

White Paper
Dynamic Blocks in Autodesk® AutoCAD 2006
Part 3 of 3: Advanced Features of Dynamic Blocks
By Ellen Finkelstein, EllenFinkelstein.com,
Author of *AutoCAD 2006 and AutoCAD LT 2006 Bible*

This is the third (and last) of three white papers on dynamic blocks in AutoCAD 2006. This white paper introduces some of the special techniques and advanced features that you can use to make complex dynamic blocks. The first white paper explained dynamic block basics and included a quick start tutorial. The second white paper documented the parameters, actions, and parameter sets available for creating dynamic blocks. If you're new to dynamic blocks, first read Parts 1 and 2.

Naming parameters and actions

Parameters have a label that appears when you edit the block in the Block Editor. By default, the label is generic and if you create more than one of the same parameter, they are numbered consecutively. For example, if you create two linear parameters, they are labeled Distance and Distance1. To help make your labels more meaningful, you can, and probably should, change them. If your linear parameter measures the width of a door, you could change the label to Door width.

To change a parameter label, use the Label option at the first parameter prompt when you create the parameter. Or, you can create the parameter with the default label, select the parameter, and change the label in the Properties palette.

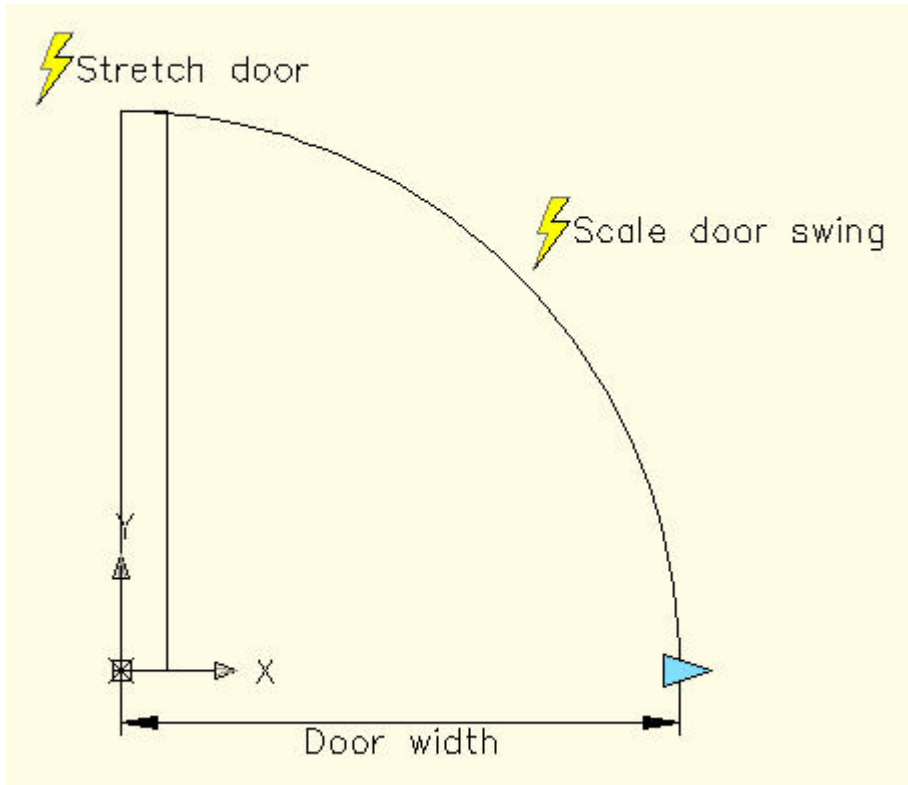
Note that parameters also have a name, which doesn't appear in the block editor and simply denotes the type of parameter. For example, if you create a linear parameter, its name is Linear. If you create a second linear parameter, its name is Linear1. Although you can change a parameter name in the Properties palette, it's probably best not to do so, because these names can help you understand the type of parameter you've used when authoring the dynamic block.



Property Labels	
Distance label	Door width
Distance descri...	
Parameter name	Linear

A parameter has both a name and a label.

Actions have names only (no labels), and you may want to change these to make them more meaningful. If you create more than one of the same action type, they are also numbered consecutively (for example, Stretch and Stretch1) as needed. If your stretch action changes the width of the door, you could change the action name to Stretch door.



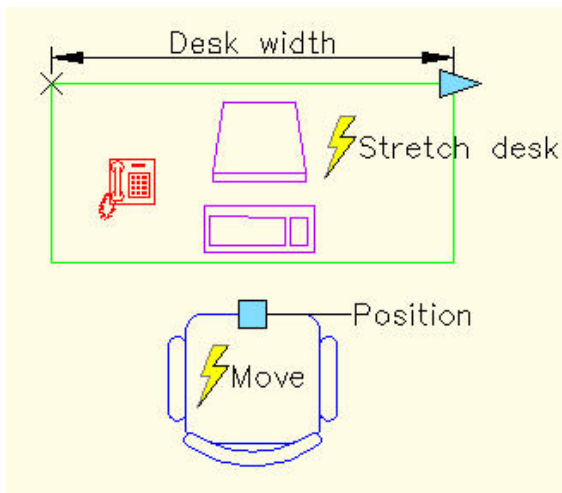
The linear parameter's label is Door width. Two actions are named Stretch door and Scale door swing.

Actions also have a type (such as Stretch). The action type appears in the Properties palette, but you can't change it.

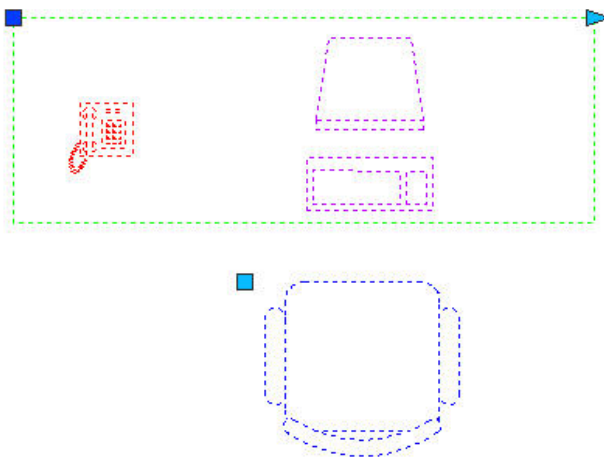
Selecting objects for actions

When you add an action to a parameter, you must select objects to create a selection set for the action. Often, you want to select not only drawing geometry, but the parameter as well. In fact, sometimes, you need to select other parameters on other objects to get the results you want.

For example, let's say that you have a desk with a linear parameter and a stretch action so that you can stretch the width of the desk. You want to be able to move the chair independently (so it has a point parameter and a move action) but you also want it to move to the right when you stretch the desk to the right. To accomplish this, when you select objects for the desk's stretch action, you need to include the chair and its point parameter. Therefore, you need to create the parameter for the chair before you create the stretch action for the desk. In general, if you want to include another object in an action, that has a parameter of its own, you should create the two parameters first, before adding the first action.



You can create different results by changing the selection set for the desk's stretch action.



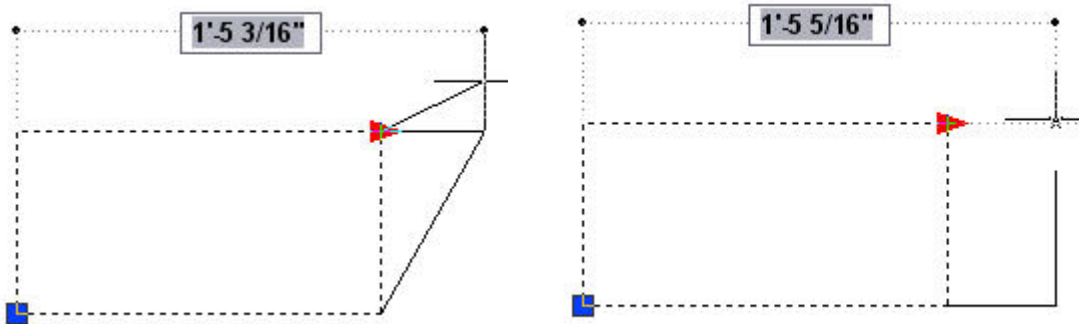
If the selection set for the desk's stretch action includes the chair but not its parameter, when you stretch the desk, the chair moves, but its grip is left behind.

A common use for including a second parameter in the selection set of an action is for a base parameter. Use a base parameter if you want the base point of the block reference to remain in the same position relative to the block, for example, at the lower-left corner of a rectangle, even when that position moves. If an action (such as stretch or move) moves that lower-left corner, then you should include the base parameter in the selection set of that action, along with its parameter. Then the base point of the block will always remain in its proper location when you edit the block in a drawing.

Specifying stretch frames

Stretch actions require you to specify a stretch frame before selecting objects. The purpose of the stretch frame is not always obvious at first. This stretch frame specifies that *part* of geometry that is included in the action. You can specify it with a selection or

crossing window. Then you can select objects with a crossing window or by picking the objects. Often these two windows are very similar (although they need to use slightly different points). The example shows the result of two different stretch frames while stretching a rectangle in a drawing. In the example on the left, the stretch frame only includes the upper-right corner of a rectangle, so only that corner stretches. In the example on the right, the stretch frame includes the entire right side so the entire side stretches.

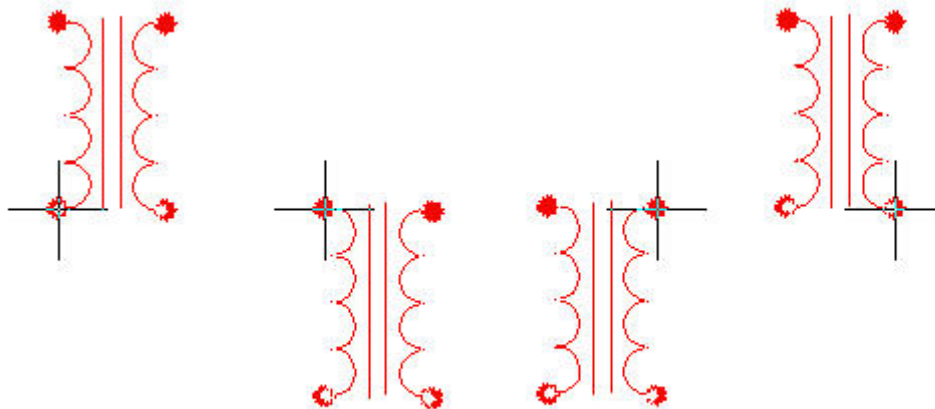


Upper-right corner included in stretch frame

Entire right side of rectangle included in stretch frame

Cycling through insertion points

If your block has several actions, or alignment or base point parameters (which require no action), the block has several grips. You can cycle through these grips when you insert the dynamic block by pressing the Ctrl key.

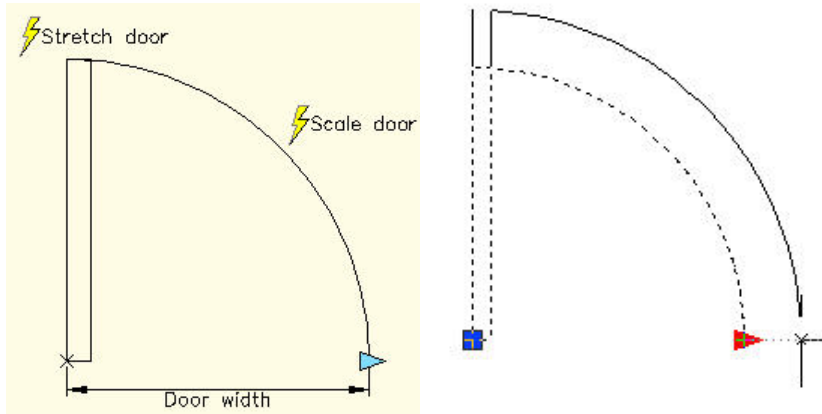


Pressing the Ctrl key while inserting the block moves the insertion point from grip to grip, although the cursor position does not change.

By default, insertion cycling is on. You can turn it off for any grip by selecting just the grip and changing the Cycling property in the Properties palette to No. You can also use the BCYCLEORDER command in the Block Editor to change the cycling order of the grips in the Insertion Cycling Order dialog box.

Changing the direction of an action

For move, stretch, and polar stretch actions, you can change the direction of an action relative to its parameter. For example, you may want to drag to the right but make an object stretch vertically at a 90 degree angle. This is called an *angle offset*. A common use is a door block that you stretch to the right to change the width of the door opening. The door itself should vertically stretch the same amount and the arc swing should scale at the same time. Note that if you just scaled the entire door, the door itself would get thicker as you widened the door opening—a result that you don't want.



Door in block editor

Editing the door in a drawing

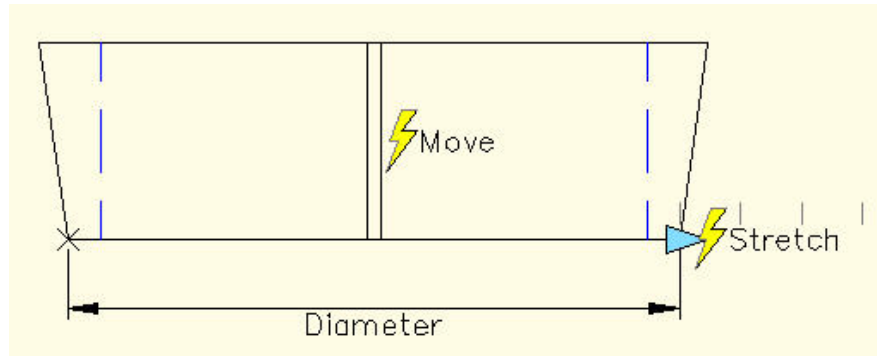
To create this angle offset, you author the dynamic block shown in the figure as follows:

1. Create the door by drawing a rectangle and an arc. Make a block from these objects and use the lower-left corner of the door as the base point.
2. Open the Block Editor and choose the door to edit.
3. Create a linear parameter as shown on the left side of the figure, to represent the door width. Select the parameter, right-click it, and choose Grip Display > 1 to show only a grip on the right end of the parameter.
4. Add a stretch action to the linear parameter. For the stretch frame, specify a window that covers the top part of the rectangle. When you select objects, select everything except the arc.
5. Select the stretch action. In the Properties palette, change its Angle Offset value to 90.
6. Add a scale action to the linear parameter and select the arc.
7. Save the block and close the block editor. When you grip-edit the block, click the stretch grip to change the door opening. The door stretches in the 90 degree direction and the arc scales correspondingly.

You can also create the angle offset while creating the action. At the *Specify action location* or *[Multiplier/Offset]:* prompt, use the Offset option and set its value to 90.

Keeping objects centered

Many blocks have centered components that need to remain centered as you stretch the entire block. For example, this valve part may come in several diameters, but the hole must always be centered.



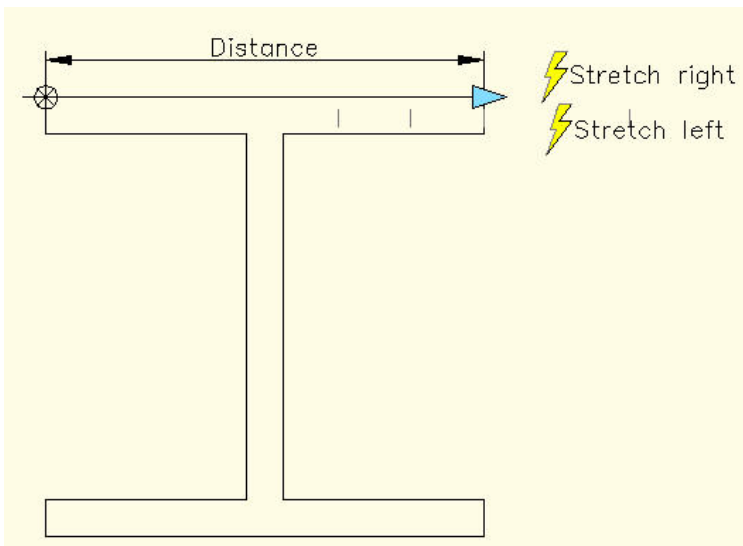
This valve part's central hole needs to remain centered regardless of the diameter of entire part.

To keep a component centered, you use a distance multiplier of 0.5, so that the component always moves half the distance of the rest of the block. In this example, you can use a stretch action to change the diameter (a linear parameter) of the entire block. The two vertical lines in the middle (the central hole) have a move action attached to the same linear parameter, with a distance multiplier of 0.5.

To create a distance multiplier, you add an action as usual. After the prompt to select objects, you see the *Specify action location or [Multiplier/Offset]:* prompt. Use the Multiplier option to specify the distance multiplier. You can also change the multiplier afterwards by selecting the action and using the Properties palette.

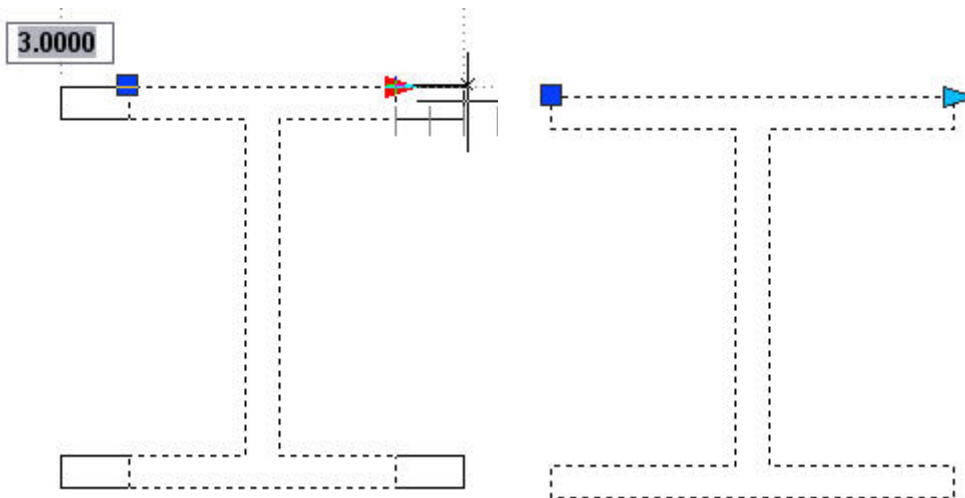
Stretching a block equally in opposite directions

Another way of handling the previous example would be to stretch the two sides and leave the middle alone. Some blocks always need to remain symmetrical, so if you stretch one side, the other side needs to stretch an equal amount.



An I-beam in the block editor.

The I-beam has a linear parameter and two stretch actions, both attached to the same grip point on the right side of the parameter. Because you don't need the grip on the left, select the parameter, right-click it, and choose Grip Display > 1. The Stretch Left action's angle offset is 180 degrees. A base point parameter at the upper-left corner, which is included in the selection set of the Stretch Left action, keeps the base point at that corner, even when that corner moves in the stretching action. As you drag the rectangular grip in a drawing, both sides of the I-beam stretch by the same amount.



Stretching the I-beam in a drawing.

Chaining parameters

Sometimes, you need one action to cause another action to occur. If those two actions can share a parameter grip, then you can accomplish this easily. For example, the door shown in the “Changing the direction of an action” section has two actions: a scale action that scales the arc and a stretch action that stretches the door. Because these two actions can share one parameter and grip, when you stretch the door, you also scale the arc.

However, sometimes your geometry is more complex and you need more than one parameter. Yet you still want one action to activate another action. You do this using the chaining feature. Because you want one action to activate another one, you need two actions and two parameters. The principles of chaining are as follows:

- Parameter 1 has an action, whose selection set includes parameter 2, in addition to any other objects it needs to function. (Note: If the action is a stretch action, the stretch frame also needs to include parameter 2.)
- Parameter 2 has an action; parameter 2’s chaining property is set to Yes.

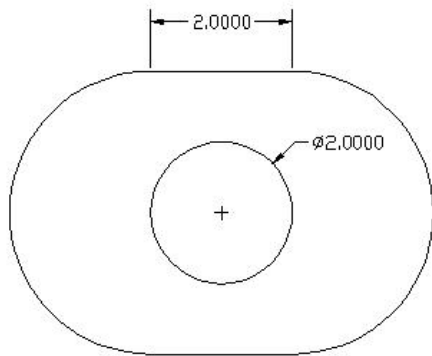
When you grip-edit the block using the action of parameter 1, the action of parameter 2 is activated at the same time.

You need to set up chaining in a logical order:

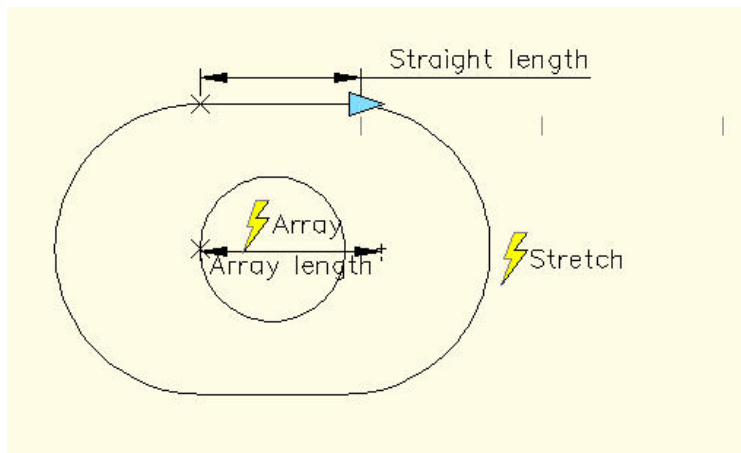
1. Decide which action you’ll want to grip edit. This is the action that will activate another action. You can call it the main action.
2. Decide on the parameters you’ll need and their actions.
3. Create the parameters first.
4. Create the main action and attach it to its parameter (parameter 1).
5. When you specify the main action’s selection set, be sure to include the parameter of the second action. (Don’t include the objects that are in the selection set of the second action.)
6. Create the action for parameter 2.
7. Set the chaining property of parameter 2 to Yes.

Tip: Because parameter 2’s action is automatically activated, it doesn’t need any grips. To avoid confusion during editing, you can remove all of its grips. Select it, right-click it, and choose Grip Display > 0.

In the example below, you want to stretch the cut-out sheet metal plate and array the small circle (the cutout) at the same time. You want the circles to have 0.5 units between them and you also want to maintain equal spacing at either end of the plate.



A cut-out sheet metal plate. (The dimensions aren't part of the block.)



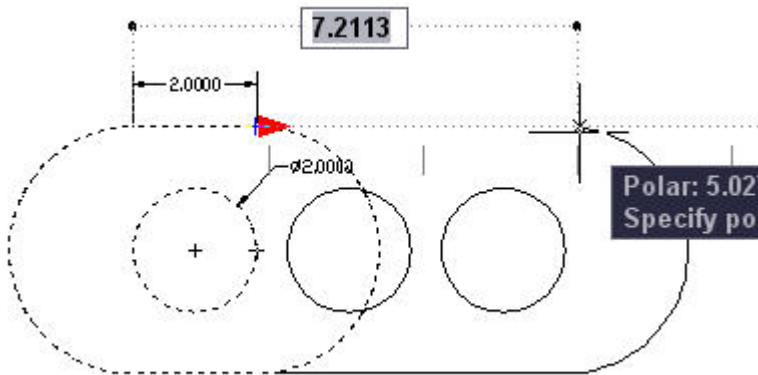
The block in the Block Editor.

The Stretch action is the main action that you want to be able to grip edit. As you stretch to lengthen the plate, you want the circle to array. Here's how to set up this block:

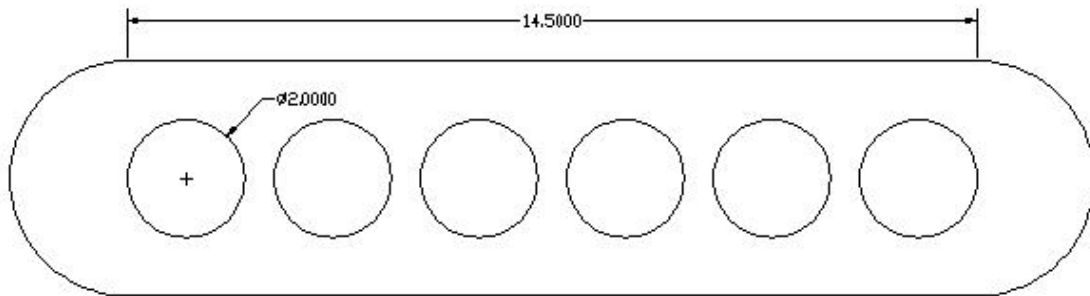
1. Create the block and open it in the Block Editor.
2. Add a linear parameter (parameter 1), called Straight Length, along the straight part of the object. (The plate is a polyline.) In the example, this parameter is 2 units long and has an increment value set with an increment of 2.5, a minimum of 2 and a maximum of 20. Set the grip display to 1.
3. Add a second linear parameter (parameter 2), called Array Length. This parameter is 2.5 units long.
4. Create the stretch action for the Straight Length parameter. When you specify the stretch frame, include the entire Array Length parameter. When you select objects, again include the array length parameter, but not the circle. (Because the Array Length parameter is inside the circle, if you use a crossing window, you need to include the circle in the crossing window and then use the Remove option to deselect the circle. Another method is to just pick the Array Length parameter) Set the grip display to 0 and the Chain Actions property to Yes.

5. Create the Array action for the Array Length parameter. In the example, the column offset is 2.5 units.
6. Save the block and close the Block Editor.

When you grip edit the block in the drawing, you see only one grip, so you don't have to remember which grip to stretch.



Stretching the block in the drawing also arrays the circle.



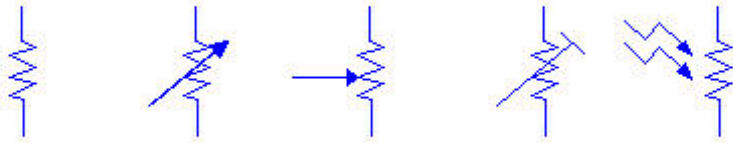
The block is now stretched and the circles are arrayed.

As you drag parameter 1 a specific distance and angle, parameter 2 moves the same distance and angle. For this reason, chaining is useful when you want to maintain a constant relationship between two components in a block. In the example, the relationship between the circles and the ends of the plate stays the same, so that the circles are always centered inside the plate.

Using visibility states

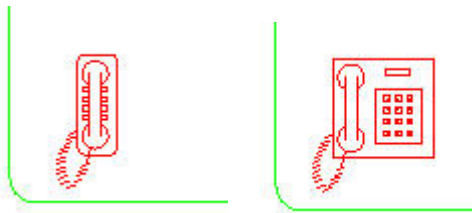
A Visibility parameter turns the visibility of a block component on and off. The visibility parameter doesn't take an action. A dynamic block can only have one visibility parameter.

You define visibility states, each of which is a variation of visibility or invisibility. You can make one or more components visible or invisible. In the example, all the resistor variations are one dynamic block. The components that are different are specified as visible or invisible for each of five visibility states. You've just combined five blocks into one!



Fixed value, variable, adjustable, temperature variable and photo resistors—visibility variations of one block.

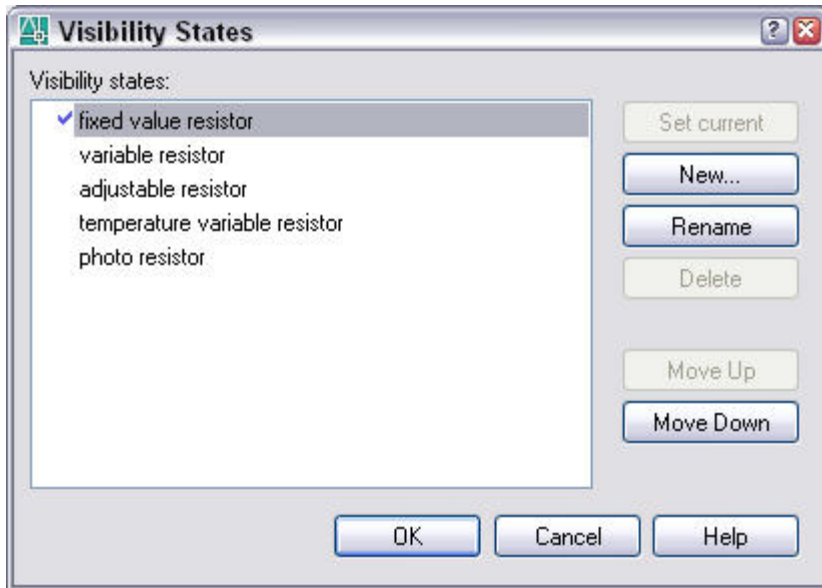
You can use visibility states to switch among objects. For example, you might want to have two variations of a telephone on a desk—a single-line and a multi-line phone. By putting these two phones in the same location and setting two visibility states, you can choose which phone to display when you edit the block.



These two phones are in the same location in the block, one on top of the other. You can choose which to display using the visibility parameter.

To add a visibility parameter, follow these steps:

1. Create a block that contains all the components that you need for all the visibility states.
2. In the Block Editor, choose Visibility Parameter from the Parameters tab of the Block Authoring palettes, and place it near the items you want to make visible or invisible.
3. Choose Manage Visibility States on the Block Editor toolbar (or double-click the visibility parameter) to open the Visibility States dialog box.

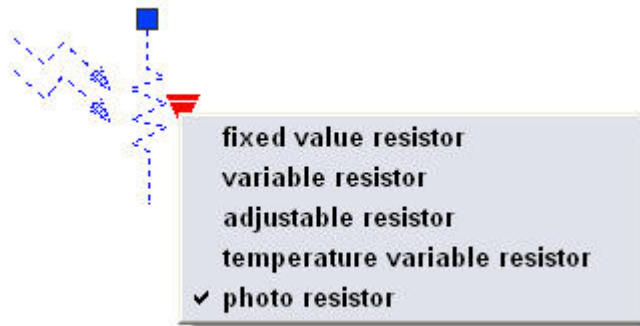


The Visibility States dialog box.

4. Click the default visibility state, called VisibilityState0. Enter the name of your first visibility state and press Enter.
5. Click New to open the New Visibility State dialog box. Enter the second visibility state name and click OK. Repeat for all the visibility states that you want to create.
6. Click OK to close the Visibility States dialog box. (The visibility state at the top of the list will be the default state when you insert the block. You can choose a visibility state and use the Move Up button to move it to the top of the list.)
7. Choose the first visibility state from the Visibility States drop-down list at the right side of the Block Editor toolbar. Select all the components that you want to be invisible for that state. Click the Make Invisible button on the Block Editor toolbar.
8. Repeat the previous step for each state. You can also select objects and make them visible by clicking the Make Visible button. If you need to select an object that is invisible, click the Visibility Mode button, which displays invisible objects in gray so you can see and select them.

When you're done, select each state from the drop-down list and check that it displays the correct objects. Save the block and close the Block Editor.

When you select the dynamic block in the drawing, click the down arrow to drop down the list of visibility states. Choose a state to display it.

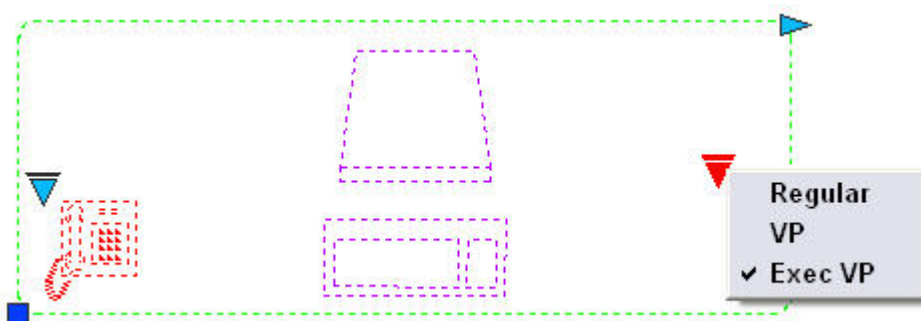


Choosing a visibility state in a drawing

Visibility states are a powerful, yet simple way to add great flexibility to a block.

Using lookup parameters and tables

A lookup parameter pairs with a lookup action to create a table that matches labels with values. Lookup tables are great when you want pre-set sizes for a block. For example, you might have a part that comes in three sizes. When you insert and grip-edit the block, you just choose a size from a drop-down list.



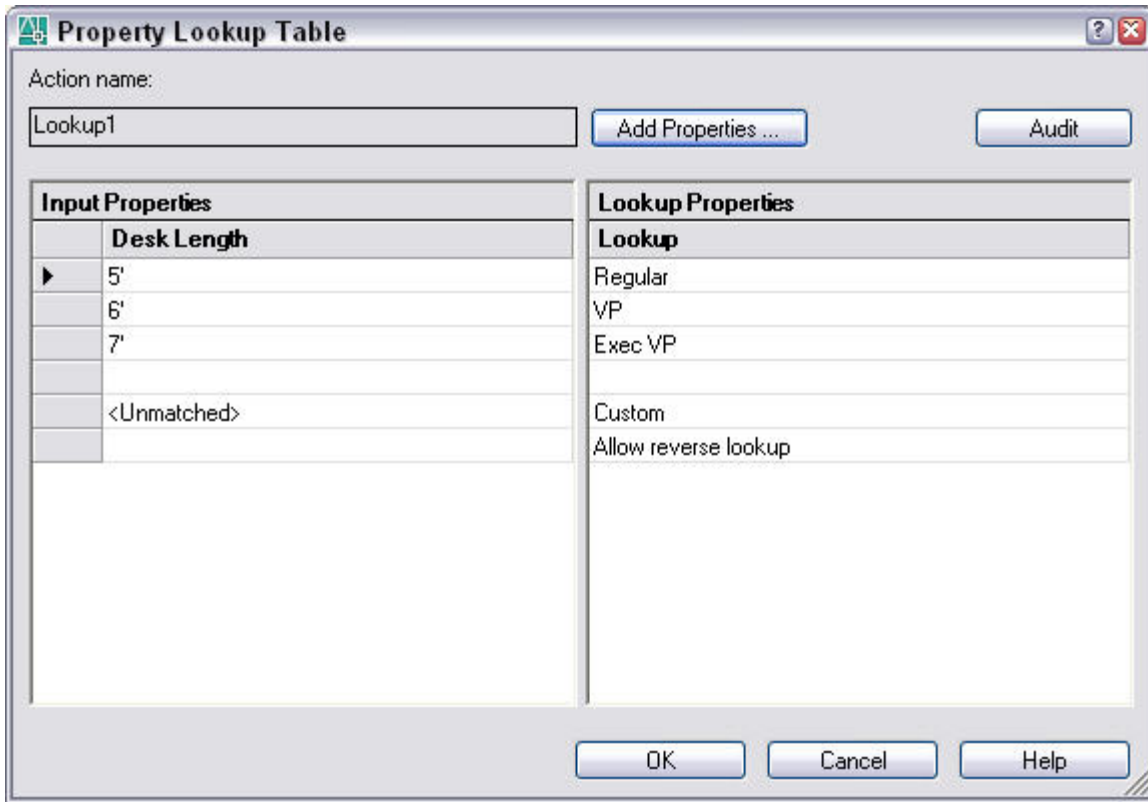
Choosing a desk length from a lookup table.

The lookup parameter and action are based on another parameter and action, such as a linear parameter and stretch action. You may want to use a value set to preset the values that you can use in the lookup table, but you can also set the values when you create the lookup table.

Here's how to create a lookup parameter and action:

1. In the Block Editor, add the parameter and action that you want to use as the basis for the lookup parameter and action. The example shown above uses a linear parameter and a stretch action. If you add a value set (list or increment), the measurements are available when you create the lookup table.
2. From the Parameters tab of the Block Authoring palettes, add a lookup parameter.
3. From the Actions tab, add a lookup action. The Property Lookup Table dialog box opens.

4. Click the Add Properties button, choose the parameter you want to work with, and click OK. You're back in the Property Lookup Table dialog box.



The Property Lookup Table dialog box

5. If you have values from a value set, click the first row of the Input Properties side and click the drop-down arrow that appears. Choose the first value. Otherwise, just enter values on each row. Click the corresponding row on the Lookup Properties side and enter the label that you want for that value.
6. Click the lower-right cell in the dialog box, which says Read Only by default. Choose Allow Reverse Lookup. For this to work, all of the rows in the table must be unique. You need to use this option in order to choose a value from a drop-down list when you insert the block.
7. Click OK.
8. Save the block and close the Block Editor.

Now, when you grip-edit the block, you'll see a down arrow. Click the arrow to choose one of the labels and apply its corresponding value to the block.

Tip: You can extract parameter values like attributes. For example, if you choose a desk length of 7 feet, you can extract that value. You can also create invisible values that you can extract using a lookup table. To do so, add a lookup parameter and action. Change the label of the parameter to the property (like an attribute tag) that you want. For example,

you could label the parameter “Chair color.” In the Property Lookup Table dialog box, add the colors on the Lookup Properties side, leaving the Input Properties side blank. In the drawing, you can choose a chair color by clicking the lookup parameter’s down arrow. You don’t see any change in your drawing, but when you extract attributes, the color appears in the output.