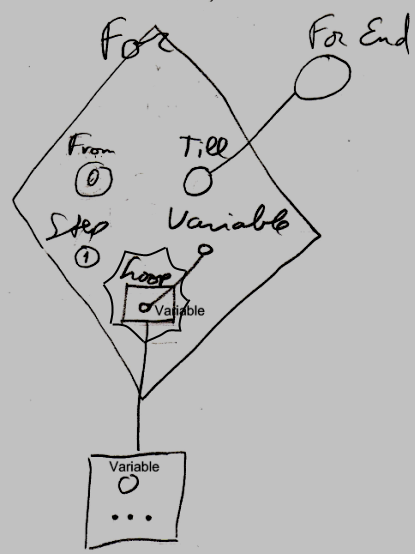
|  |
| --- |
| Circle Language Spec: Execution Control |

## For (range) in a Diagram

The concept of the For statement for value ranges is already covered by the article *For (range)*. This article only explains its expression in a diagram.

Below is an example of the diagrammatic expression of a For statement, that goes through a contiguous range of numbers.

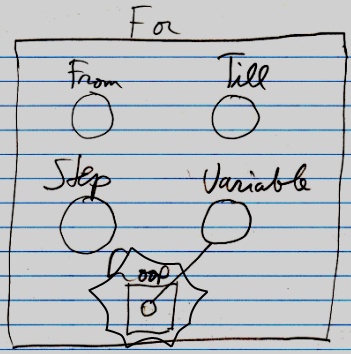


The diamond is a call to the For command. The For loop will go through a range of numbers. The first of the range of numbers is passed to the For command as the From parameter, which is visible inside the diamond as the circle named From. In the example above, the literal value of 0 is filled into the From parameter. This will be the first value of the range. The last value in the range is passed to the For command as the Till parameter. In the example above, the Till parameter is visible inside the diamond as the circle named Till. The Till parameter is a pointer to something defined outside the call to the For command. The Till parameter is pointing to a variable defined outside the call, displayed in the diagram above outside of the diamond, as the circle named For End. The For loop will go through the values 0 till the value indicated by the variable For End. The For loop will increment the current value by 1 on each loop, because the Step argument of the For command is set to the literal value of 1, visible inside the diamond as the circle named Step. The diamond contains another circle, named Variable, which will hold the current value of the range. This current value is passed to the Loop , that will be run multiple times. The Loop procedure is displayed inside the diamond, as the square named Loop. The the Loop procedure is *defined* outside the diamond. The Loop parameter, visible as a square named Loop inside the diamond, points out of the diamond to the larger square just below the diamond, with the ellipsis in it. The ellipsis stands for whatever you fill into it. The larger square below the diamond defines the command that will run multiple times, once for each value in the range. The circle inside the large square is the Variable passed to the Loop procedure. That circle is not tied to the Variable parameter inside the For call, but it is implicitly connected to it, because of the connection between the squares that contain the circles, and the name Variable repeated for each of the circles. (This is due to the rule of *implicit connection through parent.* See the article *Automatic Containment*.)

The Loop parameter is placed inside a nonagon purely for esteatic reasons. It is a single reference to a command, and the nonagon will never contain multiple references to commands, but it is obvious the command in it will be run multiple times. The nonagon stands for multiple, so it is put around the Loop parameter, but it will always contain a single item and you cannot add any more items to it.

In the example above, two of the circles got literal values filled in. One of the circles is tied to something defined outside the diamond. But any of the three From, Till and Step parameters could have had a literal value filled in or could have been pointing to something outside the diamond. The diagram above is just an example. The Loop procedure reference in our example, pointed out of the diamond as well. The loop procedure may as well have been defined right inside the diamond. The Variable of the loop can’t be given a literal value or be set to point to something outside the diamond, because the Variable of the loop is controlled by the For command definition, and can only read.

The *definition* of the For execution control command is part of a system module for execution control commands. The public elements of the definition look like this:



Nothing is filled in yet as the From , Till , Step or Loop . The Variable will be controlled by the For command. It is referenced from the Loop procedure.