CST8221 – Java Application Programming

Unit 10 - Introduction to Java Networking

The Java networking is based on client/server model, TCP/IP and UDP protocols. Most of the network related classes are organized in ***java.net*** package.

Java Port Representation

Ports are represented in Java as integer numbers in the range 0 – 65535. Each port can be allocated to a particular service. Port numbers between 1 and 1025 are reserved for well-known services.

Java IP Addresses

In Java, IP addresses are represented by the *java.net.InetA*ddress class. It contains two fields: hostName (a String) and address (an integer number). The host name field contains the name of the host (algonqincollege.com). The address contains the 32-bit (or 16 bytes with IPv6) IP address. The class does not have public constructors but provides three static methods that return initialized InetAddress objects. They are:

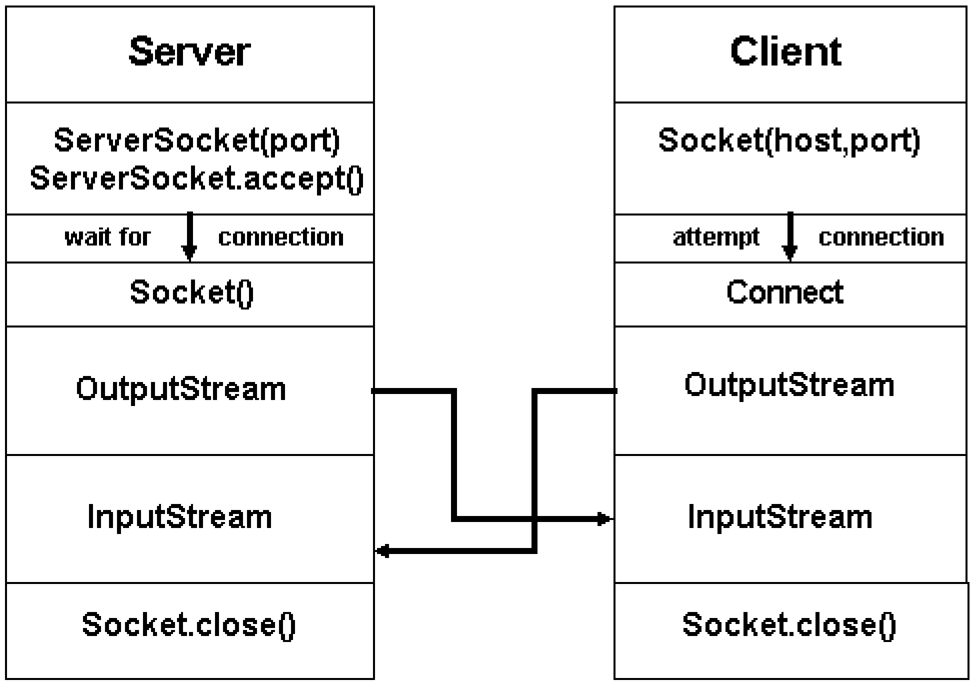
getByName(String hostname), getAllByName(String hostname), getLocalHost()

Representing URLs in Java

The java.net.URL class is an abstraction of a Uniform Resource Locator. In Java URL is an object with fields that include the protocol, hostname, port, filename, and document section. Unlike the InetAddress class, you can construct instances of java.net.URL.

Java TCP Implementation

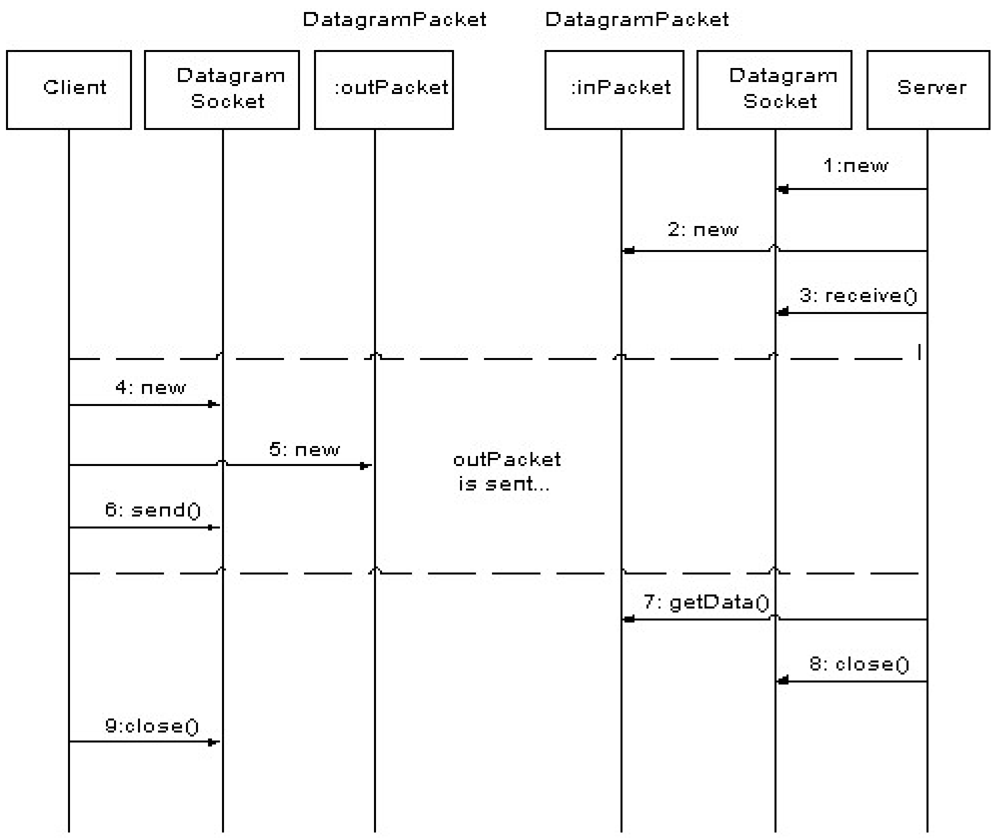
Java TCP implementation is based on sockets and streams. In the Java language TCP socket connections are implemented with classes organized in ***java.net*** package. Below is a diagram of what occur on the server side and the client side.



Java UDP Implementation

To understand the difference between TCP and UDP you should try to compare the phone system and the post office. TCP is like the phone system. It is connection-oriented. UDP, by contrast, is like the postal system. You send packets of mail to an address. Most of the letters will arrive, but some may be lost on the way. There is no guarantee that the letters will arrive in the order you sent them.
Even UDP is “unreliable”, it has its applications. The Java implementation of UDP is concentrated in two classes: DatagramPacket and DtagramSocket. To send data, you put the data in a DatagramPacket, using a DatagramSocket; to receive data, you receive a DatagramPacket object from a DatagramSocket, and then read the contents of the packet. The UDP sockets are very simple: In UDP, everything about a datagram, including the address to which it is directed, is included in the packet itself; the socket only need to know the local port on which to listen or send. UDP does not have a server socket. You use the same kind of socket to send data and to listen for incoming connection. Contrary to TCP, UDP does not allow you to treat a network connection as a stream. Finally, the UDP DatagramSocket is not dedicated to a single connection. The UDP communication process is illustrated in the diagram below.

UDP Communication Process



Description

4. Client creates socket
5. Client creates an outgoing packet
   p = new DatagramPacket()
6. Client sends datagram
   send(p)

9. Client closes socket

1. Server creates socket
   s = new DatagramSocket()
2. Server creates incoming packet
   p = new DatagramPacket()
3. Server waits for incoming datagrams
   receive(p)
8. Server closes socket