AUTODESK UNIVERSITY 2015

GEN10753

Ins and Outs of Attributes

John G Jordan / Automation & Control Concepts

Learning Objectives

- Learn how to create an attributed block
- Learn how to modify provided lisp routine to work with your block
- Format your data for use with the block and lisp
- Learn how to use existing data and the lisp routine to populate the drawing
- Learn how to extract this information for sharing in the design process

Description

Automating Attributes is a great way to speed up the drawing process. During this class you will learn how to import existing design data into your drawings through an automated process. By using existing data, you will decrease user input and errors, thus increasing accuracy. We will use Notepad, Microsoft Excel, and AutoCAD software to accomplish these goals.

Your AU Experts

John is an Electrical Designer and Cad Manager in St. Louis, Mo with over 30 years of experience using and customizing AutoCAD. Serving as Cad Manager John has created CAD Standards and incorporated those standards into the drawing process through customization using tool palettes and AutoLISP. Developed AutoLISP routines to automate small and large parts of the drawing process and for populating data into drawings. John has Associate degrees in Engineering Sciences and Electrical Engineering Technology. John has served as a lab assistant at Autodesk University since 2004.

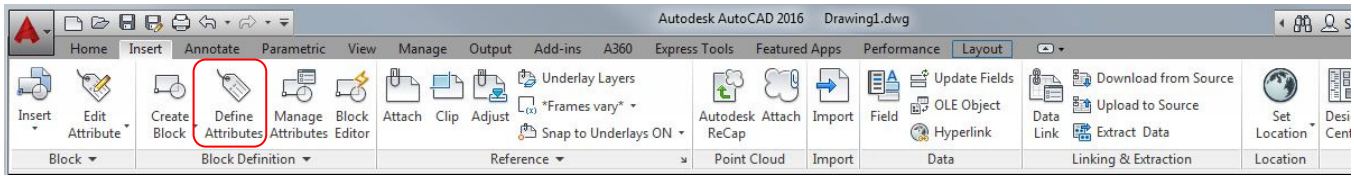
Philosophy of the Class

The goal of any production staff personnel is to create things faster, better and with less expense. Every click or keystroke you don't have to execute is a savings of time and dollars. So if you can use existing data from other parts of the design process to populate items in your drawing, think of all the typing that you could save. The goal of this class is to show you how with a little manipulation,to use that data in your drawing.

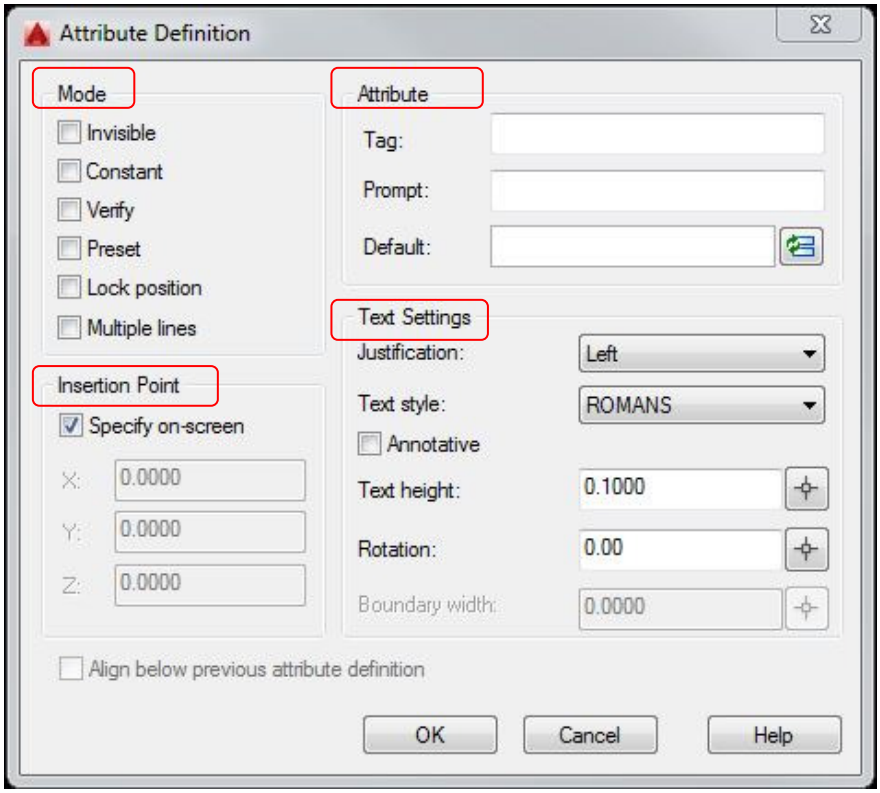
Part 1 - Creating an Attributed Block

Plan your block, define the attributes and add graphics if desired.

We want to create an attribute by using the define attribute command, found in the ribbon on the insert palette, or key in "ATTDEF"



Once you pick this command you will get a dialog box like below.



This box is made up of four sections, Mode, Insert Point, Attribute and Text Settings. We will go through these quickly so that we can create the block and move on to the lisp for inserting the information. For a detailed listing of each item in the box, search AutoCAD help with "Attribute Definition Dialog Box".

The **Mode** section has a variety of options that allow you to do several creative things most are obvious; however today we are going to use the defaults. Some of these can help you to quickly add information to the drawing and to load the block with a wealth of information.

The **Insertion Poi**nt is obvious, pick a point or specify coordinates.

Next we will move to the **Text Settings**, pick the justification that you want, the text style will only give you what you have defined in the drawing. Height and rotation are standard. I have never used annotative; however you would probably want to consider if the block is actual (drawing to real world dimensions) or symbolic (scaled to the drawing scale).

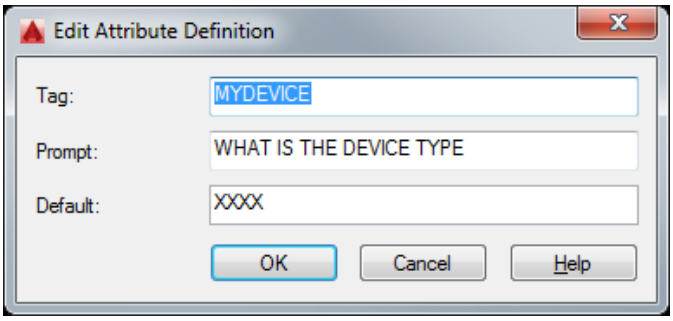
Now we come to the **Attribute Data**; this is what makes the whole thing work - the important part.

The **Tag** is the identifying variable or index. This should be unique to the attribute so that you can differentiate between the different data entities, examples would be the item number, type of device, sequence or location.

Next is the **Prompt**, this is the line that you will see to guide you when manually inputting data into the attribute; example "Enter the Equipment Item Number"

Last is the **Default Value**, which is the value that you are prompted for on insertion and if you enter a return without entering data you will get the default value for data.

Once we have created one attribute, we can repeat to create all the other attributes that we need, or since I typically want the all of them to look alike I just copy the first and edit the tag/prompt/data fields and I have a new attribute.

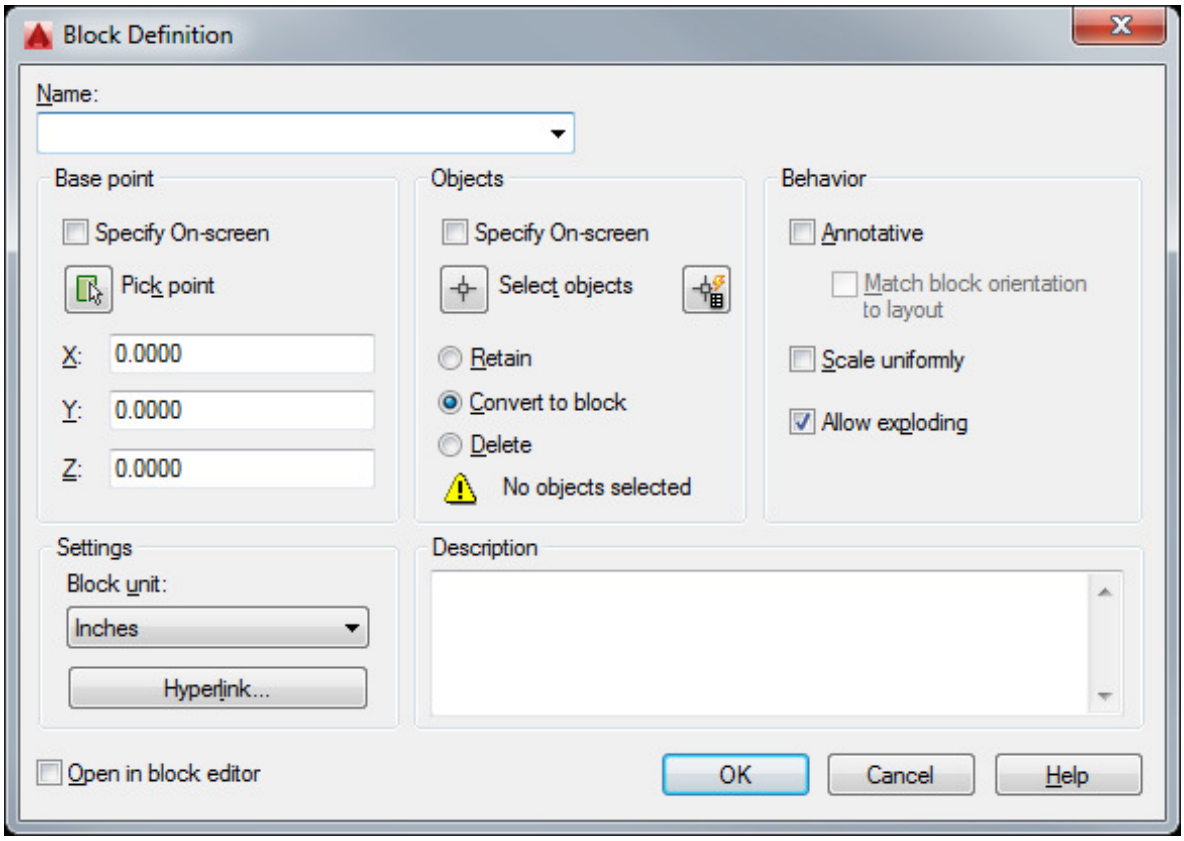


Until the attribute is part of a block definition you will not get the Enhanced Attribute Editor dialog box.

Once you create the attributes and place them where you want them arranged, you can add graphics for the block if needed.

Create the block using the block command. The graphics can be picked in any order. **The key point to remember here is that the attributes will appear in the order that they are selected when creating the block.** Therefore you want them to appear in the order that makes sense for your data.

Block dialog box



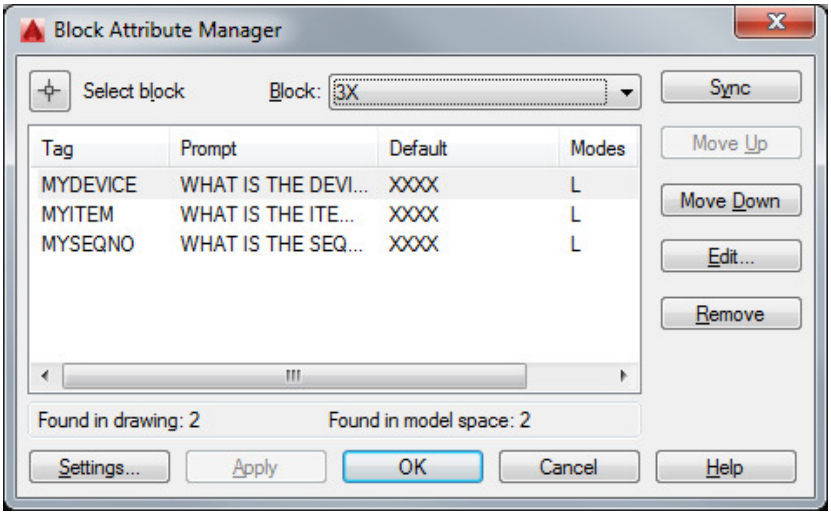
Start with the **Name**, give your block a meaningful name.

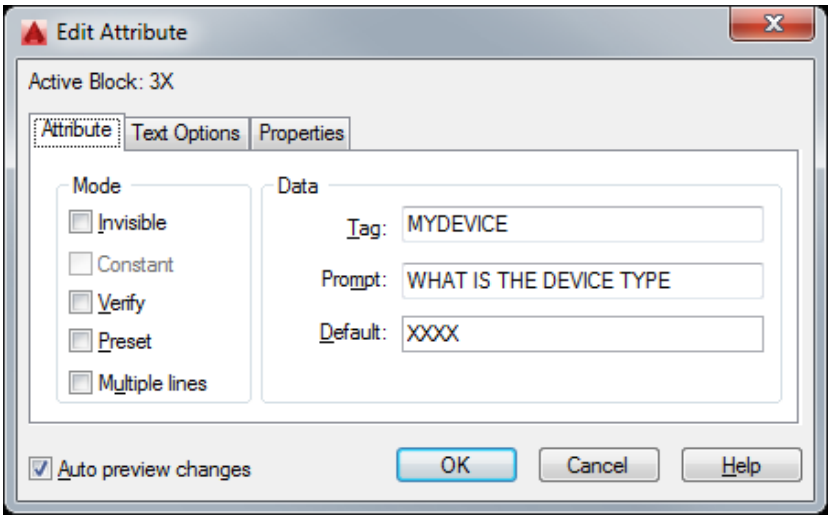
Specify the **Base point**, generally it is easiest to pick the insert on screen relevant to the graphics.

Select the **Objects** - this is an important part of the process. The attributes need to be picked in the order that you want them to be displayed in the dialog box. Graphics can be randomly picked.

Once the block is created we want to insert it to check the order of the attributes as the block is inserted because it is important that it match the order of the data we want to import to the attributes.

If you get them in the wrong order, you can use the command **BATTMAN** which will allow you to reorder the attributes. By double clicking on the attribute you can open the dialog box Edit Attribute which is different than the Edit Attribute Definition dialog box and will allow you to change the tag, prompt, and default as well as the text options and properties.





So now the block is created, we have the attributes in the order that we want and have added any graphics to make it pictorially representative of something we are trying to communicate. We are ready to move on to the Lisp portion of the process.

You can use the block editor if needed to modify the graphics to the block.

Part 2 - Modifying the AutoLisp File to Work with Your Block

AutoLisp files are saved in ab ASCII format, you need to use an editor that savves in that format, I tend to use Notepad for simplicity.

Minimum change is block name and more or less attributes.

Maximum is open to the imagination.

Taking the provided lisp routine, with minimal programming knowledge (enough to hack) and adding additional comments. We get the following.

Basic Lisp Lesson - 100.5 (not even good enough for 101)
Anything following a ; "semicolon" is a comment and is ignored.
Every ("open parenthesis" must have a "closing parenthesis", not necessarily in the same line.
Every quotation mark " must have a closing quotation mark ", items inside quotation marks are taken as literal. Numeric values work best with leading and following zeros.

```
;~*****************************************************************************
;~* FILE NAME:DATA-IN.lsp
;~* FUNCTION: INSERT BLOCKS WITH ATTRIBUTES INTO DRAWINGS AND FILL ATTRIBUTES
;~* WRITTEN BY: JOHN G. JORDAN @ Automation & Control Concepts
;~* REVISIONS:  Version 1:
;~*****************************************************************************
```

Above is all comments that are the header to the program so that we know what it is and does.

```
(defun C:DATAIN ()
```

Defun C: defines it as a program, that can be called from the command line, the name of the command is DATAIN, the () is required and is used to store variable names for the program if done properly.

```
(setvar "cmdecho" 0)              turns of the command line echo
;#1a----------     layer & text settings  ------------
          ;create and set layer to blocks - create romans text style
```

Creates and sets the layer and the text style

```
(command "layer" "m" "BLOCKS" "c" "GREEN" "" "l" "continuous" "" "")
(command "style" "romans" "romans" "0.125" "1" "" "" "" "")
```

Should be self explanatory

```
;#1b----------     DEFINING VARIABLE SETTINGS
          ; capture user settings
(setqsmd (getvar "osmode"))     ;osnap setting
(setqarq (getvar "attreq"))     ;attribute request
(setqatd (getvar "attdia"))     ;attribute dialog box
          ; change to what settings need to be
```

```
(setvar "osmode" 0)                ;osnap to none
(setvar "attreq" 1)        ;ask for attribute values on insert
(setvar "attdia" 0)        ;issues command prompts for data
```

Creates the values used to create the data input points

```
          ;set starting points for inserts and not nil
(setqxval4.0)                      ;initial x
(setqyval 29.0)            ;initial y
(setqeqdata "notnil")              ;dummy variable

(command "zoom" "all")

;prompt user for the location of the data file
(princ "\n Please Enter Name of Data File - Include the path and extension")
(princ "\n")
(setqeqdataf (strcase (getstring "\n Data file = ")))
```

Do not modify from #3 to #3a

```
;#3---------- *** while motor data ***
(setqeqdatal (open eqdataf "r")); open data file

(while (/= eqdata nil)             ; while data continue
  (setqeqdata (read-line eqdatal))         ; read new line of data
  (setqeqdatac (strcaseeqdata))  ; change to upper case

  (setqeqdat (sparse eqdatac ","))
;#3a---------- do not change anything between 3 - 3a
```

The names of the variables (names following a setq) in caps can be changed, if so you need to change them six lines down in the insert command

```
  (setq EQPID (nth 0 eqdat)                ;EQUIPMENT NO
          EQDES (nth 1 eqdat)              ;DESCRIPTION
          NEWIO (nth 2 eqdat)              ;SEQUENCE

   )      ; end setq

;insert tag & text
          (setqblkinpnt (list xvalyval))       ; create xy insert point for tag

          (command "insert" "3X" blkinpnt "1" "" "" EQPID EQDES NEWIO)

; set new point
```

```
            (setqyval (- yval 1.0))      ; y point
```

Evaluates the position of the last insert and adjust for the drawing space.

```
            (if (<yval 2.5)
            (PROGN
            (setqyval 29.0)
            (setqxval (+ xval 3.0))
            )  ;END PROGN
            (SETQ SIXPACK "EMPTY")
            )  ;END IF

)  ;end while
```

Resets the users variables

```
(setvar "osmode" smd)
(setvar "attreq" arq)
(setvar "attdia" atd)

(close "eqdataf")            ;CLOSES THE OPEN DATA FILE

(setvar "cmdecho" 1)
) ; end defun
```

Subroutine that will change the order of the variables when read in.

```
;------------------
            ;SPARSE (from bryan)

(defun sparse (s asep / slenll temp cnt)
            (setq temp "" slen (strlen s) cnt 0)
(while
            (<cntslen)
            (setqcnt (1+ cnt))
(if (= (substr s cnt 1) asep)          ; test if character is a seperator
            (setqll (cons temp ll)      ; then adds var. to list
            temp "")
            (setq temp (strcat temp (substr s cnt 1)))            ; else
            )           ; end if
            )           ; end while
(reverse (cons temp ll))
            )           ; return list ll in order data was read
;----------------------
```

If you open in VlispEditor inside of AutoCAD it would look like the two pictures following:

Visual LISP Console

```
$ ;~*************************************************************************************
;~* FILE NAME:DAT-IN.lsp
;~* FUNCTION:
;~* WRITTEN BY: JOHN G. JORDAN @ Automation & Control Concepts
;~* REVISIONS:  Version 1: INSERT BLOCKS WITH ATTRIBUTES INTO DRAWINGS
;~*************************************************************************************
****

(defun C:DATAIN ()

(setvar "cmdecho" 0)
;#1a----------- layer & text settings  ------------
        ;create and set layer to blocks - create romans text style
(command "layer" "m" "BLOCKS" "c" "GREEN" "" "l" "continuous" "" "")
(command "style" "romans" "romans" "0.125" "1" "" "" "" "")

;#1b----------- DEFINING VARIABLE SETTINGS
        ; capture user settings
(setq smd (getvar "osmode"))     ;osnap setting
(setq arq (getvar "attreq"))     ;attribute request
(setq atd (getvar "attdia"))     ;attribute dialog box
        ; change to what settings need to be
(setvar "osmode" 0)
(setvar "attreq" 1)
(setvar "attdia" 0)

        ;set starting points for inserts and not nil
(setq xval 1.0)                  ;initial x
(setq yval 29.0)                 ;initial y
(setq eqdata "notnil")           ;dummy variable

(command "zoom" "all")

;prompt user for the location of the data file
(princ "\n Please Enter Name of Data File - Include the path and extension")
(princ "\n")
(setq eqdataf (strcase (getstring "\n Data file = ")))

;#3----------- *** while motor data ***
(setq eqdatal (open eqdataf "r"))        ; open data file

(while (/= eqdata nil)                   ; while data continue
  (setq eqdata (read-line eqdatal))      ; read new line of data
  (setq eqdatac (strcase eqdata))        ; change to upper case

  (setq eqdat (sparse eqdatac ","))
;#3a---------- do not change anything between 3 - 3a

  (setq EQPID (nth 0 eqdat)              ;EQUIPMENT NO
        EQDES (nth 1 eqdat)              ;DESCRIPTION
        NEWIO (nth 2 eqdat)              ;SEQUENCE

   )     ; end setq

;insert tag & text
```

Visual LISP Console

```
   )     ; end setq

;insert tag & text
        (setq blkinpnt (list xval yval))        ; create xy insert point for
tag

        (command "insert" "3X" blkinpnt "1" "" "" EQPID EQDES NEWIO)

; set new point
        (setq yval (- yval 1.0))        ; y point

        (if (< yval 2.5)
        (PROGN
        (setq yval 29.0)
        (setq xval (+ xval 3.0))
        )  ;END PROGN
        (SETQ SIXPACK "EMPTY")
        )  ;END IF

)  ;end while

(setvar "osmode" smd)
(setvar "attreq" arq)
(setvar "attdia" atd)

(close "eqdataf")       ;CLOSES THE OPEN DATA FILE


(setvar "cmdecho" 1)
) ; end defun

;------------------
        ;SPARSE (from bryan)

(defun sparse (s asep / slen ll temp cnt)
        (setq temp "" slen (strlen s) cnt 0)
(while
        (< cnt slen)
        (setq cnt (1+ cnt))
(if (= (substr s cnt 1) asep)   ; test if character is a seperator
        (setq ll (cons temp ll) ; then adds var. to list
        temp "")
        (setq temp (strcat temp (substr s cnt 1)))      ; else
        )       ; end if
        )       ; end while
(reverse (cons temp ll))
        )       ; return list ll in order data was read
;-----------------------
```

I will not be using the Vlisp Editor in the demonstration for this class.

Part 3 - Formatting Your Data

The data needs to be in an ASCII file format, with a comma delimiter.

The data can come from any source such as a database or spreadsheet, I prefer someone elses data. However if i have to create it, I want a way to share it with others. So I create a document to share or extract the data from the drawing if it has to be done the old fashioned way (keystroked).

The most compatible tool that is a commonly used software by most people is Excel. It is fairly easy to set up the three or so columns from existing data, or create your own to match the attribute data.

If you look at the file DATA.xls you will see that there are three columns to match the number of attributes in our block.

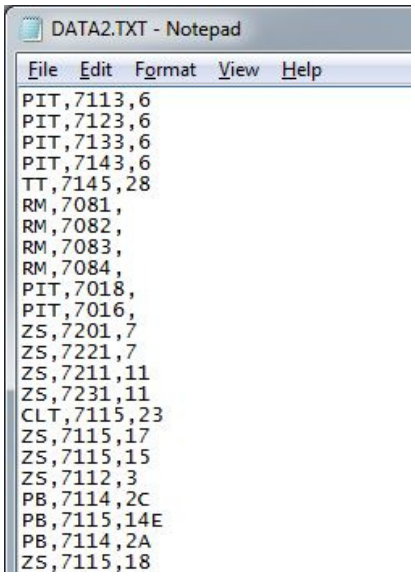
You will notice where the highlighted cell is there is no data. It is important to realize that there must be a blank or "space" entered here or the default value of XXXX will appear in the drawing for that attributes data.

	A	B	C	D
1	PIT	7113	6	
2	PIT	7123	6	
3	PIT	7133	6	
4	PIT	7143	6	
5	TT	7145	28	
6	RM	7081		
7	RM	7082		
8	RM	7083		
9	RM	7084		
10	PIT	7018		
11	PIT	7016		
12	ZS	7201	7	
13	ZS	7221	7	
14	ZS	7211	11	

After you have completed your data, save it in XLS format, but also save it in a CSV format. Note that a CSV format is an ASCII format and can only support one worksheet per file.

Once the file is saved as a CSV you can copy it or rename it to a TXT file, at this point it is a good idea to open it in Notepad to see what the format looks like. It should look something like the file below depending on your data.

Note that there are no headers, and don't forget to check for those spaces. If your file looks like the data is all there and in the correct order then proceed to the next step. Otherwise go back and fix it so that you get the right information into the drawing in the correct order.



Part 4 - Making It All Work Together

Once the block is created, the data is in the format that you want it, and the lisp file is edited to work with your block, it is time to put it all together.

Open a drawing, and insert the block created for the data. Place the block somewhere on the side so that it is not in the way when you run the program. The block has to be in the drawing for the lisp file to find it to insert it.

The lisp file needs to be loaded to the open file so the command is available. The easiest way to load the file is to drag and drop it into the open drawing file from the Windows Explorer window. The command line will show that the lisp loaded properly.

At this point we can type the command name DATAIN at the command prompt. Follow the prompt " Please Enter Name of Data File - Include the path and extension". I like to copy the path from the Windows Explorer window, then past it onto the command line and add the name and extension of the file. Then press enter and watch the data appear.

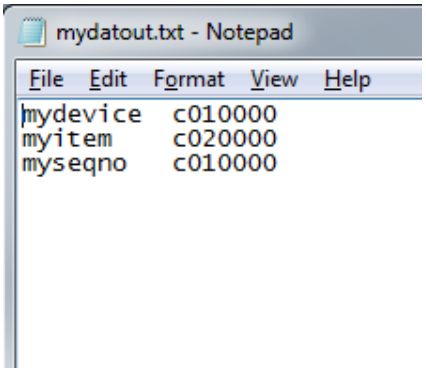
Each set of data should have been inserted into the block and arrayed in the drawing. From where you can now move them to the desired location. All the data is in the drawing and it did not have to type or mistype it.

Part 5 - Getting the Data Out

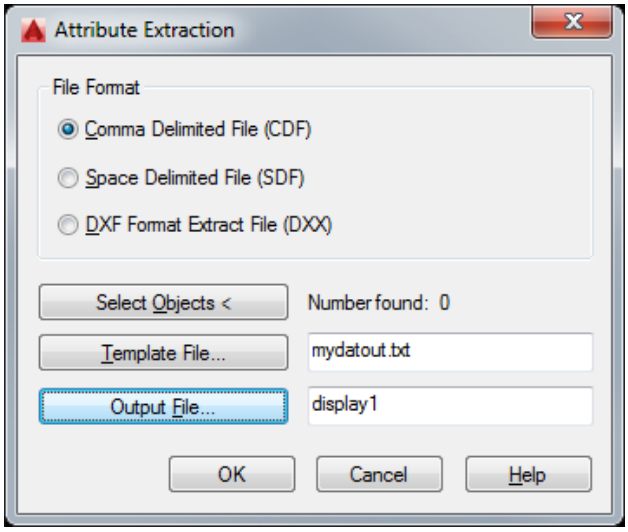
Once you have your data in the drawing, and things change as the project progresses. Well you have been keeping up with the changes in your drawing so the data does not look like it did originally. Well you can get it out of the drawing to share with others that might need it.

Old School -

Start by creating a data extraction template file. This is a simple ASCII text file. The minimal file would have the attribute tag and field definition. The one for the block in this demonstration would look like this.



Each line in the file is to extract one attribute, the first field "mydevice" is the attribute tag. The next sequence of seven characters is made up of three segments. First is the letter "c" which stands for character, the other option is "n" for numeric value. The first three of the numbers "010" defines the length of the character string, or how long the text will be. The second set of numbers are for the number of decimal places if the first character is a "n". If the first character is a "c" then the last three will always be "000".



Using the ATTEXT command you can extract all the data in the drawing out to a txt file for uploading to Excel.

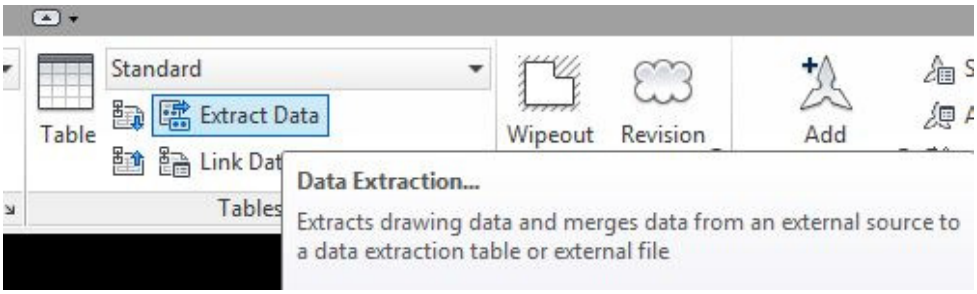
When using the ATTEXT command you will have to direct it to the template file, and look at where the output file is being saved and change it to where you want. After you do it once it should stay set that way. Just remember that you need to change those items for each project.

New School -

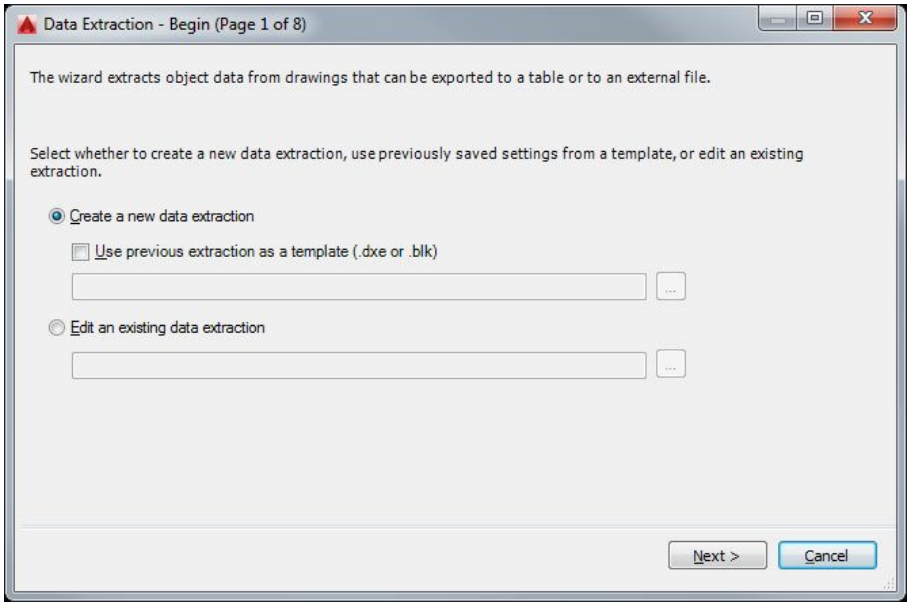
There is also the DATAEXTRACTION command that will walk you through an eight page dialog box to put the data in an external file or table in the drawing. This command can be accessed through the menu.

Ribbon - Annotate > Table > Extract Data

Or from the command line with the command DATAEXTRACTION or EATTEXT.

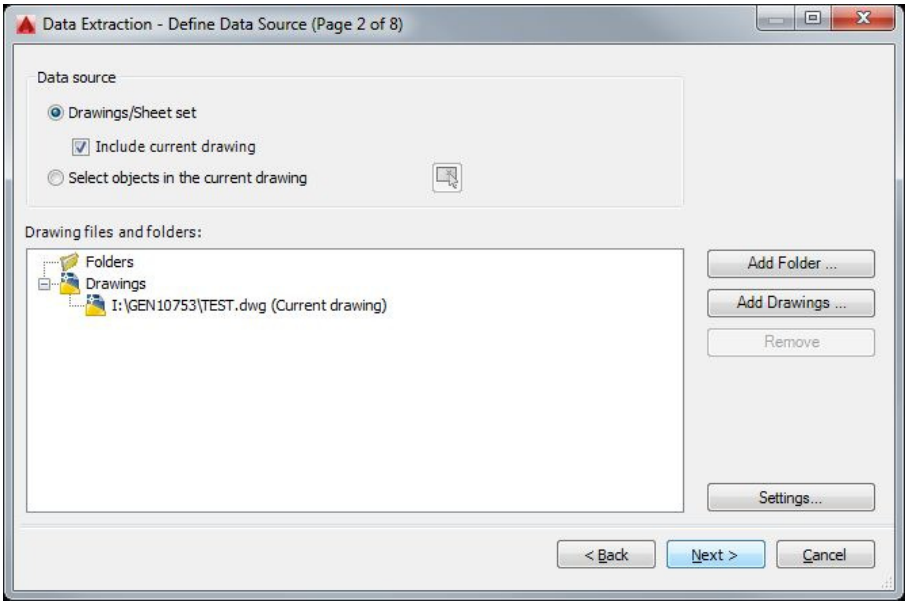


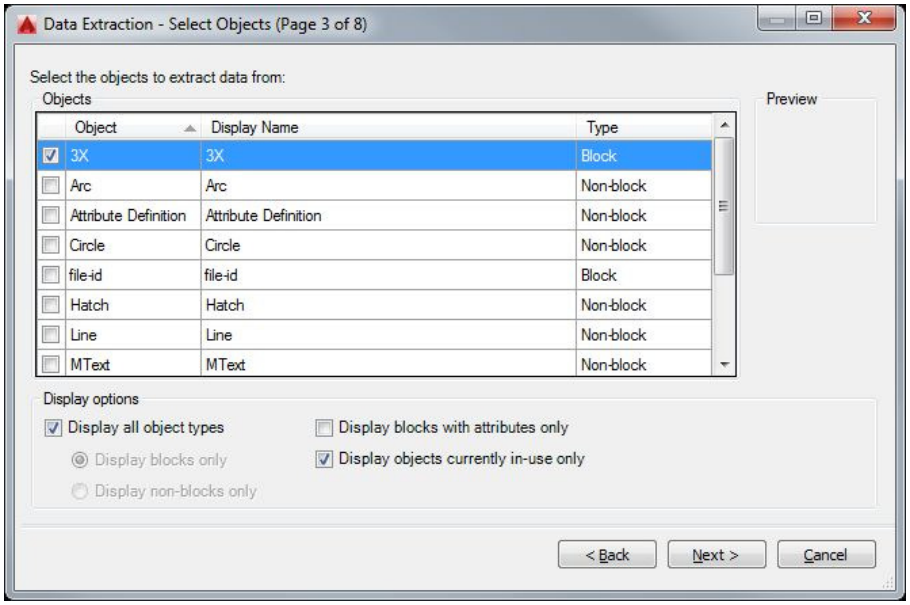
This will open a dialog box and prompt you for selection of all the data that you could possibly dream of out of the drawing.



The first box allows you to create a new template or use and modify an old one.

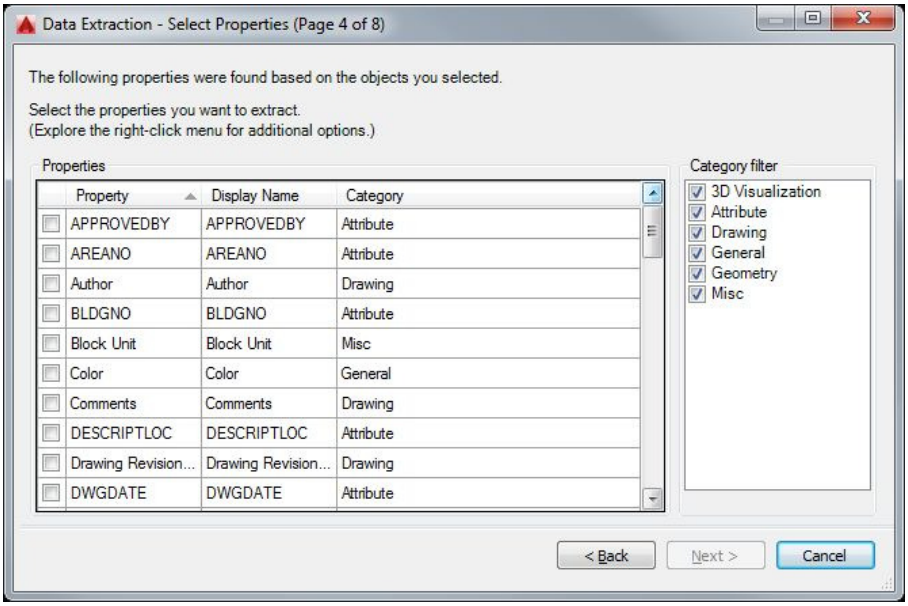
The second box will allow you to pick the current drawing or multiple an whole folders which is nice.

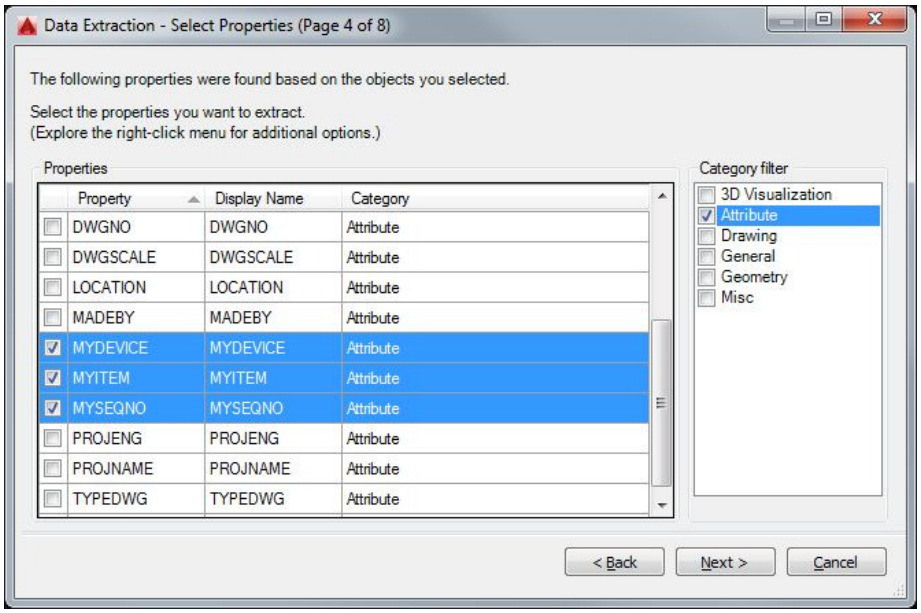




The third box will allow you to pick the items in general that you want to collect data from.

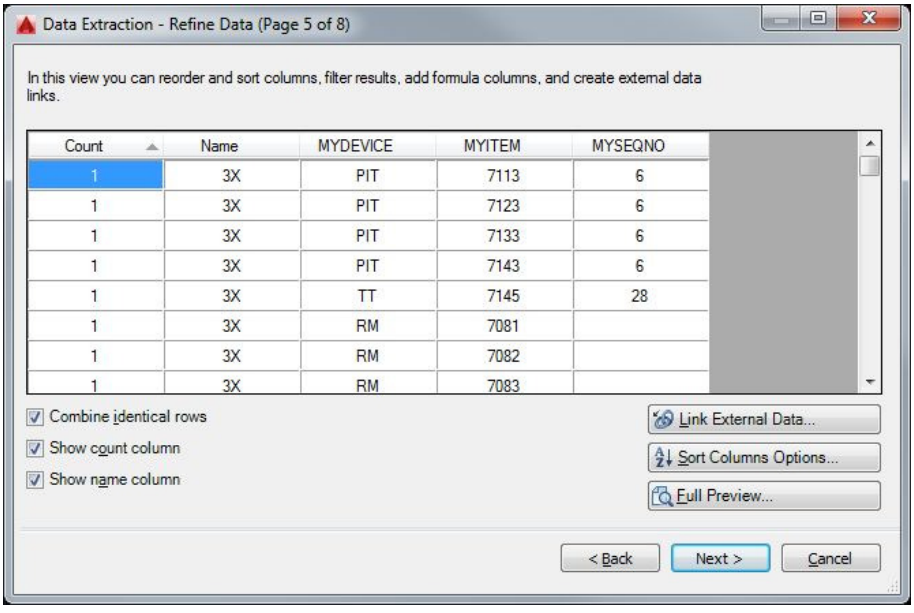
Box number four you get to refine your selection and the properties that you want.

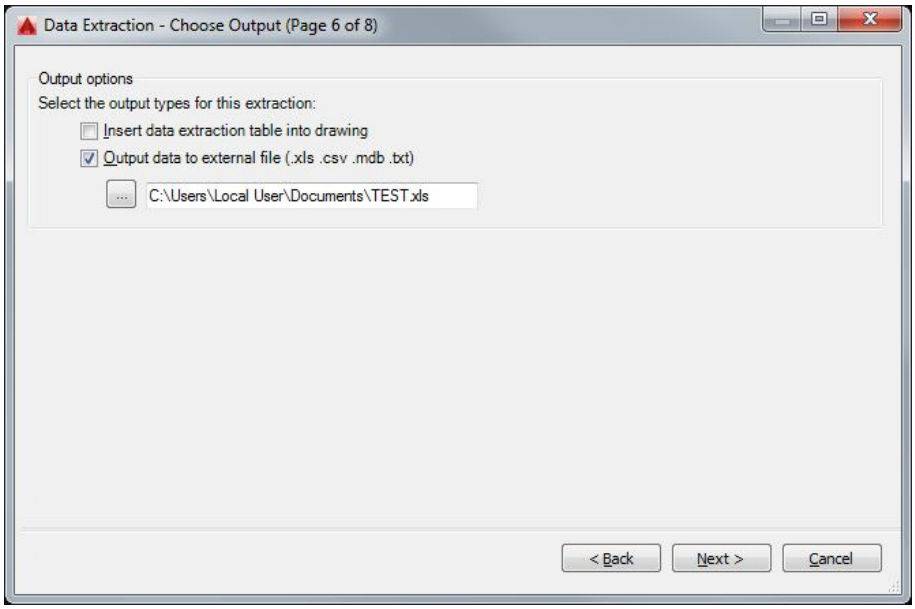




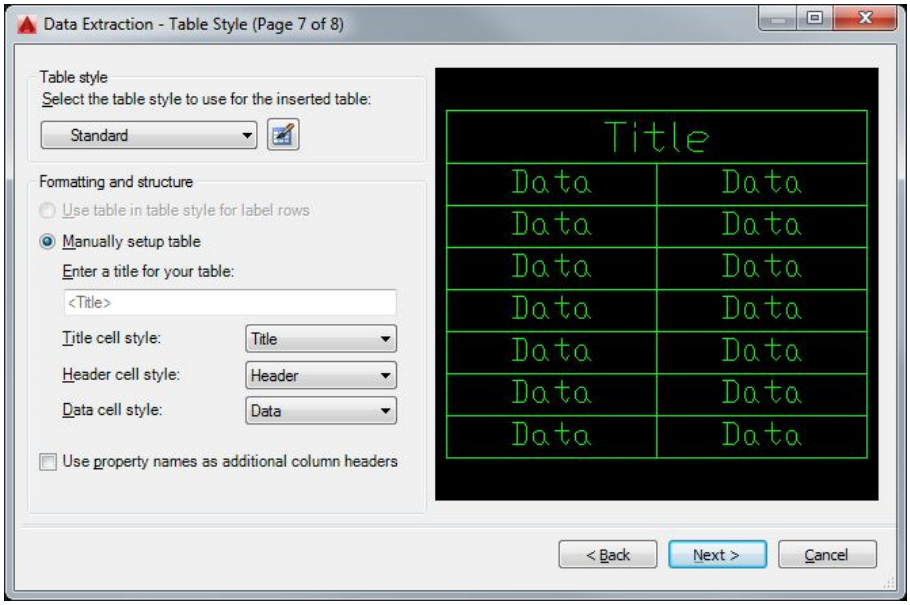
As you see there are more attributes than we are looking for in this drawing, the extraction takes all the entities in the drawing unless specified. So anything that is Layout as well as Model will show up in the list. Notice that by checking on and off different boxes you can select exactly what you need.

Box five gives you a preview of the resulting data search so that you can see if you got what you really thought you were getting.

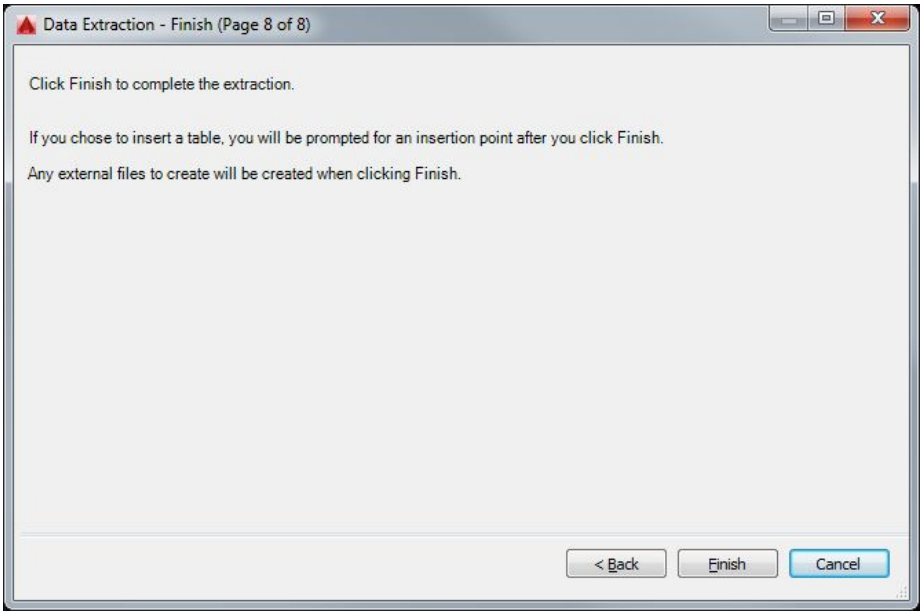




Six allows you to save the data in the drawing as a table (which will get you to box seven) or out of the drawing in one of the following formats xls, csv, mdb, or txt.



Box seven will only appear if you are inserting a table into the drawing.



And finally page eight that only confirms that you complete the process by clicking on the finish button. At which point the dialog box goes away and you can go find your data.

Now you can review and share the data in the drawing in a format that others can use.

Homework:

Go back to work and see what existing data is being used in other parts of your projects that you can use or supply with your data to improve accuracy and efficiency of the project.

Hope you have enjoyed the session and that you picked up some tools to use in the future.

Thank you for your time.

See the following for additional information on this topic.

Additional resources:

The AutoCAD help function: search the following topics with cut and paste.

For help with defining an attribute

" Attribute Definition Dialog Box"

For help with modifying the attribute

" Edit Attribute Dialog Box"

For help with setting up the data extraction template file (old school)

" About Setting Up an Attribute Extraction Template File"

For help with data extraction (new school)

" Data Extraction Wizard"

Autodesk University website has classes from other years.

- A tribute to Attributes: Adding Intelligence to your drawing

http://au.autodesk.com/au-online/classes-on-demand/class-Catalog/2014/autocad/ac6496#chapter=0

- Excel-ing-with-Autocad-no-programming-required

http://au.autodesk.com/au-online/classes-on-demand/class-catalog/2012/autocad/excel-ing-with-autocad-no-programming-required#chapter=0

- Lisping on purpose part 1

http://au.autodesk.com/au-online/classes-on-demand/class-catalog/2013/autocad-design-suite/cm1755-l#chapter=0

- Lisping on purpose part 2

http://au.autodesk.com/au-online/classes-on-demand/class-catalog/2013/autocad-design-suite/cm1757-l#chapter=0

Autocad User Group International: you can join for free and be able to access older AU classes

- Batch processing an "unlimited" amount of drawings.

http://forums.augi.com/showthread.php?148307-CP12-3L-Changing-Hundreds-of-AutoCAD-Drawings-in-a-Hurry