

Membuat Aplikasi Chatting Dengan Java

M Zainal Arifin

agendogget@yahoo.com

Lisensi Dokumen:

Copyright © 2004 IlmuKomputer.Com

*Seluruh dokumen di **IlmuKomputer.Com** dapat digunakan, dimodifikasi dan disebarkan secara bebas untuk tujuan bukan komersial (nonprofit), dengan syarat tidak menghapus atau merubah atribut penulis dan pernyataan copyright yang disertakan dalam setiap dokumen. Tidak diperbolehkan melakukan penulisan ulang, kecuali mendapatkan ijin terlebih dahulu dari **IlmuKomputer.Com**.*

Sudah tidak asing lagi bagi kita tentang chatting di internet. Mengobrol lewat internet itu memang asyik dan menyenangkan sehingga tak jarang seseorang yang pergi ke warung internet hanya untuk chatting saja sampai berjam-jam. Dengan chatting lewat internet anda bisa mendapatkan banyak teman dari penjuru dunia. Hal keamanan merupakan faktor penting bagi kita dalam berbagi informasi. Misalnya, jika bos kita ingin mendengar laporan kerja karyawan, maka jika kita memberitahu dengan SMS lewat ICQ, email dan sebagainya serasa tidak efektif bila kita menginginkan komunikasi dua arah yang saling bertemu dan menjamin keamanan tinggi sehingga menciptakan suasana yang akrab dan nyaman.

Dalam hal ini anda bisa memakai program chatting mIRC32, ICQ, dan sebagainya yang sudah disediakan oleh warung internet. Apakah orang lain yang tidak diinginkan bisa menyusup dan mengambil data kita?, tentu saja bisa. Sebab kebanyakan program chatting justru memilih dari segi kegunaan yang umum. Nah, sekarang kita akan mencoba membuat program chatting yang sangat berguna dan meningkatkan keamanan data sehingga mengurangi orang yang tidak diinginkan masuk ke dalam forum kita dan 'mencuri' data kita.

Hal yang menarik disini adalah kita akan membuat sendiri program chatting ini. Dalam hal pemrograman ini saya menggunakan bahasa java yang sudah kita kenal selama beberapa tahun ini. Java adalah bahasa yang handal, mudah dipelajari bagi pengguna C++, rumus umum dasar java menyerupai C++ dan ada juga yang tidak mirip, tetapi java lebih mengkhususkan diri untuk *full* obyek.

Begini ceritanya, masing-masing server dan client akan menjalankan dua program. Lho kok dua program?, karena saya akan menyajikan program yang mengirim data (tulisan) dari server ke klien dan dari klien ke server. Jadi disini anda akan dihadapkan pada bagaimana menyatukan dua program pada masing-masing computer sehingga masing-masing computer hanya menjalankan satu program saja. Lebih efektif bukan?, untuk itulah anda bisa memilih dan

memodifikasi port, kemananan dan sebagainya secara acak dari kedua program berikut yang menurut anda memiliki kehandalan.

Listing programnya sebagai berikut :

1. Server mengirim data ke Client

```
// program server: simpan dalam ServerKu1.java
// kompilasi menjadi ServerKu1.class

import java.net.*;
import java.io.*;

public class ServerKu1 {
    public static void main(String[] args) {
        try {
            ServerSocket ss = new ServerSocket(2002);
            System.out.println("Menunggu Klien...");
            Socket s = ss.accept();
            System.out.println("Klien telah masuk");
            System.out.println("Server pada port = "+
                s.getLocalAddress().getHostAddress() + " : " + s.getLocalPort());
            System.out.println("Client pada port = "+
                s.getInetAddress().getHostAddress() + " : " + s.getPort());

            PrintWriter pw = new PrintWriter(s.getOutputStream(), true);
            BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
            String greeting = "";
            while(!(greeting=br.readLine()).equals("")) { pw.println(greeting); }

            pw.close(); s.close(); ss.close();
        } catch (Exception e) { e.printStackTrace(); }
    }
}
```

```
// program klien : Simpan dalam ClientKu1.java  
// kompilasi menjadi ClientKu1.class
```

```
import java.net.*;  
import java.io.*;  
  
public class ClientKu1 {  
    public static void main(String[] args) {  
        try {  
            Socket s = new Socket("127.0.0.1",2002);  
            System.out.println("Komunikasi terhubung");  
            System.out.println("Client pada port = "+  
                s.getLocalAddress().getHostAddress() + " : "+ s.getLocalPort());  
            System.out.println("Server pada port = "+  
                s.getInetAddress().getHostAddress() + " : "+ s.getPort());  
  
            BufferedReader br = new BufferedReader( new InputStreamReader(s.getInputStream()));  
            System.out.println("Menunggu pesan...");  
  
            String greeting = "";  
            while ((greeting=br.readLine())!=null) { System.out.println(greeting); }  
            System.out.println("Selesai");  
            br.close();  
            s.close();  
        }  
        catch (Exception e) { e.printStackTrace(); }  
    }  
}
```

2. Client mengirim data ke Server

```
// program server : Simpan dalam ServerKu2.java
// kompile menjadi ServerKu2.class

import java.net.*;
import java.io.*;

public class ServerKu2 {

    public static void main(String args[]) {

        int port;
        ServerSocket server_socket;
        BufferedReader input;

        try {
            port = Integer.parseInt(args[0]);
        }
        catch (Exception e) {
            System.out.println("port = 2000 (Standarnya)");
            port = 2000;
        }

        try {

            server_socket = new ServerSocket(port);
            System.out.println("Server menunggu klien di port " +
                               server_socket.getLocalPort());

            // server mondar-mandir disini
            while(true) {
                Socket socket = server_socket.accept();
                System.out.println("Hubungan baru diterima " +
                                   socket.getInetAddress() +
                                   ":" + socket.getPort());
                input = new BufferedReader(new InputStreamReader(socket.getInputStream()));
                // menampilkan data yang diterima
                try {
                    while(true) {
                        String message = input.readLine();
                        if (message==null) break;
                        System.out.println(message);
                    }
                }
                catch (IOException e) {
                    System.out.println(e);
                }

                // Client menutup hubungan
                try {
                    socket.close();
                    System.out.println("Client telah menutup hubungan");
                }
                catch (IOException e) {
                    System.out.println(e);
                }
            }
        }
    }
}
```



```
// program client : Simpan dalam ClientKu2.java
// kompilasi menjadi ClientKu2.class
import java.net.*;
import java.io.*;

public class ClientKu2 {

    public static void main(String[] args) {

        int port = 2002;
        String server = "localhost";
        Socket socket = null;
        String lineToBeSent;
        BufferedReader input;
        PrintWriter output;
        int ERROR = 1;

        // read arguments
        if(args.length == 2) {
            server = args[0];
            try {
                port = Integer.parseInt(args[1]);
            }
            catch (Exception e) {
                System.out.println("server port = 1500 (Standarnya)");
                port = 2002;
            }
        }

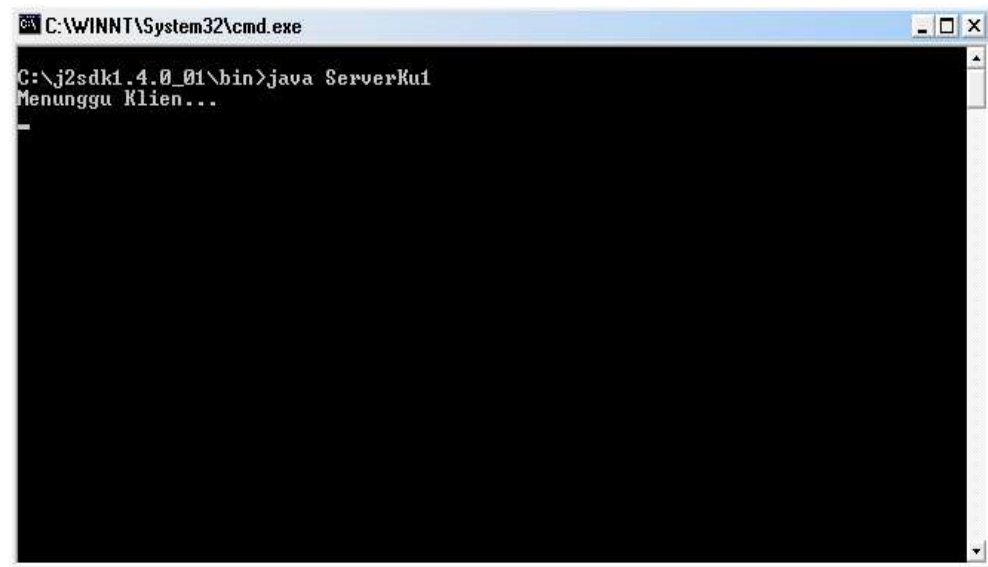
        // Koneksi ke server
        try {
            socket = new Socket(server, port);
            System.out.println("Terkoneksi ke server " +
                               socket.getInetAddress() +
                               ":" + socket.getPort());
        }
        catch (UnknownHostException e) {
            System.out.println(e);
            System.exit(ERROR);
        }
        catch (IOException e) {
            System.out.println(e);
            System.exit(ERROR);
        }

        try {
            input = new BufferedReader(new InputStreamReader(System.in));
            output = new PrintWriter(socket.getOutputStream(), true);
        }
```

Untuk mengkompile dan menjalankannya, maka ikuti langkah berikut :

- Server ke Client
 1. Kompile ServerKul.java :
C:\> javac ServerKul.java
Jalankan program :
C:\> java ServerKul

Output :

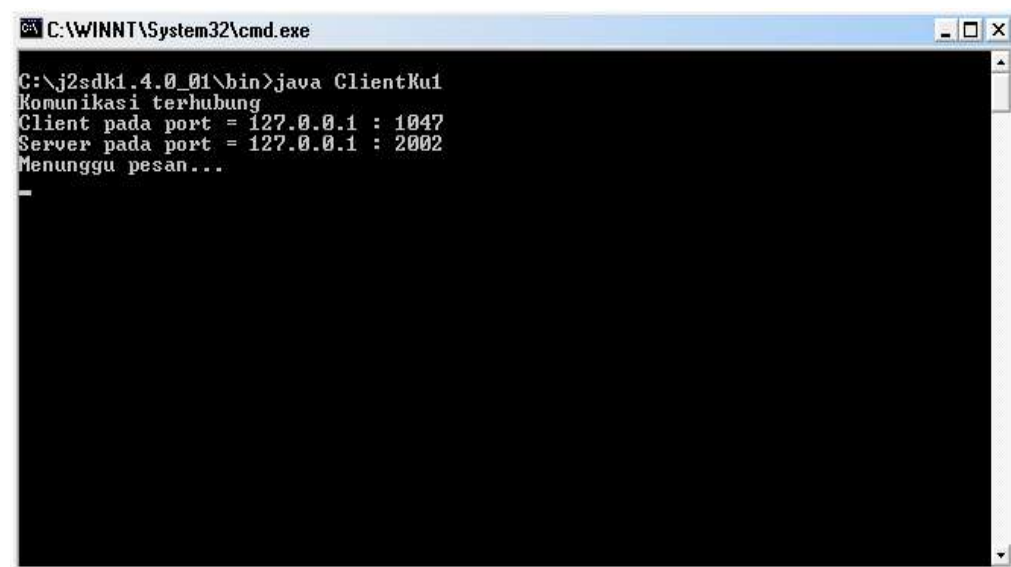


```
C:\WINNT\System32\cmd.exe

C:\j2sdk1.4.0_01\bin>java ServerKul
Menunggu Klien...
```

2. Kompile ClientKul.java :
C:\> javac ClientKul.java
Jalankan program :
C:\> java ClientKul


Output :



```
C:\WINNT\System32\cmd.exe

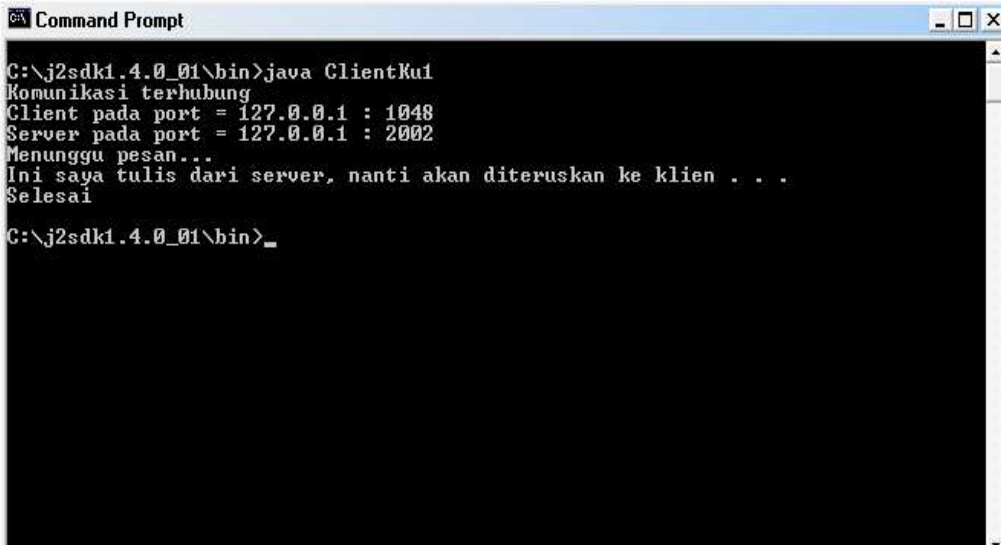
C:\j2sdk1.4.0_01\bin>java ClientKul
Komunikasi terhubung
Client pada port = 127.0.0.1 : 1047
Server pada port = 127.0.0.1 : 2002
Menunggu pesan...
```


Setelah ClientKu1.class dijalankan maka anda bisa mengetikkan kalimat di jendela Server dan akan di tampilkan di jendela klien. Untuk mengakhiri koneksi, anda bisa menekan enter dua kali atau menutup dari jendela windows.



```
Command Prompt
C:\j2sdk1.4.0_01\bin>java ServerKu1
Menunggu Klien...
Klien telah masuk
Server pada port = 127.0.0.1 : 2002
Client pada port = 127.0.0.1 : 1048
Ini saya tulis dari server, nanti akan diteruskan ke klien . . .

C:\j2sdk1.4.0_01\bin>_
```

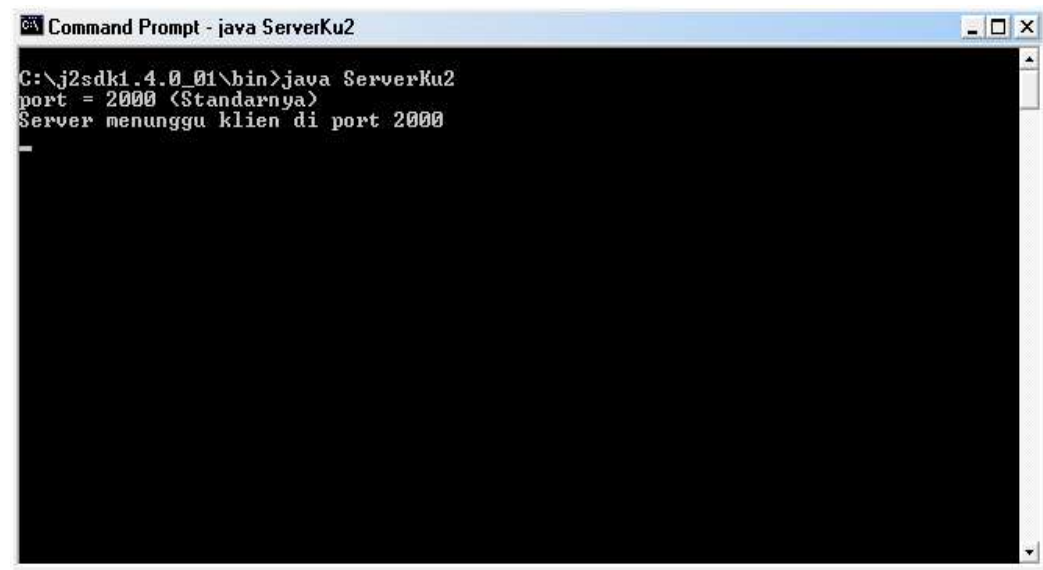


```
Command Prompt
C:\j2sdk1.4.0_01\bin>java ClientKu1
Komunikasi terhubung
Client pada port = 127.0.0.1 : 1048
Server pada port = 127.0.0.1 : 2002
Menunggu pesan...
Ini saya tulis dari server, nanti akan diteruskan ke klien . . .
Selesai

C:\j2sdk1.4.0_01\bin>_
```

- Client ke Server
 1. Kompile ServerKu2.java :
C:\> javac ServerKu2.java
Jalankan program :
C:\> java ServerKu2

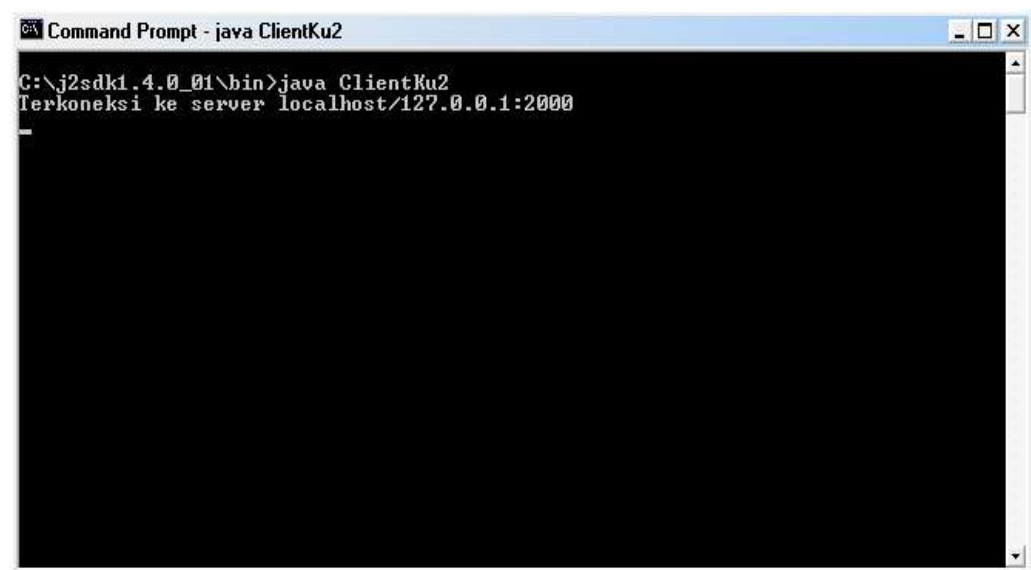
Output :



```
C:\j2sdk1.4.0_01\bin>java ServerKu2
port = 2000 (Standarnya)
Server menunggu klien di port 2000
```

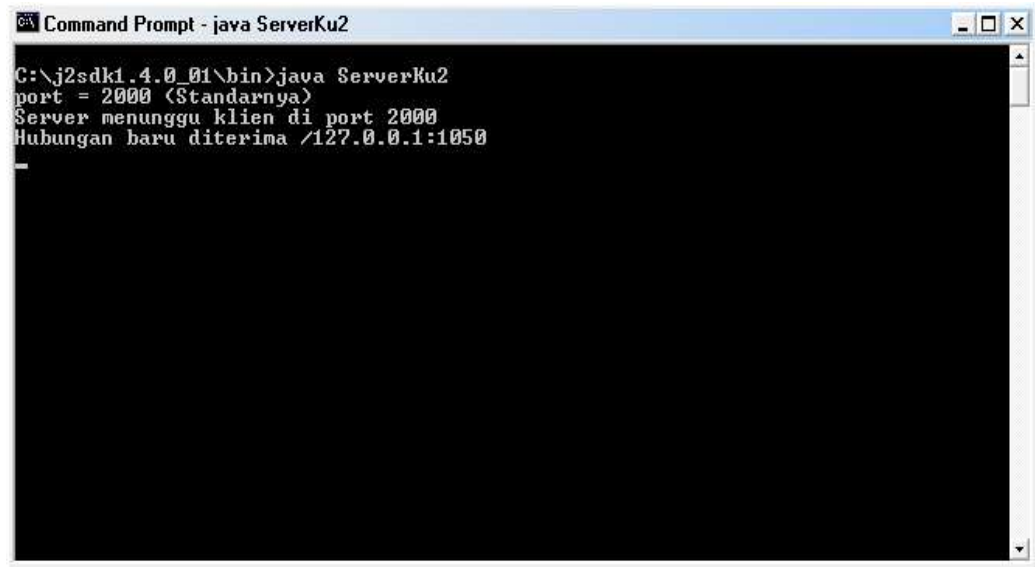
2. Kompile ClientKu2.java :
C:\> javac ClientKu2.java
Jalankan program :
C:\> java ClientKu2

Output :



```
C:\j2sdk1.4.0_01\bin>java ClientKu2
Terkoneksi ke server localhost/127.0.0.1:2000
```

Setelah ServerKu2.class dan ClientKu2.class di jalankan, sekarang kesempatan bagi anda untuk menulis pesan di klien dan nantinya akan di teruskan ke server.



```
Command Prompt - java ServerKu2

C:\j2sdk1.4.0_01\bin>java ServerKu2
port = 2000 (Standarnya)
Server menunggu klien di port 2000
Hubungan baru diterima /127.0.0.1:1050
_
```



```
Command Prompt - java ClientKu2

C:\j2sdk1.4.0_01\bin>java ClientKu2
Terkoneksi ke server localhost/127.0.0.1:2000
Ini dari klien, nanti akan dikirim ke server . . .
_
```



```
Command Prompt - java ServerKu2

C:\j2sdk1.4.0_01\bin>java ServerKu2
port = 2000 (Standarnya)
Server menunggu klien di port 2000
Hubungan baru diterima /127.0.0.1:1050
Ini dari klien, nanti akan dikirim ke server . . .
_
```

Nah, sekarang adalah suatu tantangan bagi anda untuk menyatukan dua program ini menjadi satu program saja. Selamat mencoba!.

BIOGRAFI PENULIS



M Zainal Arifin, lahir di Malang, 7 agustus 1980. Mengikuti pendidikan SD sampai Kuliah di Malang. Masih menempuh kuliah jurusan Matematika di Universitas Brawijaya Malang.

Menggeluti dunia pemrograman sejak tahun 1997 dan sekarang telah menulis beberapa artikel tentang jaringan dan pemrograman di majalah komputer. Sekarang sedang melakukan penelitian tentang teknologi suara dengan program java dengan teman-teman. Sangat tertarik dengan teknologi java terutama pada jaringan komputer dan mobile.

Penulis dapat dihubungi melalui :
Email: agendogget@yahoo.com