return;
}
finally {
  if (fMainListenSocket == null) {
    System.err.println("Couldn’t create a new ServerSocket!");
    return;
  }

  if (fDebugOn)
    System.out.println("fMainListenSocket initialized on port..." + fPort);
}

try {
  while (fContinueListening ) {
    if (fDebugOn) System.out.println("server accepting...");
    Socket clientSocket = fMainListenSocket.accept( );//this blocks!

    if (clientSocket != null) {
      HttpConnectionMgr mgr = new HttpConnectionMgr(clientSocket);
    }
  } //while fContinueListening
} //try
  catch (Exception loopEx) {
    System.err.println("main loop ex: " + loopEx);
  }
}  // HttpdMulti.run
}/*! class HttpdMulti */

We now have a fairly complete Web server. You should be able to use this example as the core of other servers you may write. Obviously, there are other improvements we could make to this server. For instance, we could use ThreadGroups to make it easier to shut down all client connections at once (when the server is being shut down, for example).

**New for JDK 1.1: New Socket Options**

In JDK 1.0 there were several major limitations to what you could do with a Socket or ServerSocket. Specifically, there were major limitations as to how you could create and configure these classes.