Static Diagram for Socket Server

<<struct>>
ST_SERVER_INIT

+wPort : WORD
+connections : std::vector<CConnectionSuper*>

CConnectionSuper::ConnectionThread()
  OnConnect()
  While True
    Peek(m_Socket)
    If FIN found Then break
    OnRecv()
  WEnd
  OnClose()

---

main()
  server : CServer
  server.StartUp()

  Do Until Terminate Signal Found
    Sleep(1000)
  While

  server.ShutDown()
exit

CServer

-m_vecConnectionInstance : std::vector<CConnectionSuper*>
-m_queReady : std::queue<CConnectionSuper*>
-m_setConnected : std::set<CConnectionSuper*>
-m_queDiscon : std::queue<CConnectionSuper*>
-m_ListenSocket : SOCKET

+int StartUp(in stInit : ST_SERVER_INIT)
+void ShutDown()
+void Broadcast()
-void AcceptThread()
-void DisconnectThread()

CConnectionSuper

-m_pServer : CServer*
-m_Socket : SOCKET

+int Establish(in acceptedSocket:SOCKET, in pServer : CServer*)
+int Send(in const LPBYTE pData, in size_t tSize)
+int Recv(in LPBYTE pBuffer, in size_t tBufferSize)
-void ConnectionThread()
#virtual void OnConnect() = 0()
#virtual void OnRecv() = 0()
#virtual void OnClose() = 0()

1   *

CServer::AcceptThread()
  m_ListenSocket = socket()
  bind(m_ListenSocket, port:N)
  listen(m_ListenSocket)

  While True
    newConnectionSock = accept(m_ListenSocket)
    If on accept error Then break

    newConnection = queReady.pop()
    newConnection.Establish(newConnectionSock, this)
    setConnected.insert(newConnection)
  WEnd

  closesocket(m_ListenSocket)

CChatConnection

+void OnConnect()
+void OnRecv()
+void OnClose()

IConnectionInterface

CStreamConnection

+void OnConnect()
+void OnRecv()
+void OnClose()

IConnectionInterface