Module 9 - Java
Network Programming

Advanced Java Certification Training

Akram M'Tir

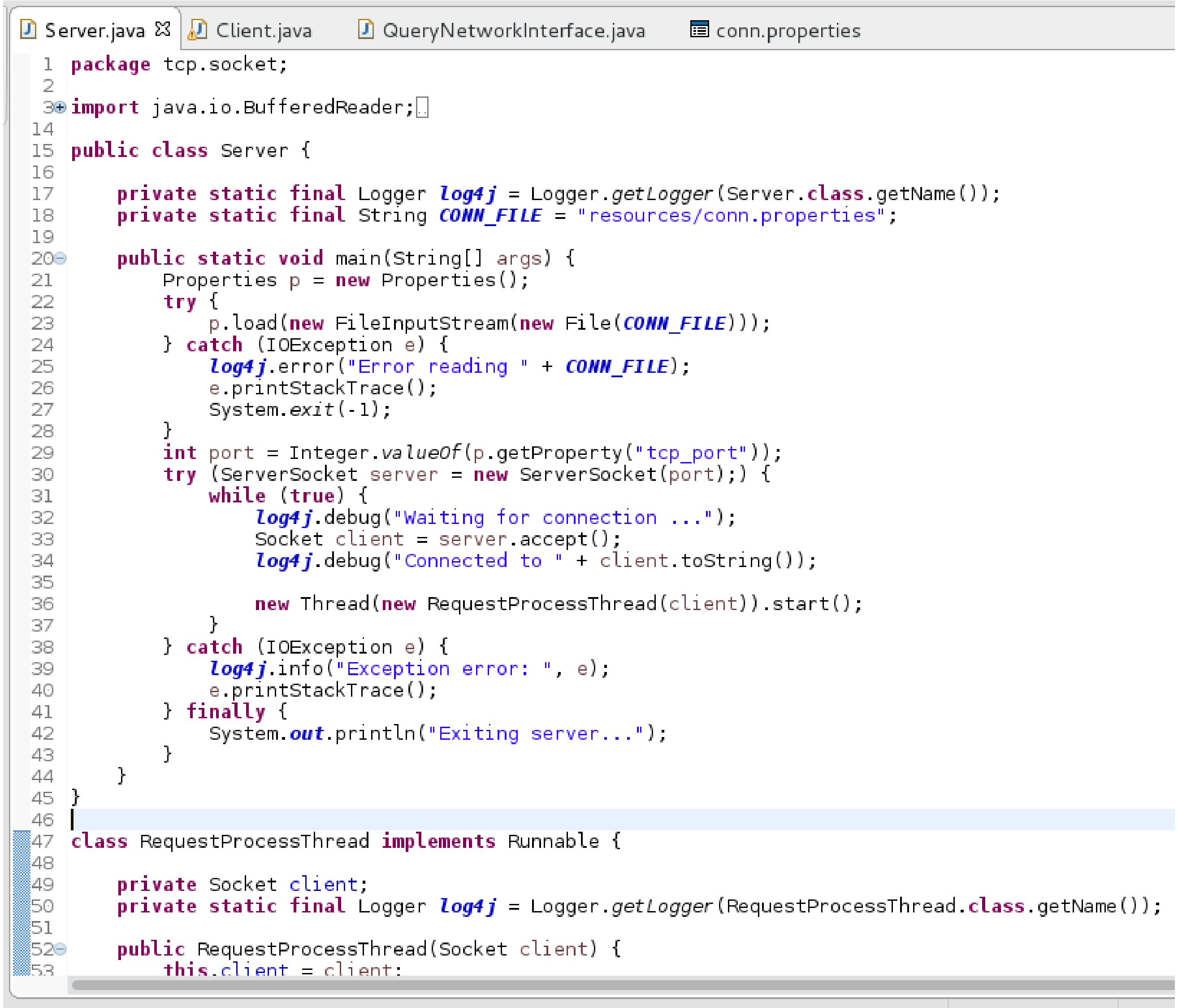
1. Write a programme to support multiple socket client connections to a server.

a. For simplicity, make it an echo server like what we saw in the code walkthrough module9.tcp.sockets.Server.

b. Use properties file to load the host and port details.

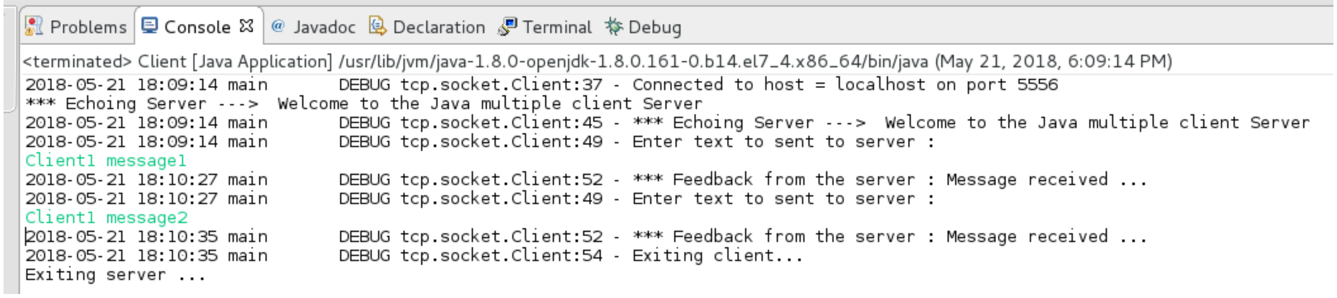
c. Handle appropriate logging with Log4j

d. Print a welcome message telling the connection number when a client connects to the server.



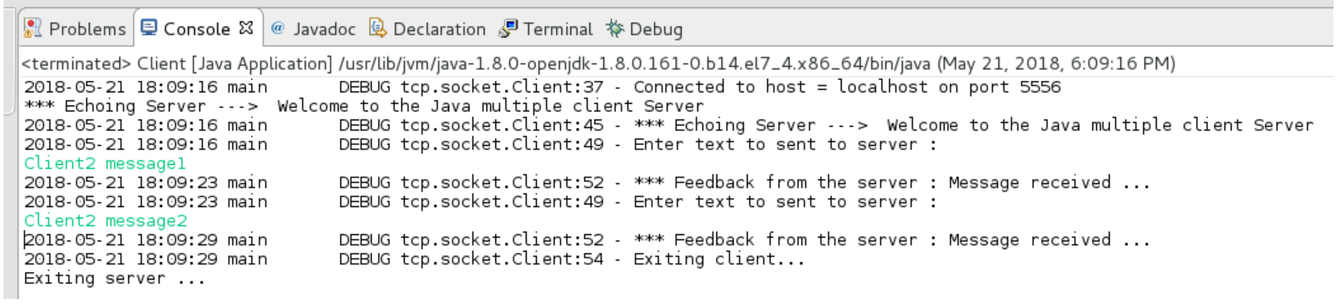
```java
package tcp.socket;

import java.io.BufferedReader;

public class Server {

    private static final Logger log4j = Logger.getLogger(Server.class.getName());
    private static final String CONN_FILE = "resources/conn.properties";

    public static void main(String[] args) {
        Properties p = new Properties();
        try {
            p.load(new FileInputStream(new File(CONN_FILE)));
        } catch (IOException e) {
            log4j.error("Error reading " + CONN_FILE);
            e.printStackTrace();
            System.exit(-1);
        }
        int port = Integer.valueOf(p.getProperty("tcp_port"));
        try (ServerSocket server = new ServerSocket(port);) {
            while (true) {
                log4j.debug("Waiting for connection ...");
                Socket client = server.accept();
                log4j.debug("Connected to " + client.toString());

                new Thread(new RequestProcessThread(client)).start();
            }
        } catch (IOException e) {
            log4j.info("Exception error: ", e);
            e.printStackTrace();
        } finally {
            System.out.println("Exiting server...");
        }
    }
}

class RequestProcessThread implements Runnable {

    private Socket client;
    private static final Logger log4j = Logger.getLogger(RequestProcessThread.class.getName());

    public RequestProcessThread(Socket client) {
        this.client = client;
```

Client1



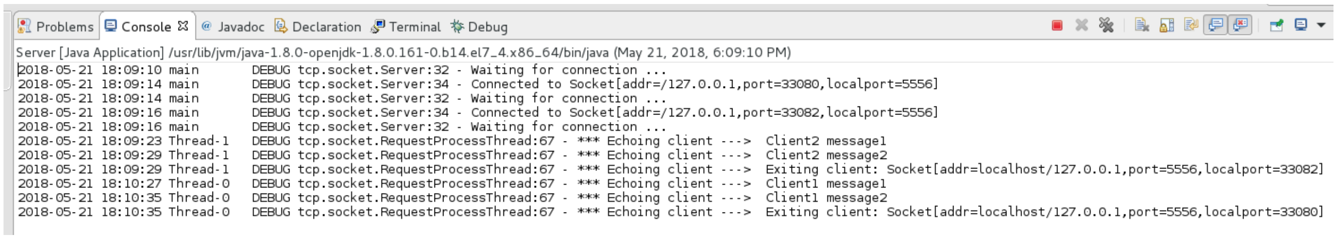
```
<terminated> Client [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 21, 2018, 6:09:14 PM)
2018-05-21 18:09:14 main       DEBUG tcp.socket.Client:37 - Connected to host = localhost on port 5556
*** Echoing Server --->  Welcome to the Java multiple client Server
2018-05-21 18:09:14 main       DEBUG tcp.socket.Client:45 - *** Echoing Server --->  Welcome to the Java multiple client Server
2018-05-21 18:09:14 main       DEBUG tcp.socket.Client:49 - Enter text to sent to server :
Client1 message1
2018-05-21 18:10:27 main       DEBUG tcp.socket.Client:52 - *** Feedback from the server : Message received ...
2018-05-21 18:10:27 main       DEBUG tcp.socket.Client:49 - Enter text to sent to server :
Client1 message2
2018-05-21 18:10:35 main       DEBUG tcp.socket.Client:52 - *** Feedback from the server : Message received ...
2018-05-21 18:10:35 main       DEBUG tcp.socket.Client:54 - Exiting client...
Exiting server ...
```

Client2



```
<terminated> Client [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 21, 2018, 6:09:16 PM)
2018-05-21 18:09:16 main       DEBUG tcp.socket.Client:37 - Connected to host = localhost on port 5556
*** Echoing Server --->  Welcome to the Java multiple client Server
2018-05-21 18:09:16 main       DEBUG tcp.socket.Client:45 - *** Echoing Server --->  Welcome to the Java multiple client Server
2018-05-21 18:09:16 main       DEBUG tcp.socket.Client:49 - Enter text to sent to server :
Client2 message1
2018-05-21 18:09:23 main       DEBUG tcp.socket.Client:52 - *** Feedback from the server : Message received ...
2018-05-21 18:09:23 main       DEBUG tcp.socket.Client:49 - Enter text to sent to server :
Client2 message2
2018-05-21 18:09:29 main       DEBUG tcp.socket.Client:52 - *** Feedback from the server : Message received ...
2018-05-21 18:09:29 main       DEBUG tcp.socket.Client:54 - Exiting client...
Exiting server ...
```

Server Side

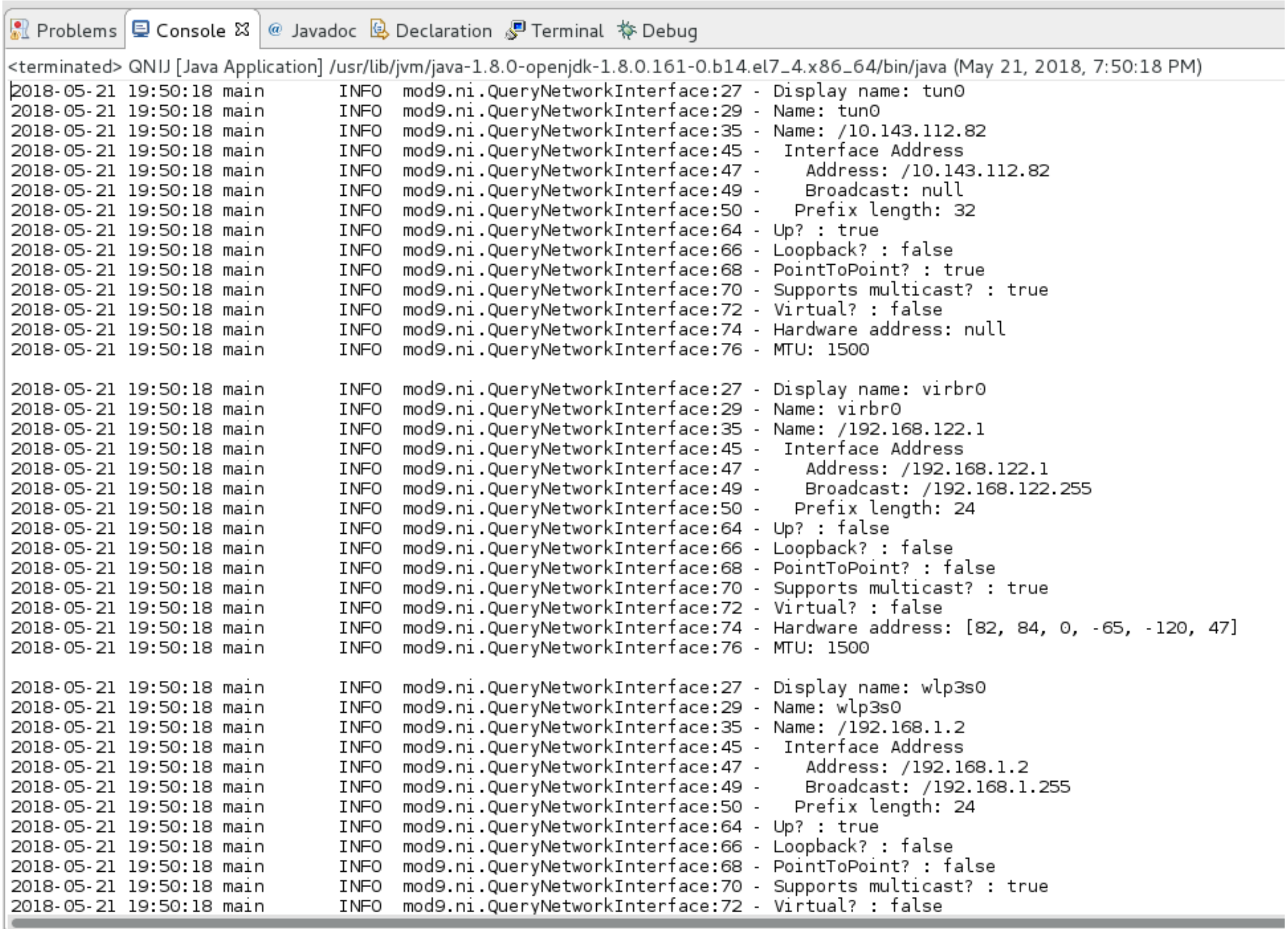


```
Server [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 21, 2018, 6:09:10 PM)
2018-05-21 18:09:10 main       DEBUG tcp.socket.Server:32 - Waiting for connection ...
2018-05-21 18:09:14 main       DEBUG tcp.socket.Server:34 - Connected to Socket[addr=/127.0.0.1,port=33080,localport=5556]
2018-05-21 18:09:14 main       DEBUG tcp.socket.Server:32 - Waiting for connection ...
2018-05-21 18:09:16 main       DEBUG tcp.socket.Server:34 - Connected to Socket[addr=/127.0.0.1,port=33082,localport=5556]
2018-05-21 18:09:16 main       DEBUG tcp.socket.Server:32 - Waiting for connection ...
2018-05-21 18:09:23 Thread-1   DEBUG tcp.socket.RequestProcessThread:67 - *** Echoing client --->  Client2 message1
2018-05-21 18:09:29 Thread-1   DEBUG tcp.socket.RequestProcessThread:67 - *** Echoing client --->  Client2 message2
2018-05-21 18:09:29 Thread-1   DEBUG tcp.socket.RequestProcessThread:67 - *** Echoing client --->  Exiting client: Socket[addr=localhost/127.0.0.1,port=5556,localport=33082]
2018-05-21 18:10:27 Thread-0   DEBUG tcp.socket.RequestProcessThread:67 - *** Echoing client --->  Client1 message1
2018-05-21 18:10:35 Thread-0   DEBUG tcp.socket.RequestProcessThread:67 - *** Echoing client --->  Client1 message2
2018-05-21 18:10:35 Thread-0   DEBUG tcp.socket.RequestProcessThread:67 - *** Echoing client --->  Exiting client: Socket[addr=localhost/127.0.0.1,port=5556,localport=33080]
```

2. Write a programme to list the network interfaces on your computer. The programme should list the following:

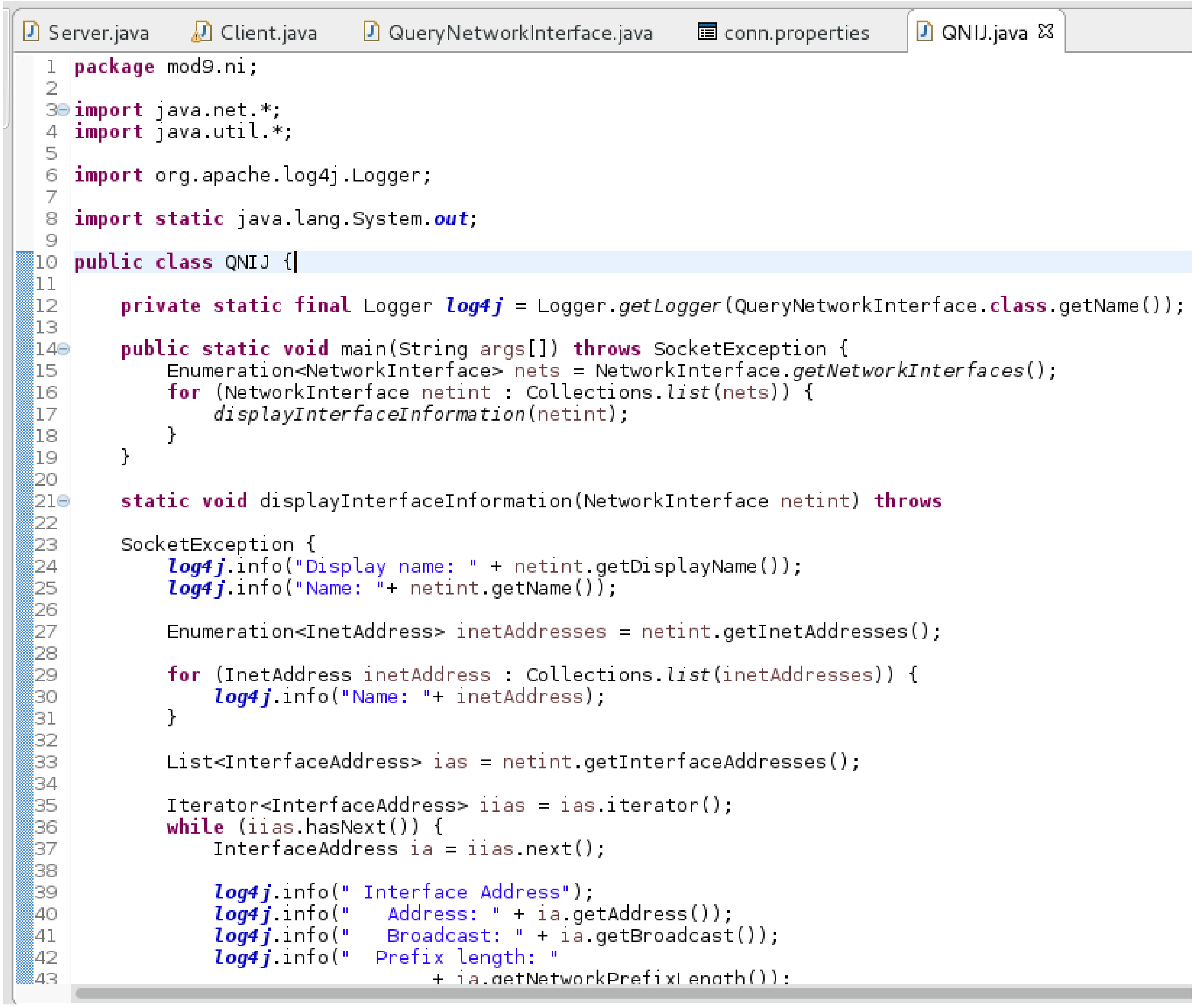
a. Display name.
b. Is this interface running?
c. Does this interface supports multicast?

[Hint: Check the methods of java.net.NetworkInterface]



<terminated> QNIJ [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x86_64/bin/java (May 21, 2018, 7:50:18 PM)

```
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:27 - Display name: tun0
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:29 - Name: tun0
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:35 - Name: /10.143.112.82
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:45 -  Interface Address
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:47 -    Address: /10.143.112.82
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:49 -    Broadcast: null
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:50 -   Prefix length: 32
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:64 - Up? : true
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:66 - Loopback? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:68 - PointToPoint? : true
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:70 - Supports multicast? : true
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:72 - Virtual? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:74 - Hardware address: null
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:76 - MTU: 1500

2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:27 - Display name: virbr0
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:29 - Name: virbr0
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:35 - Name: /192.168.122.1
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:45 -  Interface Address
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:47 -    Address: /192.168.122.1
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:49 -    Broadcast: /192.168.122.255
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:50 -   Prefix length: 24
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:64 - Up? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:66 - Loopback? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:68 - PointToPoint? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:70 - Supports multicast? : true
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:72 - Virtual? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:74 - Hardware address: [82, 84, 0, -65, -120, 47]
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:76 - MTU: 1500

2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:27 - Display name: wlp3s0
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:29 - Name: wlp3s0
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:35 - Name: /192.168.1.2
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:45 -  Interface Address
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:47 -    Address: /192.168.1.2
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:49 -    Broadcast: /192.168.1.255
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:50 -   Prefix length: 24
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:64 - Up? : true
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:66 - Loopback? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:68 - PointToPoint? : false
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:70 - Supports multicast? : true
2018-05-21 19:50:18 main        INFO  mod9.ni.QueryNetworkInterface:72 - Virtual? : false
```



```java
package mod9.ni;

import java.net.*;
import java.util.*;

import org.apache.log4j.Logger;

import static java.lang.System.out;

public class QNIJ {

    private static final Logger log4j = Logger.getLogger(QueryNetworkInterface.class.getName());

    public static void main(String args[]) throws SocketException {
        Enumeration<NetworkInterface> nets = NetworkInterface.getNetworkInterfaces();
        for (NetworkInterface netint : Collections.list(nets)) {
            displayInterfaceInformation(netint);
        }
    }

    static void displayInterfaceInformation(NetworkInterface netint) throws

    SocketException {
        log4j.info("Display name: " + netint.getDisplayName());
        log4j.info("Name: "+ netint.getName());

        Enumeration<InetAddress> inetAddresses = netint.getInetAddresses();

        for (InetAddress inetAddress : Collections.list(inetAddresses)) {
            log4j.info("Name: "+ inetAddress);
        }

        List<InterfaceAddress> ias = netint.getInterfaceAddresses();

        Iterator<InterfaceAddress> iias = ias.iterator();
        while (iias.hasNext()) {
            InterfaceAddress ia = iias.next();

            log4j.info(" Interface Address");
            log4j.info("    Address: " + ia.getAddress());
            log4j.info("    Broadcast: " + ia.getBroadcast());
            log4j.info("   Prefix length: "
                    + ia.getNetworkPrefixLength());
```

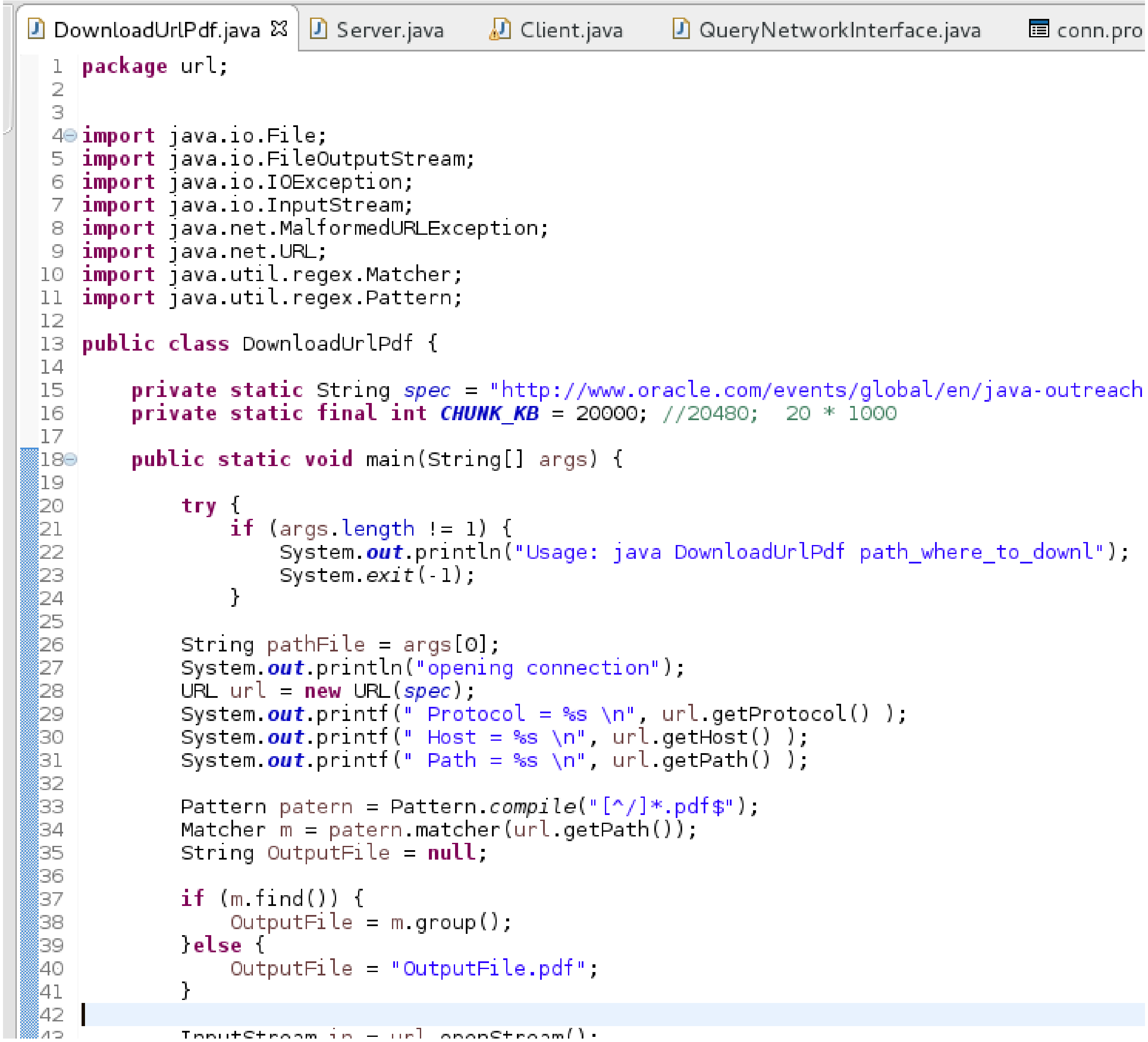
3. Write a programme to download the contents of http://www.oracle.com/events/global/en/java-outreach/resources/java-a-beginners-guide-1720064.pdf to a file with the same name.

a. Extract the filename from the URL, don't hardcode.

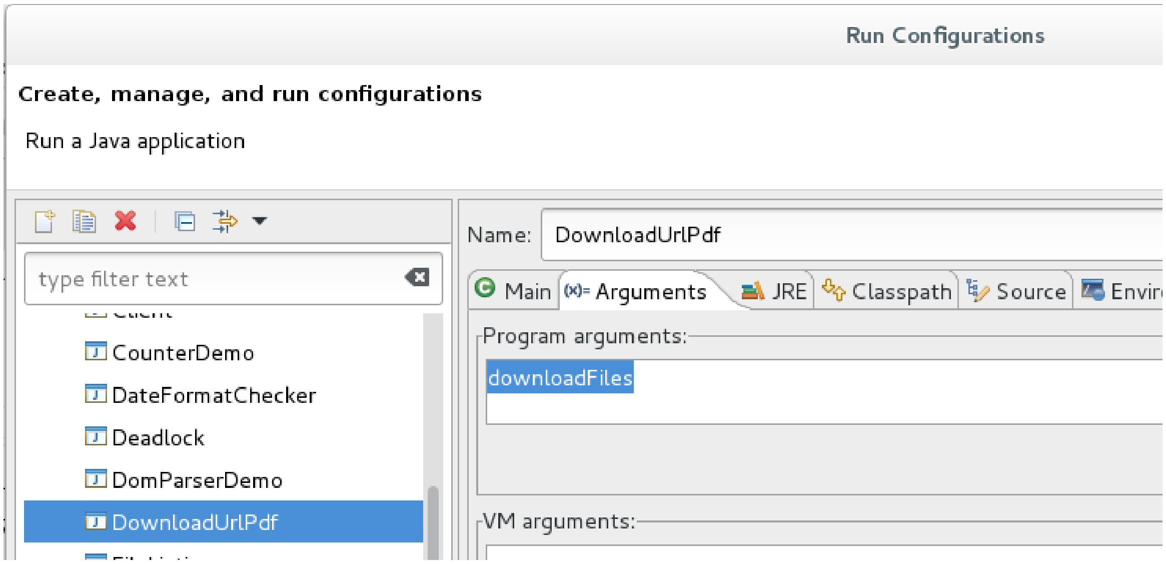
b. The download folder will be supplied through the command line.

c. Show the progress as % downloaded while the downloading is happening.

d. Download in chunks of 20KB.



```java
package url;


import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class DownloadUrlPdf {

    private static String spec = "http://www.oracle.com/events/global/en/java-outreach
    private static final int CHUNK_KB = 20000; //20480;  20 * 1000

    public static void main(String[] args) {

        try {
            if (args.length != 1) {
                System.out.println("Usage: java DownloadUrlPdf path_where_to_downl");
                System.exit(-1);
            }

            String pathFile = args[0];
            System.out.println("opening connection");
            URL url = new URL(spec);
            System.out.printf(" Protocol = %s \n", url.getProtocol() );
            System.out.printf(" Host = %s \n", url.getHost() );
            System.out.printf(" Path = %s \n", url.getPath() );

            Pattern patern = Pattern.compile("[^/]*.pdf$");
            Matcher m = patern.matcher(url.getPath());
            String OutputFile = null;

            if (m.find()) {
                OutputFile = m.group();
            }else {
                OutputFile = "OutputFile.pdf";
            }

            InputStream in = url.openStream();
```



Run Configurations

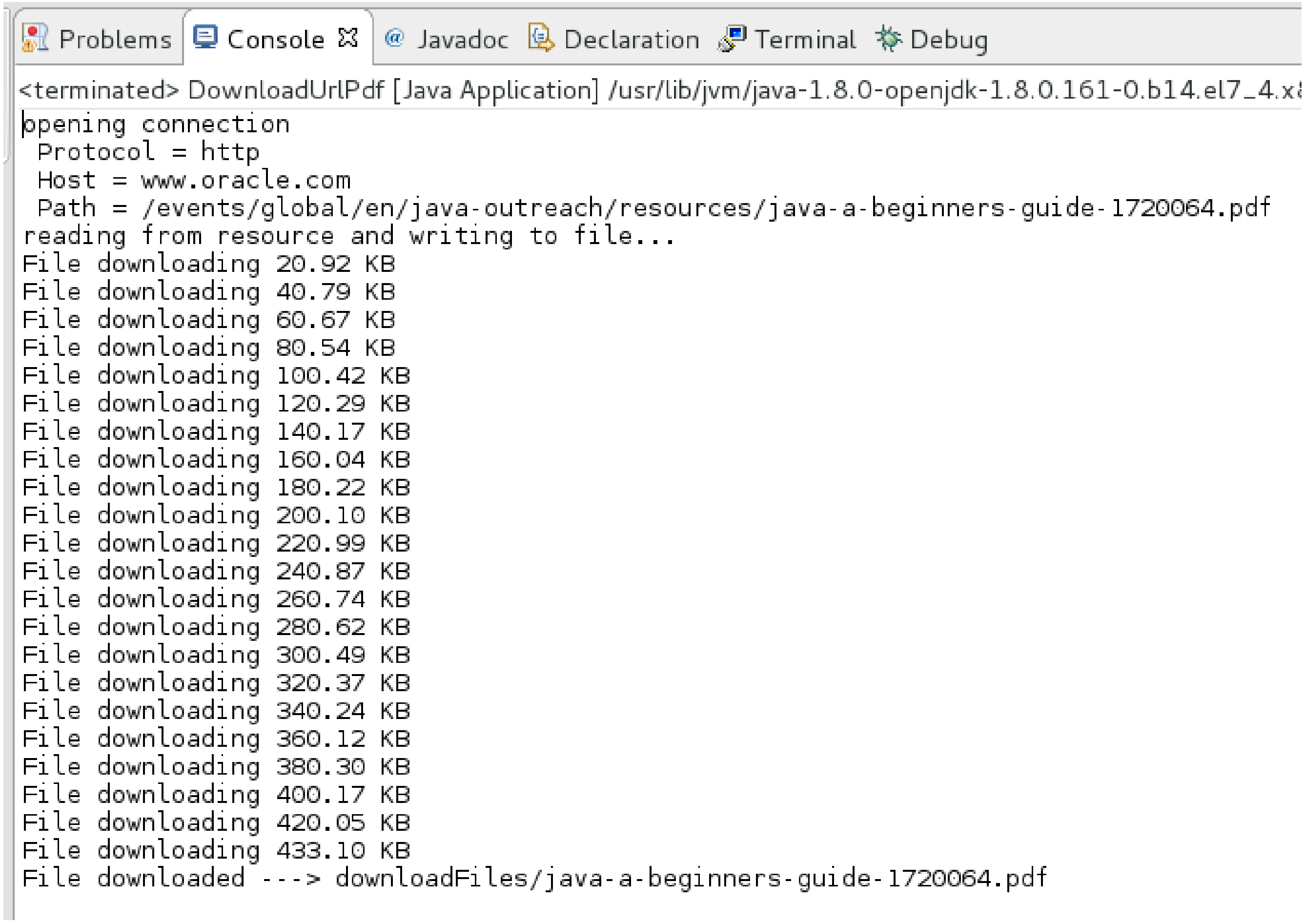
Create, manage, and run configurations

Run a Java application

Name: DownloadUrlPdf

Program arguments: downloadFiles

VM arguments:



```
<terminated> DownloadUrlPdf [Java Application] /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.161-0.b14.el7_4.x
opening connection
 Protocol = http
 Host = www.oracle.com
 Path = /events/global/en/java-outreach/resources/java-a-beginners-guide-1720064.pdf
reading from resource and writing to file...
File downloading 20.92 KB
File downloading 40.79 KB
File downloading 60.67 KB
File downloading 80.54 KB
File downloading 100.42 KB
File downloading 120.29 KB
File downloading 140.17 KB
File downloading 160.04 KB
File downloading 180.22 KB
File downloading 200.10 KB
File downloading 220.99 KB
File downloading 240.87 KB
File downloading 260.74 KB
File downloading 280.62 KB
File downloading 300.49 KB
File downloading 320.37 KB
File downloading 340.24 KB
File downloading 360.12 KB
File downloading 380.30 KB
File downloading 400.17 KB
File downloading 420.05 KB
File downloading 433.10 KB
File downloaded ---> downloadFiles/java-a-beginners-guide-1720064.pdf
```