq *error* old_error)
  (alert "Returning drawing to original status.")
)
;;; Program places a point midway between any two parallel lines -- helpful for dimensioning to wall centers.
(defun C:MID (/ p1 p2 old_error)
  (setq old_error *error*)
  (setq *error* da_error)
  (setq ap (getvar "aperture"))
  (setq os (getvar "osmode"))
  (setvar "aperture" 3)
  (setvar "osmode" 512)
  (setq p1 (getpoint "\nFirst point: ")
        p2 (getpoint "\nSecond point: " p1)
  )
  (setq m (polar p1 (angle p1 p2) (/ (distance p1 p2) 2.0)))
  (command "point" "non" m)
  (setvar "aperture" ap) (setvar "osmode" os)
  (setq *error* old_error)
  (princ)
)
```