Following the text area is a section where you can specify how the breakpoint should behave each time it is encountered. Conveniently, the behavior can depend upon the return value of the custom code entered in the text area. The behavior options are as follows:

- Continue Regardless of Result
- Stop Regardless of Result
- Stop If Result Is True
- Early Return from Caller with Result

The options are mostly self-explanatory. The option that is probably the most powerful is the Stop If Result Is True option. Choosing this option means that the breakpoint will suspend execution only if the return value from the custom code is true.

Imagine how convenient it could be to suspend execution depending on a certain condition. Consider the situation where you're iterating through a large list of objects and performing some calculations on each object. During the development process you notice that one particular object on the calculation is failing. If you're able to isolate the data source so that you're dealing with only a singular object or a small list of objects, you have no problem. However, if you can reproduce the error only when using a large list of objects, and the failing object is toward the end of the list, then you have to set a breakpoint and step through it each time, looking for the particular object on which you know the error is occurring.

This scenario is quite easy to solve if using the conditional breakpoint functionality. If you know the particular scenario that is failing, then you can write a conditional statement so that the breakpoint will suspend execution only when the condition is met.

Listing 7-1 shows a small snippet of simplistic JavaScript code. All the code does is create an array of strings and iterate over the array, and for each string in the array, another string is prepended to the string. This snippet of code shows how to conditionally stop the execution of the script based on a condition you define.

**Listing 7-1. Iterating over a String Array**

```javascript
function testBreakpointProperties() {
  var list = ["one", "two", "three", "four", "five", "six", "seven", "eight"];;

  var item = null;
  for(var i = 0; i < list.length; i++) {
    item = list[i];
    item = "Text is: " + item;
  }
}
```

For this example, you’ll want to suspend execution on only the seventh item in the list, which is the seven item. If you simply set a breakpoint within the iteration, execution will be suspended each time that line of code is encountered, and you’ll have to manually restart the execution each time until you get to the item within the list in which you’re interested. In this example, the list isn’t very long, but if the list were several hundred items long, it would be convenient to halt execution only when a certain condition is met.