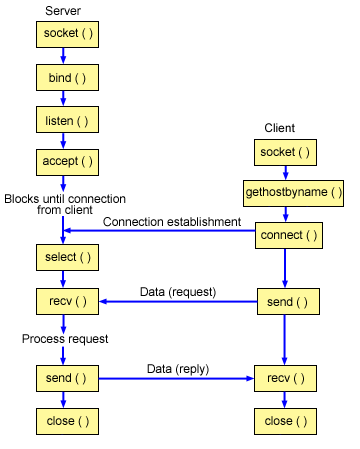
**Create a connection-oriented socket**

These server and client examples illustrate socket APIs written for a connection-oriented protocol such as Transmission Control Protocol (TCP).

The following figure illustrates the client/server relationship of the sockets API for a connection-oriented protocol.



**Socket flow of events: Connection-oriented server**   
The following sequence of the socket calls provide a description of the graphic. It also describes the relationship between the server and client application in a connection-oriented design. Each set of flows contain links to usage notes on specific APIs. If you need more details on the use of a particular API, you can use these links. The[Example: A connection-oriented server](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/rzab6/rzab6xconoserver.htm#xconoserver) uses the following sequence of function calls:

1. The [**socket()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/socket.htm) function returns a socket descriptor representing an endpoint. The statement also identifies that the INET (Internet Protocol) address family with the TCP transport (SOCK\_STREAM) will be used for this socket.
2. The **[setsockopt()](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/ssocko.htm" \t "_self)** function allows the local address to be reused when the server is restarted before the required wait time expires.
3. After the socket descriptor is created, the [**bind()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/bind.htm) function gets a unique name for the socket. In this example, the user sets the s\_addr to zero, which allows connections to be established from any IPv4 client that specifies port 3005.
4. The [**listen()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/listen.htm) allows the server to accept incoming client connections. In this example, the backlog is set to 10. This means that the system will queue 10 incoming connection requests before the system starts rejecting the incoming requests.
5. The server uses the [**accept()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/accept.htm) function to accept an incoming connection request. The **accept()** call will block indefinitely waiting for the incoming connection to arrive.
6. The [**select()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/sselect.htm) function allows the process to wait for an event to occur and to wake up the process when the event occurs. In this example, the system notifies the process only when data is available to be read. A 30 second timeout is used on this select call.
7. The **[recv()](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/recv.htm" \t "_self)** function receives data from the client application. In this example we know that the client will send 250 bytes of data over. Knowing this, we can use the SO\_RCVLOWAT socket option and specify that we do not want our **recv()** to wake up until all 250 bytes of data have arrived.
8. The [**send()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/send.htm)function echoes the data back to the client.
9. The [**close()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/close.htm) function closes any open socket descriptors.

**Socket flow of events: Connection-oriented client**   
The [Example: A connection-orientated client](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/rzab6/rzab6xconoclient.htm#xconoclient)uses the following sequence of function calls:

1. The [**socket()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/socket.htm) function returns a socket descriptor representing an endpoint. The statement also identifies that the INET (Internet Protocol) address family with the TCP transport (SOCK\_STREAM) will be used for this socket.
2. In the client example program, if the server string that was passed into the **inet\_addr()** function was not a dotted decimal IP address, then it is assumed to be the hostname of the server. In that case, use the[**gethostbyname()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/ghostnm.htm) function to retrieve the IP address of the server.
3. After the socket descriptor is received, the [**connect()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/connec.htm) function is used to establish a connection to the server.
4. The [**send()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/send.htm)function sends 250 bytes of data to the server.
5. The **[recv()](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/recv.htm" \t "_self)** function waits for the server to echo the 250 bytes of data back. In this example, we know that the server is going to respond with the same 250 bytes that we just sent. In client example, the 250 bytes of the data may arrive in separate packets, so we will use the **recv()** function over and over until all 250 bytes have arrived.
6. The [**close()**](http://publib.boulder.ibm.com/infocenter/iseries/v5r3/topic/apis/close.htm) function closes any open socket descriptors.

**TCP/IP:**

1. TCP/IP (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet. It can also be used as a communications protocol in a private network (either an [intranet](http://searchwindevelopment.techtarget.com/definition/intranet) or an [extranet](http://searchenterprisewan.techtarget.com/definition/extranet)). When you are set up with direct access to the Internet, your computer is provided with a copy of the TCP/IP program just as every other computer that you may send messages to or get information from also has a copy of TCP/IP.
2. TCP/IP is a two-layer program. The higher [layer](http://searchsoftwarequality.techtarget.com/definition/layer), Transmission Control Protocol, manages the assembling of a message or file into smaller [packet](http://searchnetworking.techtarget.com/definition/packet)s that are transmitted over the Internet and received by a TCP layer that reassembles the packets into the original message. The lower layer, [Internet Protocol](http://searchunifiedcommunications.techtarget.com/definition/Internet-Protocol), handles the [address](http://searchnetworking.techtarget.com/definition/address) part of each packet so that it gets to the right destination. Each [gateway](http://searchnetworking.techtarget.com/definition/gateway) computer on the network checks this address to see where to forward the message. Even though some packets from the same message are routed differently than others, they'll be reassembled at the destination.
3. TCP/IP uses the [client/server](http://searchnetworking.techtarget.com/definition/client-server) model of communication in which a computer user (a client) requests and is provided a service (such as sending a Web page) by another computer (a server) in the network. TCP/IP communication is primarily point-to-point, meaning each communication is from one point (or [host](http://searchcio-midmarket.techtarget.com/definition/host) computer) in the network to another point or host computer. TCP/IP and the higher-level applications that use it are collectively said to be "stateless" because each client request is considered a new request unrelated to any previous one (unlike ordinary phone conversations that require a dedicated connection for the call duration). Being stateless frees network paths so that everyone can use them continuously. (Note that the TCP layer itself is not stateless as far as any one message is concerned. Its connection remains in place until all packets in a message have been received.)
4. Many Internet users are familiar with the even higher layer application protocols that use TCP/IP to get to the Internet. These include the World Wide Web's Hypertext Transfer Protocol ([HTTP](http://searchwindevelopment.techtarget.com/definition/HTTP)), the File Transfer Protocol (FTP), Telnet ([Telnet](http://searchnetworking.techtarget.com/definition/Telnet)) which lets you logon to remote computers, and the Simple Mail Transfer Protocol ([SMTP](http://searchexchange.techtarget.com/definition/SMTP)). These and other protocols are often packaged together with TCP/IP as a "suite."
5. Personal computer users with an analog phone [modem](http://searchmobilecomputing.techtarget.com/definition/modem) connection to the Internet usually get to the Internet through the Serial Line Internet Protocol (SLIP) or the Point-to-Point Protocol ([PPP](http://searchnetworking.techtarget.com/definition/PPP)). These protocols encapsulate the IP packets so that they can be sent over the dial-up phone connection to an access provider's modem.
6. Protocols related to TCP/IP include the User Datagram Protocol ([UDP](http://searchsoa.techtarget.com/definition/UDP)), which is used instead of [TCP](http://searchnetworking.techtarget.com/definition/TCP) for special purposes. Other protocols are used by network host computers for exchanging [router](http://searchnetworking.techtarget.com/definition/router) information. These include the Internet Control Message Protocol ([ICMP](http://searchnetworking.techtarget.com/definition/ICMP)), the Interior Gateway Protocol ([IGP](http://searchsecurity.techtarget.com/definition/IGP)), the Exterior Gateway Protocol (EGP), and the Border Gateway Protocol ([BGP](http://searchtelecom.techtarget.com/definition/BGP)).