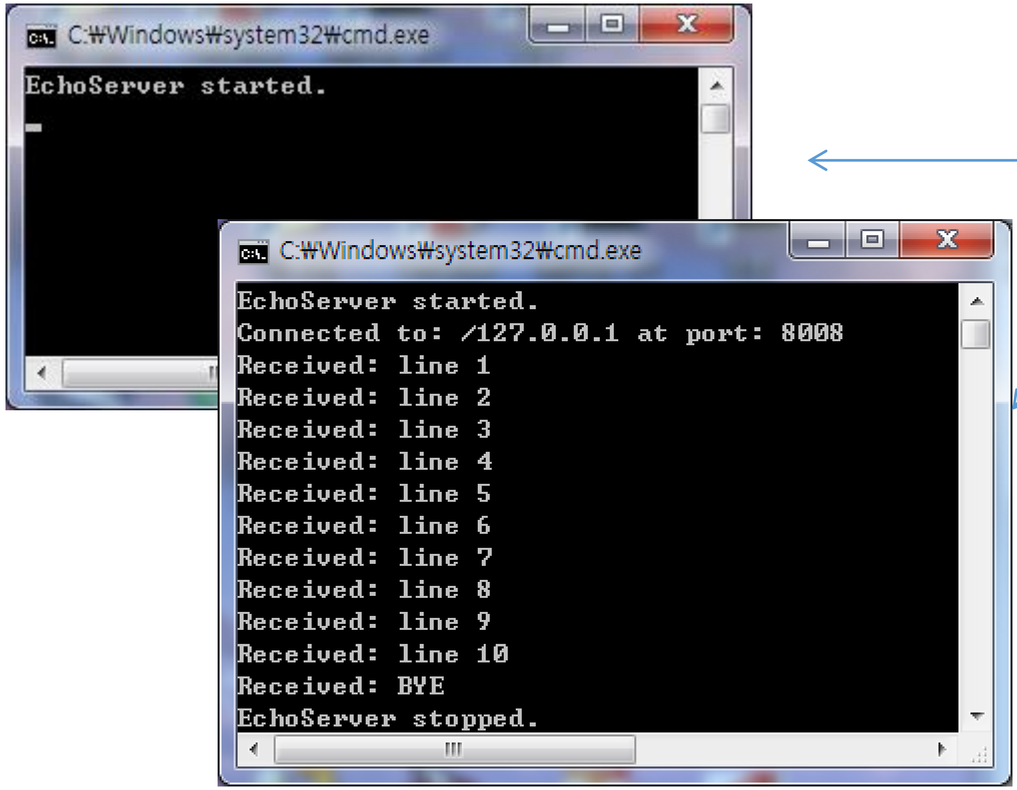
Socket Communication Test

OODP 2022

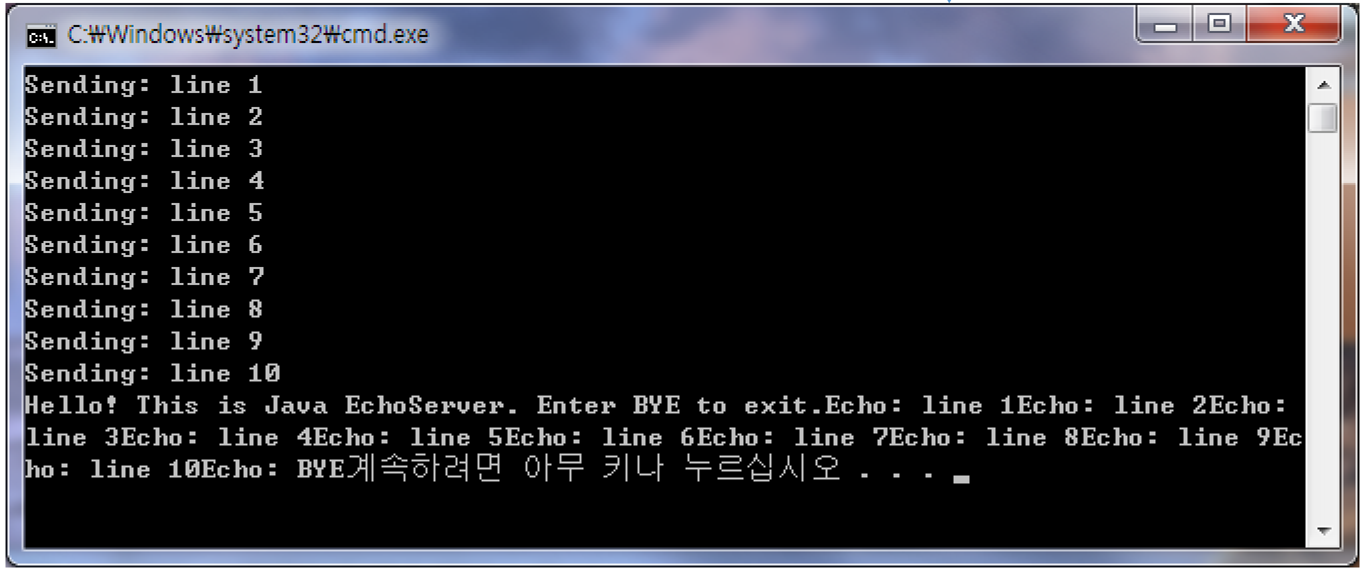
Remote proxy

They are responsible for representing the object located remotely. Talking to the real object might involve marshalling and unmarshalling of data and talking to the remote object. All that logic is encapsulated in these proxies and the client application need not worry about them.



```
EchoServer started.
_
```

```
EchoServer started.
Connected to: /127.0.0.1 at port: 8008
Received: line 1
Received: line 2
Received: line 3
Received: line 4
Received: line 5
Received: line 6
Received: line 7
Received: line 8
Received: line 9
Received: line 10
Received: BYE
EchoServer stopped.
```



```
Sending: line 1
Sending: line 2
Sending: line 3
Sending: line 4
Sending: line 5
Sending: line 6
Sending: line 7
Sending: line 8
Sending: line 9
Sending: line 10
Hello! This is Java EchoServer. Enter BYE to exit.Echo: line 1Echo: line 2Echo:
line 3Echo: line 4Echo: line 5Echo: line 6Echo: line 7Echo: line 8Echo: line 9Ec
ho: line 10Echo: BYE계속하려면 아무 키나 누르십시오 . . . _
```

```
import java.io.*;
import java.net.*;
class EchoServer {
  public static void main(String[] args) {
    System.out.println("EchoServer started.");
    try {
      ServerSocket s = new ServerSocket(8008);
      Socket incoming = s.accept();

     System.out.println("Connected to: " + incoming.getInetAddress() +
                          " at port: " + incoming.getLocalPort());
          BufferedReader in
          = new BufferedReader(new InputStreamReader(incoming.getInputStream()));
      PrintWriter out
          = new PrintWriter(new OutputStreamWriter(incoming.getOutputStream()));
          out.println("Hello! This is Java EchoServer. Enter BYE to exit.");
      out.flush();
for (;;) {
        String str = in.readLine();
        if (str == null) {
          break;
        } else {
          out.println("Echo: " + str); //to client
          out.flush();
          System.out.println("Received: " + str);
          if (str.trim().equals("BYE"))
            break;
        }
      }
      incoming.close();
    } catch (Exception e) {
      System.out.println("Error: " + e);
    }
    System.out.println("EchoServer stopped.");
 }  }
```

```
public class EchoClient {
  public static void main(String[] args) {
    try {
      String host;
      if (args.length>0) {
        host=args[0];
      } else {
        host="localhost";
      }
      Socket socket=new Socket(host, 8008);
      BufferedReader in
          = new BufferedReader(new
                InputStreamReader(socket.getInputStream()));
      PrintWriter out
          = new PrintWriter(new
OutputStreamWriter(socket.getOutputStream()));
      // send data to the server
for (int i=1; i<=10; i++) {
        System.out.println("Sending: line "+i);
        out.println("line "+i);
        out.flush();
      }
      out.println("BYE");
      out.flush();
      // receive data from the server
      while (true) {
        String str=in.readLine();
        if (str==null) {
          break;
        } else {
          System.out.print(str);
        }
      }
    } catch (Exception e) {
      e.printStackTrace();       }   }}
```