

Re: java socket i/o with callback/non-blocking

Source: <http://coding.derkeiler.com/Archive/Java/comp.lang.java.programmer/2005-06/msg01314.html>

- *From:* "John C. Bollinger" <jobollin@xxxxxxxxxxx>
 - *Date:* Tue, 14 Jun 2005 11:31:56 -0500
-

Shane Wealti wrote:

I have the O'Reilly Book Java NIO (with the mouse on it) and I'm trying to figure out how to do non-blocking socket i/o. Basically I want to have some subroutine to be called whenever there is data available to read on a socket object without having to poll it. I'm having trouble understanding how to do this in Java and the NIO book isn't really helpful in that regard. Right now the way I have my program written is that I spawn a separate thread that has a while loop that checks the socket for data and then goes to sleep for a while but that doesn't seem like a very efficient implementation. I'm having trouble finding much information on NIO, especially tutorials and code samples that deal with this.

The NIO mechanism for waiting on data without polling is the Selector. You obtain a new instance via `Selector.open()`, register your `SocketChannel` with it via the `SocketChannel`'s `register()` method (specifying `SelectionKey.OP_READ` for its interest operations), then invoke one of the Selector's `select()` methods. When the method returns, you can query its selected keys set to determine which channels are ready for which operations, and do whatever you want to with the results. Typically you would put all this in a loop.

Now, all that may be dramatic overkill if you only have one channel to worry about. You still need to run it in its own thread (or at least, it's much simpler to do so), and all it really saves you is guessing at how long to wait for data. Selectors are really intended for handling multiple channels with a single thread. Note also that the technique described is blocking, not non-blocking, as indeed is the technique you described even if the channel is in non-blocking mode. You can use a Selector in a non-blocking way (see `Selector.selectNow()`), but I don't yet see how that would help you in your particular situation.

For just one socket, you are probably much better off to dump all the NIO stuff, and simply use blocking I/O in its own thread. (Even with multiple sockets this is often a viable approach, with one thread per socket.) What you are doing now is an inefficient simulation of that, and the Selector-based scheme for a single channel is a somewhat less inefficient simulation of it. Non-blocking I/O sounds good, but it leaves you with

Re: java socket i/o with callback/non-blocking

questions about what the thread is to do when it's not performing I/O and how t