Goal of this function:
-Provide AutoCAD users with a versatile and powerful means to solve many dynamic block situations that cannot be done currently out-of-the-box (OOTB). At the very least, it is an example that customized coding can be used to accommodate for the unexpected needs not available to dynamic blocks at the latest release level. It is expected that you either suggest functionality and report bugs or develop your own code.
-It is not intended to make any capital gains or have any spyware attached. Consider this is as "freeware".
-Please continue to let AutoDesk know you want what these "workarounds" can do to be built into future releases of AutoCAD.

Before Getting Started:
-By enabling the utility, you accept any potential risks and that it is provided as is.
-Express Tools required to be installed for the "dbr-command" functions to work (described later).
-The utility works when the user selects a dynamic block grip and makes a change. The utility will check the new values of the block and make the changes defined in the matching DBR file.
-It is not designed to work with changes made from the Properties panel or toolbar controls.
-Flip grips and alignment grips do not trigger the reactor (use a lookup table to control the flip parameter).
-Advise using attribute justifications that use a combination point like "TL" or "BR" to avoid text shifting (detailed later).
-This is not tested on nested block situations.
-This has not been tested on Annotative blocks introduced in AutoCAD2008.
-This is coded using LISP, don't expect it to work on AutoCAD LT.

Setup:
1. Unzip the *.DBR file and DynamicBlockUtil.fas to desired support directories.
2. Unzip the drawing file to a folder to test from.
3. Add the "DynamicBlockUtil.fas" to your Startup Suite (use appload command to find it).
4. Run the command "DynBlockUtil". This will enable/disable the utility and set the path location you are storing the DBR files. Default location is "C:\\". It will also reload changes made to DBR files.
5. Open the unzipped drawing to test.
6. Revise the formula in the *.DBR file with a text editor to suite your additional requirements.

Functions to aid in setting up:
(DBRpath *pathname*) Function to set the path where pathname is a string pointing to the DBR files folder.
(DBRset *T or F*) Function to turn on/off the function where *T* equals on and *F* equals off.
(DBRreload) Function to reload all DBR files. This is also provided as a command.

What is a DBR file for:
DBR files contain the user definable equation the DynamicBlockUtil.fas utility uses to control specified blocks. The name of the DBR file must match the name of the block to be applied to (one DBR file per effective block name). It is setup as a lisp formula using balanced parenthesis to enclose each IF THEN statement. The order the equations are placed can affect the final output. The IF portion of the equation can be several IF statements and must all be met to operate THEN, the THEN portion of the equation can be several operations. You can have multiple IF THEN statements.

Sample DBR equation:
```
(((IF IF) (THEN THEN)) ((IF) (THEN)) ((IF IF) (THEN)))
```

TIP: To help in figuring out long equations, you can first set it up like this:
```
(
 ((IF IF) (THEN THEN))
 ((IF) (THEN))
 ((IF IF) (THEN))
)
```

But it must be reduced to a single continous line when finished. Sample Putting it all together:

```
(((("Parameter" "operator" value) ("Parameter" "operator" "value")) (("Parameter" value) ("Attribute" "value")))
((("Parameter" "operator" value)) (("Attribute" "value"))))
```

Sample IF statements:
("Parametername" "operator" numericalvalue)
("Parametername" "operator" "stringvalue")
("Parametername" "operator" (lispformula))
("Parametername" "operator" (parametervalue "name")) Use this to compare to the value of another parameter.

Sample THEN statements:
("Parametername" numericalvalue)
("Parametername" "stringvalue")
("Parametername" (lispformula))
("Parametername" (objectvis *visibility objects*))
("Parametername" (objectcolor *colorindex objects*))
("Parametername" (objectoffset *dist sdist objects*))
("Parametername" (objectLinearArray *dist sdist angle objects*))
("Parametername" (dbr-command "Command"))
("Attributename" "stringvalue")
("Attributename" (justifyattribute "L"))

Parameter and Attribute name notes:
-These are string values and must be enclosed in quotes.
-Must match the names defined in the block, including correct Caps and spaces.

Numericalvalue notes:
-Numerical values must be formatted as decimal units 1.0.
-Do not use quotes on them.
-Know that the parameter name uses numerical values.

Stringvalue notes:
-String values must be enclosed in quotes.
-Know that the parameter name uses string values.

Operator notes:
-Operators give the statement equation conditions.
-Operators must be enclosed by quotes "". For example ">=".
-Accepted operators are: "=" "<" ">" ">=" "<=".

Lispformula notes:
-Primarily provided to give the statement additional decision making abilities based on other parameter values, sysvar conditions and other factors.
-Two built in functions have been provided to evaluate the block rotation angle (dbr-rotation) and block scale (dbr-scale) for the lisp formulas.
-(dbr-rotation) is evaluated to radians.

Justifyattribute notes:
-This function alters attribute justification without moving the text around the base point. (Thanks to Tim Willey for pointers on how to do this http://discussion.autodesk.com/thread.jspa?messageID=5421876.)
-Use the following abbreviation chart to change the attribute justification:

"L"	= Left
"M"	= Middle
"R"	= Right
"C"	= Center
"F"	= Fit
"A"	= Align
"TL"	= TopLeft
"TC"	= TopCenter
"TR"	= TopRight
"ML"	= MiddleLeft
"MC"	= MiddleCenter
"MR"	= MiddleRight
"BL"	= BottomLeft
"BC"	= BottomCenter
"BR"	= BottomRight

Objectvis notes:
-(objectvis *visiblity objects*)
-Use to turn off/on the visibility of objects without using visibility states.
-For *visibility*, to turn objects on, use *:vlax-true*, to turn off, use *:vlax-false*. (Do not enclose in quotes.)
- The *objects* defines what objects are to be affected. Format like this: (list 0 1 2 3). Each number represents an object ID number in the block. Attributes and Parameters can not be included in this list.
-To get the nested object ID number, create your block, save it, insert a copy into the drawing space and use the included "NestedID" command and pick over the object you need the ID number for. Attributes will not return an ID number.

Objectcolor notes:
-(objectcolor *colorindex objects*)
-Use to change the color of objects within the block.
-*Colorindex* is an integer value representing any ACAD color 1-256. (Do not enclose in quotes, do not use decimal units.)
- The *objects* defines what objects are to be affected. Format like this: (list 0 1 2 3).

Objectoffset notes:
-("Parametername" (objectoffset *dist sdist objects*))
-The value of the "Parametername" will determine the overall length offsetting will be repeated to.
-The first *dist* is a numerical decimal value indicating the distance to offset the objects by.
-The second *sdist* is a numerical decimal value to adjust the offset initial distance to start offsetting from. This value helps to adjust when the next offset should take place when the parameter distance is in between offset increaments. This value should typically be the arc's/circle's radius value.
- The *objects* defines what objects are to be affected. Format like this: (list 0 1 2 3).
-If flip parameters affect the objects being offset, the flip state will need to be checked in an IF statement, and use negative *dist* value in the corresponding THEN statements.
-This function will offset the results of objects included in the list of other "objectLinearArray" "objectPolarArray" and "objectoffset" functions.

ObjectLinearArray notes:
-("Parametername" (objectLinearArray *dist sdist angle objects*))
-Use to array a list of objects in a linear direction.
-The value of the “Parametername” will determine the overall length the arraying will be repeated to.
-The first *dist* is a numerical decimal value indicating the distance to array the objects by.
-The second *sdist* is a numerical decimal value to adjust the array initial distance to start arraying from. This value helps to adjust when the next array should take place when the parameter distance is in between array increments.
-The *angle* indicates the direction the objects are to offset from 0 radians. Angular parameters can be used to control this.
-The *objects* defines what objects are to be affected. Format like this: (list 0 1 2 3).
-This function will array the results of objects included in the list of other “objectLinearArray” “objectPolarArray” and “objectoffset” functions.

DBR-command notes:
-Use to initiate an AutoCAD command on the block.
-Recommend using a lookup table to issue this function.
-Recommend including a value in the lookup table that won't invoke any dbr-command action. In the formula, reset the lookup table to that value after it issues Rotate/Move/Scale/Copy.
-Following commands are possible:

"Explode"	(Note: In this version, this command will even explode blocks that are defined to not be exploded.)
"Resetblock"	
"Rotate"	(Note: In this version, following items work from the block insert point.)
"Move"	(Note: In this version, following items work from the block insert point.)
"Scale"	(Note: In this version, following items work from the block insert point.)
"Copy"	(Note: In this version, only asks to create one copy per command.)

Revision Notes:

1.00 Initial build
1.10 Fixed a bug
1.20 Revised setup function
1.30 Added ability to put string values to attributes
1.40 Added ability to use parameter value in IF statement
1.50 Added ability to use lisp formula in IF statement
1.60 Added ability to use lisp formula in a THEN statement
1.70 Added ability to change attribute justification
1.75 Accounted for parameters using short integer variant types (flip parameters)
1.76 Added "vl-dbrcmnd" functions
1.80 Added "objectvis" function
1.81 Fixed issue when Dynmatch component is not set
1.82 Minor changes
1.83 Fixed an issue with flipping more than one attribute under single then statement
1.84 Changed "vl-dbrcmd" to "dbr-command", added "dbr-scale" and "dbr-rotation"
1.85 Added "objectcolor" function.
1.86 Added “objectoffset” function. Added additional setup functions.
1.87 When blocks using the “object* “ functions, “Pastclip” & “Pasteblock” commands post process the blocks to appear co
1.88 Addressed same issue as 1.87 but also for when blocks are placed either through “insert” command or the Tool Palet
1.89 Addressed same issue as 1.88 but also for when blocks are drag-n-dropped for the Design Center.
1.90 Addressed a calculation issue if block is scaled for objectoffset.
1.91 Added “objectLinearArray” function. Addressed additional pasting issues.