Java Socket Programming

Java Socket programming can be connection-oriented or connection-less.

Socket and ServerSocket classes are used for connection-oriented socket programming and DatagramSocket and DatagramPacket classes are used for connection-less socket programming.

The client in socket programming must know two information:

1. IP Address of Server, and
2. Port number.

**Creating Server:**

**T**o create the server application, we need to create the instance of ServerSocket class.

1. ServerSocket ss=**new** ServerSocket(6666);
2. Socket s=ss.accept();//establishes connection and waits for the client

**Creating Client:**

To create the client application, we need to create the instance of Socket class.

1. Socket s=**new** Socket("localhost",6666);

*File: MyServer.java*

1. ServerSocket ss=**new** ServerSocket(6666);
2. Socket s=ss.accept();//establishes connection
3. DataInputStream dis=**new** DataInputStream(s.getInputStream());
4. String  str=(String)dis.readUTF();
5. System.out.println("message= "+str);
6. ss.close();

*File: MyClient.java*

1. Socket s=**new** Socket("localhost",6666);
2. DataOutputStream dout=**new** DataOutputStream(s.getOutputStream());
3. dout.writeUTF("Hello Server");
4. dout.flush();
5. dout.close();

# Java URL

A URL contains many information:

1. **Protocol:** In this case, http is the protocol.
2. **Server name or IP Address:** In this case, www.javatpoint.com is the server name.
3. **Port Number:** It is an optional attribute. If we write http//ww.javatpoint.com:80/sonoojaiswal/ , 80 is the port number. If port number is not mentioned in the URL, it returns -1.
4. **File Name or directory name:** In this case, index.jsp is the file name.

Java InetAddress class

**Java InetAddress** class represents an IP address. The java.net.InetAddress class provides methods to get the IP of any host name *for example* www.javatpoint.com, www.google.com, www.facebook.com, etc.

1. InetAddress ip=**InetAddress.getByName(**"www.javatpoint.com");
2. System.out.println("Host Name: "+ip.getHostName());
3. System.out.println("IP Address: "+ip.getHostAddress());