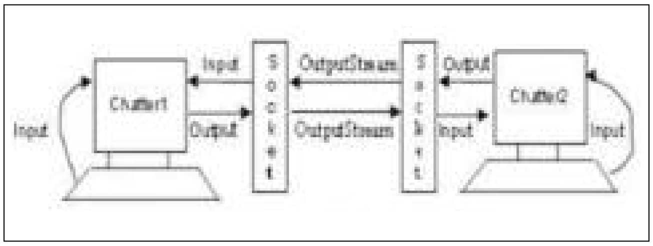
দুই বন্ধুর মধ্যে চ্যাট করুন জাভা দিয়ে

মো: আবদুল কাদের

চ্যাট করা বলতে সাধারণত গল্প করাকেই বোঝায়। তবে ইন্টারনেটে কানেক্টেড অবস্থায় দুই বা ততোধিক ইউজারের মধ্যে মেসেজ দেয়া-নেয়াই চ্যাটিং নামে বেশি পরিচিত। চ্যাটিং কয়েক ধরনের হতে পারে। যেমন– ওয়ান টু ওয়ান পদ্ধতি, ওয়ান টু মেনি ও মেনি টু মেনি। ওয়ান টু ওয়ান পদ্ধতিতে একজন শুধু অন্যজনের সাথে কথা বলতে পারেন, ওয়ান টু মেনি পদ্ধতিতে একজন অনেকজনের সাথে কথা বলতে পারেন এবং মেনি টু মেনি পদ্ধতিতে অনেক মানুষ পৃথকভাবে একজনের সাথে বা অনেকজনের সাথে কথা বলতে পারেন।

ইন্টারনেটের মাধ্যমে চ্যাট করার জনপ্রিয় সাইট হলো ইয়াহু, এমএসএন ইত্যাদি। এসব সাইটে ইংরেজিতে চ্যাট করা যায়। ইংরেজিতে চ্যাট করার উপকারিতা হলো একই সাথে ইংরেজি ভাষা বোঝার সাথে সাথে লেখার মতো ক্যাপাবিলিটিও তৈরি হয়। এজন্য দ্রুততার সাথে ইংরেজি আত্মস্থ করতে অনেকে এই সহজ ও কার্যকর পদ্ধতি কাজে লাগান। তবে এখন আমাদের দেশে বাংলাতে চ্যাট করার মতো অনেক সাইট রয়েছে। বেশিরভাগ সাইটে চ্যাট করার জন্য ওই ওয়েবসাইটের একটি ই-মেইল অ্যাড্রেস ও যেকোনো একটি পাসওয়ার্ড দরকার হয়। ইয়াহু সাইটে নির্দিষ্ট কারও সাথে চ্যাট করতে চ্যাট রুমে ঢুকে উপস্থিত চ্যাটারদের তালিকা থেকে যে ইউজারের সাথে চ্যাট করতে চাইবেন তার ওপর ডাবল ক্লিক করলেই একটি আলাদা উইন্ডো আসবে, যেখানে আপনার লেখা মেসেজ শুধু ওই চ্যাটারই দেখতে পাবে। এটিই ওয়ান টু ওয়ান পদ্ধতি।

এ পর্বে আমরা জাভা দিয়ে ইন্টারনেট ছাড়া শুধুম ল্যানে কানেক্টেড অবস্থায় ওয়ান টু ওয়ান পদ্ধতিতে চ্যাট করার প্রোগ্রাম দেখাব। এজন্য আমরা জাভার অ্যাডভান্সড ফিচার নেটওয়ার্কিংকে কাজে লাগাব। তবে জাভার প্রোগ্রামগুলো রান করার জন্য অবশ্যই আপনার কমপিউটারে Jdk সফটওয়্যার ইনস্টল থাকতে হবে। আমরা সফটওয়্যারটির Jdk1.4 ভার্সন ব্যবহার করব এবং প্রোগ্রামগুলো C: ড্রাইভের test ফোল্ডারে সেভ করব।



এই চ্যাটিং কাজ সম্পন্ন করার জন্য আমরা দুটি প্রোগ্রাম ডেভেলপ করব। একটি থাকবে চ্যাটার-১ হিসেবে ও অন্যটি চ্যাটার-২ হিসেবে। চ্যাটার-১ প্রোগ্রামটি কমপিউটারের নির্দিষ্ট পোর্টে চ্যাটার-২-এর জন্য অপেক্ষা করবে। চ্যাটার-২ উক্ত পোর্টে কানেক্ট হওয়ার পর উভয়ের মধ্যে কমিউনিকেশন শুরু হবে। এখানে লক্ষণীয়, চ্যাটার-১ প্রোগ্রামটি অবশ্যই আগে রান করতে হবে। এরপর চ্যাটার-২ প্রোগ্রাম রান করতে হবে। কেউ ইচ্ছা করলে দুটি প্রোগ্রামই একটি কমপিউটারে রান করতে পারেন অথবা দুটি কমপিউটারেও এটি রান করা যাবে।

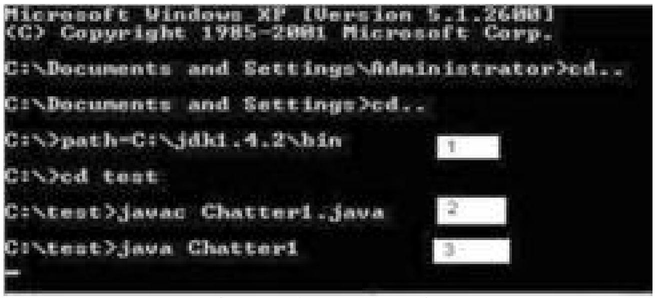
চ্যাটার-১ প্রোগ্রাম

এই প্রোগ্রামটি নোটপ্যাড বা অন্য কোনো এডিটরে টাইপ করে Chatter1.java নামে সেভ করুন।

```
   import java.net.*;
   import java.io.*;
   public class Chatter1 extends Thread
   {
   private ServerSocket serverSocket;
   public Chatter1(int port) throws IOException
   {
   serverSocket = new ServerSocket(port);         //1
   }
   public void run()
   {
   try
   {
   Socket client = serverSocket.accept();          //2
   DataInputStream              in              =              new
DataInputStream(client.getInputStream()); //3
   BufferedReader console = new BufferedReader
   (new InputStreamReader(System.in));
   DataOutputStream             out             =              new
DataOutputStream(client.getOutputStream());
   while(true)
   {
   String message = in.readUTF();
   System.out.println(“From Chatter2 : “+ message);        //4
   System.out.print(“Enter response: “);
   String response = console.readLine();          //5
   out.writeUTF(response);                        //6
   }
   }catch(IOException e){}
   }
   public static void main(String [] args)
   {
   try
   {
   Thread t = new Chatter1(5001);
   t.start();
   }catch(IOException e){ }
   }
   }
```

কোড বিশ্লেষণ

১নং চিহ্নিত লাইনে একটি সার্ভার সকেট তৈরি করা হয়েছে। এই সকেটটি একটি নির্দিষ্ট পোর্ট যেমন আমাদের ক্ষেত্রে 5001-এ উন্মুক্ত থাকবে। এরপর রান মেথডে ২নং চিহ্নিত লাইনে চ্যাটার-২-কে গ্রহণ করার জন্য accept() মেথড ব্যবহার করা হয়েছে। ৩নং লাইনে চ্যাটার-১-এর সকেট থেকে প্রোগ্রামে ইনপুট নেয়ার জন্য DataInputStream নেয়া হয়েছে। এখানে চ্যাটার-২ কানেক্ট হওয়ার পর যে মেসেজ দেবে তা-ই ইনপুট হিসেবে চ্যাটার-১-এর সকেট গ্রহণ করবে, যা ৪নং লাইনের সাহায্যে চ্যাটার-১ ▶



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>cd..

C:\Documents and Settings>cd..

C:\>path=C:\jdk1.4.2\bin

C:\>cd test

C:\test>javac Chatter1.java

C:\test>java Chatter1
_
```

চিত্র-১ : চ্যাটার-১ প্রোগ্রাম রানিং

দেখতে পাবে। ৫নং লাইনে কিবোর্ডের মাধ্যমে যে মেসেজ চ্যাটার-১ টাইপ করবে তা InputStreamReader-এর মাধ্যমে ইনপুট হিসেবে নেয়ার পর response ভেরিয়েবলে রাখা হচ্ছে, যা ৬নং লাইনের DataOutputStream-এর মাধ্যমে চ্যাটার-১-এর সকেটে পৌঁছে দেয়া হচ্ছে। চ্যাটার-২ কানেক্ট থাকলে মেসেজটি ইনপুট হিসেবে InputStream-এর মাধ্যমে গ্রহণ করবে। লক্ষণীয়, মেসেজ নেয়া ও দেয়া উভয়কেই while লুপে ব্যবহার করা হয়েছে এবং এর কন্ডিশন বা শর্ত সব সময় true থাকায় এ প্রোগ্রামটি চলতেই থাকবে যতক্ষণ পর্যন্ত না প্রোগ্রামটি বন্ধ করা হয়।

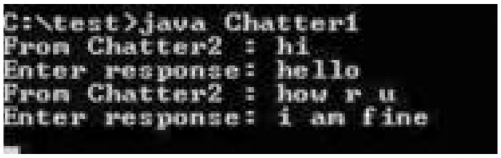
রান করা

চিত্র-১-এর ১নং চিহ্নিত লাইনে Jdk-এর পাথ C: ড্রাইভের Jdk ফোল্ডারের bin ফোল্ডারকে দেখানো হচ্ছে। কারণ, এই ফোল্ডারে জাভা রান করার সব প্রোগ্রাম রয়েছে। Jdk1.4.2 সফটওয়্যারটি ইনস্টল করার পর ফোল্ডারের নাম যদি অনেক বড় হয় বা ভিন্ন হয়, তাহলে ফোল্ডারের নাম রিনেইম করে Jdk1.4.2 ব্যবহার করতে হবে। এতে পাথ সেট করার সময় ভুল হওয়ার সম্ভাবনা কম থাকবে। জাভার কোনো প্রোগ্রাম রান করার জন্য পাথ সেটিং করতে হয়। এরপর C: ড্রাইভের test ফোল্ডারে ঢুকে ২নং লাইন অনুযায়ী জাভা ফাইলটিকে কম্পাইল করা হচ্ছে এবং ৩নং লাইন অনুযায়ী Chatter1 রানিং হচ্ছে।

```
C:\test>javac Chatter2.java

C:\test>java Chatter2 127.0.0.1
Enter response:
```

চিত্র-২ : চ্যাটার-২ প্রোগ্রাম রানিং



```
C:\test>java Chatter1
From Chatter2 : hi
Enter response: hello
From Chatter2 : how r u
Enter response: i am fine
```

কমিউনিকেশন অবস্থায় চ্যাটার-১

```
C:\test>java Chatter2 127.0.0.1
Enter response: hi
From Chatter1 : hello
Enter response: how r u
From Chatter1 : i am fine
Enter response: _
```

কমিউনিকেশন অবস্থায় চ্যাটার-২

চ্যাটার-২ প্রোগ্রাম

এই প্রোগ্রামটি নোটপ্যাডে টাইপ করে Chatter2.java নামে সেভ করুন ।

```
import java.net.*;
import java.io.*;
public class Chatter2 extends Thread
{
private String host;
private int port;
public Chatter2(String host, int port) throws IOException
{
this.host = host;
this.port = port;
}
public void run()
{
try
{
Socket socket = new Socket(host, port);          //socket
DataInputStream in = new
DataInputStream(socket.getInputStream());          //1
BufferedReader console = new BufferedReader(new
InputStreamReader(System.in)); //4
DataOutputStream out = new
DataOutputStream(socket.getOutputStream());
while(true)
{
System.out.print(“Enter response: “);
String response = console.readLine();
out.writeUTF(response);               //5
String message = in.readUTF();        //2
System.out.println(“From Chatter1 : “+ message);  //3
}
}catch(IOException e){}
}
public static void main(String [] args)
{
try
{
Thread t = new Chatter2(args[0], 5001);
t.start();
                         }catch(IOException e){}
                         }
                         }
```

কোড বিশ্লেষণ

এ প্রোগ্রামটির শুরুতেই হোস্ট এবং পোর্ট নামে দুটি ভেরিয়েবল ব্যবহার করা হয়েছে। যে কমপিউটারের সাথে প্রোগ্রামটি কমিউনিকেট করবে তার আইপি অ্যাড্রেসের জন্য হোস্ট এবং কোন পোর্টে যুক্ত হবে তার জন্য ব্যবহার করা হয়েছে পোর্ট ভেরিয়েবল। সকেট আইপি অ্যাড্রেস এবং পোর্ট নাম্বার অনুযায়ী চ্যাটার-১-এর সাথে কানেক্ট হবে। ১নং লাইনে চ্যাটার-১-এর দেয়া মেসেজ গ্রহণ করার জন্য InputStream নেয়া হয়েছে। মেসেজটি ২নং লাইনে message ভেরিয়েবলে রেখে ৩নং লাইনে তা দেখা হচ্ছে। ৪নং লাইনে চ্যাটার-২-এর টাইপ করা মেসেজ InputStream-এর মাধ্যমে গ্রহণ করে ৫নং লাইনে OutputStream-এর মাধ্যমে চ্যাটার-১-এর কাছে পাঠানো হচ্ছে।

রান করা

চ্যাটার-২-কে চিত্র-১-এর মতো পাথ সেটিং করে চিত্র-২-এর মতো কম্পাইল করে রান করতে হবে। লক্ষ রাখতে হবে, যদি দুটি প্রোগ্রামই একই কমপিউটারে রান করা হয়, তাহলে রান করার সময় লোকাল কমপিউটারের আইপি হিসেবে ১২৭.০.০.১ ব্যবহার করলেই হবে। তবে ভিন্ন ভিন্ন কমপিউটারে রান করার ক্ষেত্রে যে কমপিউটারে চ্যাটার-১ প্রোগ্রাম রানিং অবস্থায় থাকবে, সেই কমপিউটারের আইপি লিখতে হবে। কমপিউটারের আইপি অ্যাড্রেস জানার জন্য নতুন কমান্ড প্রম্পট ওপেন করে ipconfig দিয়ে এন্টার দিতে হবে।

দুটি প্রোগ্রাম রান করা অবস্থায় প্রথমে চ্যাটার-২ মেসেজ পাঠাবে। তারপর তার উত্তর চ্যাটার-১ লিখবে। শুধু নেটওয়ার্কে কানেক্টেড অবস্থায় থাকলেই এই সুবিধা পাওয়া যাবে কজ

ফিডব্যাক : balaith@gmail.com