Call Stack

Venkman keeps track of the current call stack in the Call Stack window (see Figure 7-30). This window simply shows a stack of the function call list. The function at the top of the stack is the current function; the next function in the list is the one that called the function at the top of the list, and so on. This window is useful when you have a complicated call structure and you want to see the order in which functions are called.

Double-clicking an item in the Call Stack window will bring the view in the Source Code window to the function represented by the clicked item in the call stack. The Local Variables list will update itself to show the variables in context within that function. Note that this doesn’t actually change the order of execution—you’ve merely moved to see where you’ve been, but it doesn’t change where you’re going.

Performance Profiling

As you become more comfortable with Ajax techniques, you’re likely to write more JavaScript than you have previously, especially if you decide to host more application logic on the browser. Doing so raises the possibility that you’ll at some point encounter performance bottlenecks within the JavaScript code.

Sound software development practices teach that there is no point in optimizing code for performance unless a performance problem exists in the first place. Optimized code is almost always harder to read and maintain than code that is not optimized, so unless you truly need to optimize code for performance, avoid doing so.