ЎЗБЕКИСТОН РЕСПУБЛИКАСИ АХБОРОТ ТЕХНОЛОГИЯЛАРИ ВА КОММУНИКАЦИЯЛАРИНИ РИВОЖЛАНТИРИШ ВАЗИРЛИГИ

ТОШКЕНТ АХБОРОТ ТЕХНОЛОГИЯЛАРИ УНИВЕРСИТЕТИ

МУСТАҚИЛ ИШ

Мавзу: Тармоқда клиент-сервер архитектурасини реализация қилиш

ТОШКЕНТ 2017

Тармоқда клиент-сервер архитектурасини реализация қилиш

Режа

1. Тармоқда клиент-сервер иловаларини яратишда Java класслари
2. UDP протоколидан фойдаланиб клиент-сервер илова яратиш

Таянч иборалар: *клиент-сервер иловалари, java.net пакети, DatagramPacket, DatagramSocket, MulticastSocket, InetAddress, ServerSocket, Socket.*

Иловаларнинг клиент/сервер архитектурасида сервер хости маълумотлар базаси сўровларини қайта ишлаш каби хизматларни таъминлайди. Бунда клиент ва сервер ўртасидаги ўзаро алоқа ишончли бўлиши, маълумотлар йўқотилмаслиги, шунингдек, клиентга сервер томонидан юборилган маълумотлар тартибли етказилиши лозим.

Таъкидланганидек, TCP клиент/сервер иловалари учун нуқта-нуқта ишончли уланиш каналини таъминлайди ва унинг ёрдамида клиент ва сервер дастурлари уланиш ўранатадилар, сокетларни ўзаро боғла йдилар. Сокетлар - жараёнлар ўртасида ахборотлар алмашинувини таъминлаш учун дастурий интерфейс номидир. Ўзаро алоқада бўлган жараёнлар битта комьпютерда, шунингдек, тармоқ орқали уланган бир неча хостларда бажарилиши мумкин. Сокетлар тармоқ орқали ўрнатилган иловалар ўртасидаги алоқа каналини бошқариш учун ишлатилади. Ҳар бир TCP-уланиш иккита охирги нуқталари билан идентификация қилиниши мумкин. Шу тарзда, клиент ва серверниг кўплаб уланишлари таъминланиши мумкин. Сокет яратилганидан кейин, у орқали клиент ва сервер ўртасидаги кейинги ўзаро алоқа бажарилади.

1. Тармоқда клиент-сервер иловаларини яратишда Java класслари

Тармоқли дастурлаш учун Java класслари. Java дастурлаш тили java.net пакетини ўз ичига олиб, унинг класс ва интерфейслари тармоқда ишлашни қўллаб-қувватлашни таъминлайди ва тармоқли дастурлаш учун протоколлардан фойдаланади.

java.net пакети класслари. Тармоқ класслари масофадаги хост билан алоқа ўрнатиш ва уни узиш, маълумотлар пакетларини узатиш ва қабул қилиш ва тармоқ ресурсларига рухсат олиш каби вазифаларни бажариш методларини ўз ичига олади. Қуйида java.net пакетидаги баъзи класслар келтирилган:

- *DatagramPacket:* датаграмма пакети (datagram packet) нинг объекти бўлиб, ахборотлар манбаи ва қабул қилиниш адреси тўғрисидаги ахборот, манба ва қабул қилиш порт рақамлари каби маълумотларни ўз ичига олади. Ушбу ахборот датаграмма пакетларини тармоқ бўйлаб бир комьпютердан бошқасига узатиш учун фойдаланади.

- *DatagramSocket:* датаграмма пакетларини (datagram socket) узатиши ва қабул қилиши мумкин бўлган датаграмма сокети объектидир. DatagramSocket объекти томонидан узатилган пакетлар қабул қилувчига ҳар қандай тартибда келиши мумкин.

- *MulticastSocket:* гуруҳлар учун датаграмма пакетларини узатиш ва қабул қилиш учун фойдаланиладиган датаграмма сокетининг мультикаст (multicast) объектини яратади. D классдаги IP адреслар (224.0.0.0 ва 239.255.255.255 орасидаги IP адреслар) гуруҳларни яратиш учун фойдаланилади. Маълумотлар D классдаги IP-адресга узатилганида, гуруҳга уланган барча клиентлар узатилган маълумотларни қабул қиладилар. Ушбу класс клиентларга маълум гуруҳга кириш ва ундан чиқиш имкониятларини берувчи *joinGroup()* ва *leaveGroup()* методларини ўз ичига олади.

- *InetAddress:* IP-адрес ва хост номи (host name) дан иборат бўлган ахборотларни ўз ичига оладиган объектни яратади.

- *ServerSocket:* клиент сўровлари “эшитадиган” сервер сокети объектини яратади. ServerSocket объектлари клиент сўровларини қабул қилиш учун порт рақамидан фойдаланади.

- *Socket:* серверга сўровларни юбориш учун ServerSocket класси объекти билан боғланувчи клиент сокети объектини яратади.

- *URL:* Интернетга файл ёки бошқа ресурсни юбориши мумкин бўлган объектни тақдим қилади.

java.net пакети, шунингдек, амалларни бажариш вақтида тармоқ хатоларини қайта ишлаш имконини берувчи истисно ҳолатларни қайта ишлаш классларини ўз ичига олади. Қуйида java.net пакетида мавжуд бўлган истисно ҳолатларни қайта ишловчи бир неча класслар келтирилган:

- *BindException:* ушбу истисно ҳолатлар объекти локал адрес сокети ёки порти билан алоқа ўрнатишда хатолик юзага келганида генерацияланади. Масалан, локал адресга рухсат бўлмаганида ёки сўралган порт ишлатилаётган ҳолатларда.

- *ConnectException:* ушбу истисно ҳолатлар объекти сокетнинг масофадаги IP-адрес ва порт билан боғланишида хатолик юзага келганида генерацияланади. Масалан, масофадан уланиш имкони бўлмаганида.

- *MalformedURLException:* ушбу истисно ҳолатлар объекти URL кучга эга бўлмаган протоколга эга бўлганида ёки URL “ссылка” си муваффақиятли қайта ишланмаганида генерацияланади.

- *UnknownHostException:* ушбу истисно ҳолатлар объекти хостнинг IP-адресининг аниқланиши мумкин бўлмаганида ёки мавжуд бўлмаганида генерацияланади.

InetAddress класси:

InetAddress класси хостлар номи ва улар билан боғлиқ бўлган ва тармоқ иловаларини яратишда зарур бўлган IP-адресларни аниқлаш имконини беради. InetAddress объектлари InetAddress классида эълон қилинган статистик

методлардан фойдаланган ҳолда инициализация қилинади. Қуйида InetAddress объектларини инициализация қилишда фойдаланиладиган методлардан баъзилари келтириган:

- *public static InetAddress getLocalHost():* локал комьпютернинг IP-адресини ўз ичига оладиган InetAddress объектини қайтаради.

- *public static InetAddress getByName(String host):* String сифатида узатилган хост номининг IP-адресини ўз ичига олувчи InetAddress объектини қайтаради.

- *public static InetAddress[] getAllByName(String host):* String сифатида узатилган хост номи учун IP-адресни ўз ичига олувчи InetAddress объект массивини қайтаради.

- *getLocalHost(), getByName(), getAllByName() ва getBy Adress()* UnknownHostException истисно ҳолатини чақиради.

InetAddress классида аниқланган ностатик методлардан InetAddress объектларини инициализация қилинганидан кейин фойдаланиш мумкин. Ностатик методлар InetAddress классида аниқланган ва қуйида келтирилган:

- *public boolean equals(Object obj):* InetAddress объекти параметр сифатида узатилган obj объекти билан бир хил IP адресга эга бўлганида *true* ни қайтаради.

- *public byte[] getAddress():* InetAddress объектининг IP адресини *byte* массиви кўринишида қайтаради. Масалан, InetAddress объектининг IP-адреси 194.85.160.58 бўлса, ушбу метод кейинги *byte* массивини қайтаради:

ipaddress[0] = 194;
ipaddress[1] = 85;
ipaddress[2] = 160;
ipaddress[3] = 58.

- *public String getHostAddress(): InetAddress* объектининг IP-адресини *String* сифатида қайтаради.

- *public String toString():* хостнинг IP-адресини *string* сифатида хост/ IP-адрес ном форматида қайтаради.

InetAddress объектини инициализация қилиш ва фойдаланиш намунасини кўриб чиқамиз:

```
import java.net.*;
class InetAddressClass {
public static void main(String arg[]) {
try {
/*
* getLocalHost() дан фойдаланиб, InetAddress объекти яратилади.
* InetAddress классининг статик методи
*/
InetAddress address = InetAddress.getLocalHost();
/*хост IP-адресининг қабул қилиниши*/
String addressHost = address.getHostAddress();
```

```
/* хост IP-адресининг чиқиши */
System.out.println("хост IP-адресининг чиқиши"+addressHost);
// хост номининг чиқиши
System.out.println("хост номининг чиқиши"+address.getHostName());
} catch (UnknownHostException e) {
System.out.println("Error");
}
}
}
```

Ушбу намунада *InetAddress* классининг объекти аниқланади. *InetAddressClass* классининг *main()* методида address объекти *getLocalHost()* статик методидан фойдаланиб инициализация қилинади. *getHostAddress()* методи хост IP-адресини *string* сифатида қайтаради. *getHostName()* методи хост номини string сифатида қайтаради. InetAddressClass классининг бажарилиш натижасида IP-адрес ва хост номи чиқарилади.

2. UDP протоколидан фойдаланиб илова яратиш

Юқорида таъкидланганидек, Java UDP датаграмма ва TCP/IP сокетларидан фойдаланган ҳолда тармоқ иловаларини ишлаб чиқиш имконини беради. UDP сокетлари иловаларнинг тармоқ орқали ўзаро ишлаши учун UDP протоколидан фойдаланади. UDP уланиш ўрнатиш зарурияисиз тез ва ишончсиз протоколдир. *java.net* пакети Java иловасида UDP сокетидан фойдаланиш имконини берувчи қуйидаги икки классни ўз ичига олади:

- DatagramPacket класси
- DatagramSocket класси

DatagramPacket ва DatagramSocket класслари

DatagramPacket объекти тармоқ орқали узатиладиган ёки қабул қилинадиган датаграмма пакетларидан иборат бўлган маълумотлар контейнеридир. Қуйидаги конструкторлар *DatagramPacket* объектларини инициализация қилиш учун ишлатилади:

- *public DatagramPacket(byte[] buffer, int buffer_length):* маълумотларни byte массивида қабул қиладиган ва сақлайдиган DatagramPacket объектини яратади. Byte массивининг буфер узунлиги buffer_length иккинчи параметри томонидан берилади.

- *public DatagramPacket(byte[] buffer, int buffer_length, InetAddress address, int port):* берилган узунликдаги маълумотлар пакетларини узатувчи *DatagramPacket* объектини яратади. Маълумотлар пакетлари компютерга IP-адрес ва параметр сифатида бериладиган порт номери билан узатилади.

DatagramPacket классида аниқланган методлардан *DatagramPacket* класси объекти инициализация қилинганидан кейин фойдаланилиши мумкин. 23.1-жадвалда DatagramPacket класси методлари келтирилган.

23.1-жадвал

DatagramPacket класси методлари

Метод	Таъриф
public InetAddress getAddress ()	Датаграмма пакети узатиладиган ёки датаграмма пакети қабул қилинадиган компютер IP-адресини ўз ичига олувчи InetAddress объектини қайтаради
public byte [] getData ()	Маълумотларни ўз ичига олган byte буфер массивини қайтаради
public int getLength ()	Маълумотларни ўз ичига олган буфер массивининг узунлигини қайтаради
public int getPort ()	Датаграмма пакети узатиладиган ёки қабул қилинадиган компютер порт номерини қайтаради
public void setAddress (InetAddress address)	Датаграмма пакети узатилиши керак бўлган машинанинг IP-адресини ўрнатади
public void setPort (int port)	Byte массивини пакет учун маълумотлар сифатида ўрнатади
public void setLength (intlength)	Масофадаги хостда порт номерини ўрнатади

DatagramSocket класси DatagramPacket объектларини боқшариш учун функционалликни ўз ичига олади. DatagramPacket объектлари DatagramSocket дан фойдаланган ҳолда сақланган маълумотларни узатади ва қабул қилади. Қуйидаги конструкторлар DatagramSocket объектини инициализация қилиш учун ишлатилади:

- *public DatagramSocket():* DatagramSocket объектини яратади ва уни локал комьпютердаги рухсат этилган порт билан боғлайди.
- *public DatagramSocket(int port):* объектни яратади ва уни параметрда берилган локал хостдаги порт билан боғлайди.
- *public DatagramSocket(int port, InetAddress address):* объектни яратади ва уни берилган хост порти билан боғлайди.

DatagramSocket классининг конструктори SocketException истисно ҳолатини чақиради. 23.2-жадвалда ахборотларни DatagramSocket объектидан олиш учун фойдаланиладиган DatagramSocket класс методлари келтирилган:

23.2-жадвал

DatagramSocket объектидан олиш учун фойдаланиладиган DatagramSocket класс методлари

Метод	Таъриф
public InetAddress getInetAddress ()	DatagramSocket объекти боғданадиган IP-адресни ўз ичига олган InetAddress объектини қайтаради

public InetAddress getLocalAddress ()	DatagramSocket объекти боғланадиган локал хост IP-адресини ўз ичига олган InetAddress объектини қайтаради
public int getLocalPort ()	DatagramSocket объекти боғланадиган локал хост портини тақдим қиладиган бутун қийматни қайтаради
public void bind (SocketAddress address)	DatagramSocket объектини SocketAddress объекти билан боғлайди
public void close ()	DatagramSocket объектини ёпади
public void connect (InetAdress address, int port) public void disconnect ()	DatagramSocket объектини берилган IP-адрес ва порт билан боғлайди DatagramSocket объектини узади
public boolean isBound ()	DatagramSocket объекти порт билан боғланган бўлса, true ни қайтаради
public Boolean isClosed ()	DatagramSocket объекти ёпилганида
public boolean isConnected ()	DatagramSocket IP-адрес билан боғланганида true ни қайтаради
public void receive (DatagramPacket packet)	DatagramSocket жорий объектидан датаграмма пакетини қабул қилади
public void send (DatagramPacket packet)	DatagramSocket жорий объектидан датаграмма пакетини узатади

UDP серверини яратиш

UDP сервери клиент иловаларига хизмат кўрсатиш учун UDP протоколидан фойдаланадиган тармоқ иловасидир. UDP серверини яратиш учун DatagramPacket объектларини клиентлардан қабул қиладиган DatagramSocket объектидан фойдаланилади. UDP серверини яратиш учун қуйидагиларни бажариш керак бўлади:

- DatagramSocket объектидан фойдаланиб сокет яратиш;
- DatagramPacket класс объектини яратиш ва клиент хабарларини қабул қилиш учун receive() методидан фойдаланиш;
- DatagramPacket класс объектини яратиш ва клиент хабарларини узатиш учун send() методидан фойдаланиш;
- main() методида UDP сервер класси конструкторини чақириб, серверни ишга тушириш.

Қуйидаги код фрагментидан DatagramSocket объектини яратиш учун фойдаланиш мумкин:

```
DatagramSocket:
try
{
DatagramSocket socket = new DatagramSocket(1501);
}
catch(SocketException send)
{
System.out.println("Error");
```

```
}
```

Юқоридаги код фрагментида DatagramSocket классининг socket объекти 1501 рақамли порт билан боғланади.

Датаграмма пакетини қабул қилувчи DatagramPacket объекти датаграммаларни сақлаш учун буферга эга. Қуйидаги код фрагментидан датаграмма пакетларини қабул қилувчи DatagramPacket объектини яратиш учун фойдаланиш мумкин:

```
try
{
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
socket.receive(packet);
}
catch(Exception e){
System.out.println ("Error");
}
```

Олдинги код фрагментида socket объектидан пакетни қабул қилиш учун receive() методини чақирадиган DatagramPacket классининг packet объекти яратилади.

Қабул қилувчига юборилган DatagramPacket объекти қабул қилинган маълумотлар объектидан фарқ қилади. Ушбу DatagramPacket объекти хостнинг пакет юборилган IP-адреси ва порт номерига эга бўлади. Қуйидаги код фрагментидан DatagramPacket объектини берилган манзилга юбориш учун фойдаланиш мумкин:

```
try
{
DatagramPacket packet = new DatagramPacket(buffer, length, address, port);
socket.send(packet);
}
catch(Exception e)
{
System.out.println ("Error");
}
```

Олдинги код фрагментида 4 та параметрни қабул қилувчи DatagramPacket классининг янги packet объекти яратилади.

- buffer: маълумотларга эга бўлган буферни беради.
- length: буфер узунлигин байтларда беради.
- address: датаграмма юборилган адресни беради.

- port: масофадаги комьпютер датаграммани қабул қилишда ишлатадиган порт номерини беради.

DatagramSocket классининг send() методи адресга DatagramPacket объектини юборади.

UDP серверини ишга тушириш учун main() методидаги конструктор класси чақирилади. Қуйидаги код фрагментини UDP серверини ишга тушириш учун ишлатиш мумкин:

```
public static void main(String args[]) throws Exception
{
/* серверни ишга тушириш */
new UDPServer();
}
```

Олдинги код фрагментида UDP сервер иловасини иш туширувчи UDPServer класс объекти яратилади.

Қуйидаги код клиентдан қабул қилинган хабарларни акс эттиради ва клиентга жавоб хабарларни узатувчи UDPServer ни яратиш имконини беради:

```
import java.io.*;
import java.net.*;
public class UDPServer
{
/*ўзгарувчилар эълон қилинади*/
DatagramSocket socket = null;
BufferedReader in = null;
String str = null;
byte[] buffer ;
DatagramPacket packet;
InetAddress address;
int port;
/* UDPServer класс конструктори */
public UDPServer() throws IOException
{
/* клиент сўровларини 1501 рақамли портга қабул қилувчи DatagramSocket объекти яратилади*/
socket = new DatagramSocket(1501);
/* call() методи яратилади*/
call();
}
public void call()
{
try
```

```
{
while (true)
{
buffer= new byte[256];
/* DatagramPacket  объекти инициализация қилинади*/
packet = new DatagramPacket( buffer, buffer.length);
/* DatagramSocket классининг receive() методидан фойдаланиб, датаграмма пакетлари узатилади*/
socket.receive(packet);
if(packet == null) break;
System.out.println("Request string for sending to client ");
try
{
/* консолдан маълумотларни ўқийдиган кирувчи оқим яратилади*/
in = new BufferedReader(new InputStreamReader(System.in));
}
catch(Exception e)
{
System.out.println("Error : " + e);
}
str = in.readLine();
buffer = str.getBytes();
address = packet.getAddress();
port = packet.getPort();
packet = new DatagramPacket(buffer, buffer.length, address, port);
/*  датаграмма пакети узатилади*/
socket.send(packet);
}
/* оқим ва сокет ёпилади*/
in.close();
socket.close();
}
catch(Exception e)
{
System.out.println("Error : " + e);
}
}
public static void main(String args[]) throws Exception
{
/* сервер ишга туширилади*/
new UDPServer();
}
```

}

Кўриб чиқилган кодда UDPServer класс конструкторида DatagramSocket классининг socket объекти яратилади. Сокет 1501 портга инициализация қилингандан сўнг, клиент/сервер ўзаро алоқасини бошқарувчи receive() ва send() методларидан иборат бўлган call() методи чақирилади. Олдинги код UDPServer.java сифатида сақланади ва java.exe дастури билан компиляция қилинади.

UDP клиентини яратиш

UDP клиенти UDP протоколидан серверга сўровларни юбориш ва сервер иловасидан жавобларни қабул қилиш учун фойдаланадиган иловадир. Фойдаланувчининг UDP-иловасида UDP серверидан хабарларни қабул қиладиган DatagramSocket класс объектини яратиш зарур, бу учун эса қуйидагиларни бажариш лозим бўлади:

1. Сервер билан уланиш ўрнатиш учун DatagramSocket класс объектидан фойдаланувчи сокетини яратиш.

2. DatagramPacket класс объектини яратиш ва хабарларни серверга юбориш учун send() методидан фойдаланиш.

3. DatagramPacket класс объектини яратиш ва сервердан юборилган хабарларни қабул қилиш учун receive() методидан фойдаланиш.

Қуйидаги код фрагментидан клиент иловаси учун DatagramSocket класс объектини яратиш учун фойдаланиш мумкин:

```
try
{
DatagramSocket socket = new DatagramSocket();
}
catch(Exception e)
{
System.out.println("Error");
}
```

Олдинги код фрагментида конструктор DatagramSocket класс объектини ҳар қандай рухсат этилган локал порт билан боғлайди, чунки параметрда порт номери кўрсатилмаган бўлади.

Ушбу DatagramSocket объекти сўров юбориладиган сервернинг IP-манзили ва порт рақамини ўз ичига олади. Қуйидаги код фрагменти DatagramPacket объектини берилган серверга юбориш учун ишлатилади:

```
try
{
DatagramPacket packet = new DatagramPacket(buffer, length, address, port);
```

```
socket.send(packet);
}
catch(Exception e)
{
System.out.println("Error");
}
```

Олдинги код фрагметида 4 та параметрни қабул қилувчи DatagramPacket конструктори ёрдамида packet объекти яратилади. DatagramSocket классининг send() методи DatagramPacket класс объектини серверга юборади.

Қуйидаги код фрагментидан сервердан датаграмма пакетларини қабул қилувчи DatagramPacket объектини яратишда фойдаланиш мумкин:

```
try
{
DatagramPacket packet = new DatagramPacket(buffer, buffer.length);
socket.receive(packet);
str = new String(packet.getData());
System.out.println("Принятое сообщение : "+str);
}
catch(Exception e){
System.out.println("Ошибка");
}
```

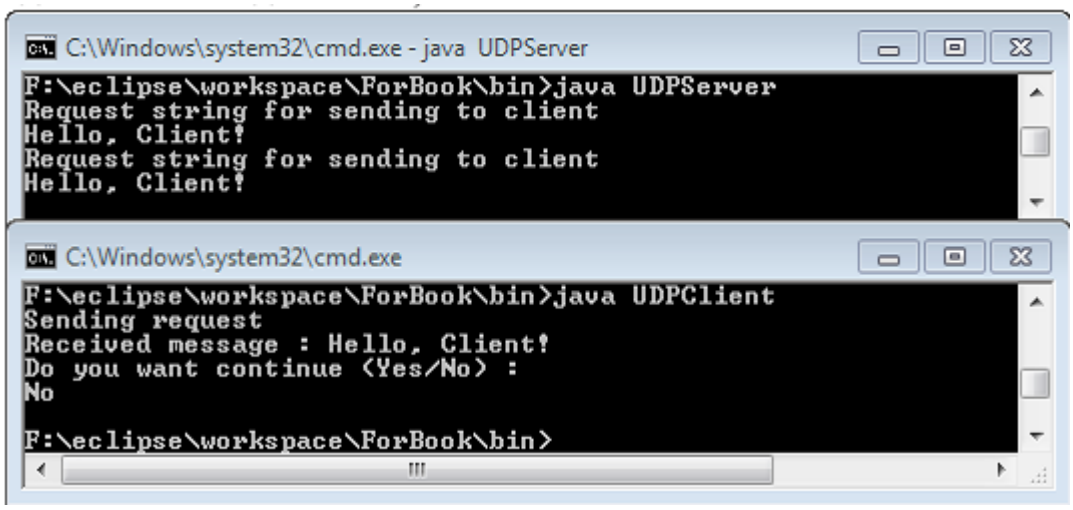
Олдинги код фрагментида receive() методини чақирувчи DatagramPacket класснинг packet объекти яратилади. getData() методи packet объектидан маълумотларни қабул қилади ва уларни string турдаги ўзгарувчида сақлайди.

Қуйидаги код UDPServer га хабарларни узатувчи ва қабул қилувчи UDPClient классини яратиш учун ишлатилади:

```
import java.io.*;
import java.net.*;
public class UDPClient
{
/* ўзгарувчилар эълон қилинади */
static DatagramSocket socket;
static InetAddress address;
static byte[] buffer;
static DatagramPacket packet;
static String str, str2;
static BufferedReader br;
public static void main(String arg[]) throws Exception{
/* консолдан ўқиладиган кирувчи оқим яратилади */
```

```
br = new BufferedReader(new InputStreamReader(System.in));
while(true)
{/* янги DatagramSocket объекти яратилади ва порт билан боғланади*/
socket = new DatagramSocket();
address = InetAddress.getByName("127.0.0.1");
buffer = new byte[256];
packet = new DatagramPacket(buffer, buffer.length, address, 1501);
/* DatagramPacket серверга юборилади */
socket.send(packet);
System.out.println("Sending request ");
packet = new DatagramPacket(buffer, buffer.length);
/* сервердан DatagramPacket қабул қилинади */
socket.receive(packet);
/* датаграммалар пакети объектидан маълумотлар қабул қилинади */
str = new String(packet.getData());
System.out.println("Received message : "+str.trim());
System.out.println("Do you want continue (Yes/No) : ");
str2 = br.readLine();
/* while циклидан чиқиш */
if(str2.equals("No")) break;
}
/* сокет объекти ёпилади */
socket.close();
}
}
```

Кўриб чиқилган кодда UDPClient классининг main() методидаги while цикли UDPServer га сўровларни юборади ва қабул қилади. Socket объектининг send() методи серверга сўровларни юборади, receive() методи эса UDPServer дан хабарларни қабул қилади. 23.1-расмда UDPClient клиент ва UDPServer сервернинг ўзаро алоқасидан олинган натижа келтирилган. (ҳар бир иловани бажариш учун алоҳида дарча очилади).



23.1-расм. UDPClient натижаси

Келтирилган расмда «Sending request» хабари клиент UDPServer га сўров юборганлигини кўрсатади. UDPServer клиент сўровини қабул қилганидан кейин, консол ойнасига «Request string for sending to client» хабарини чиқаради. Сервер фойдаланувчидан хабар олганидан кейин UDPServer натижаси сервер ойнасида акс эттирилади. UDPServer консол ойнасига «Hello, Client!» хабарини чиқаради ва клиентга юборади. Агар фойдаланувчи клиент консолига «No» ни киритса, UDPClient жараёни тугалланади.

Server классини яратиш

Server класси 233.0.0.1 гуруҳда рўйхатга олинган барча клиентларга хабарларни юбориш учун мўлжалланган. Server классини яратиш ичига қуйидаги асосий босқичлар киради:

1. DatagramSocket класси асосида сокетни яратиш. Сервер сокети хабарларини юбориш вазифасини бажаради.

2. Сервер адресини намойиш қилувчи InetAddress класс объектини яратиш. Хабарларни гуруҳли (multicast) юбориш учун адреслар 224.0.0.0 – 239.255.255.255 диапазондан танланади. Бизнинг иловамизда 233.0.0.1 адреси ишлатилади.

3. Клавиатурадан хабарларни киритишни ташкиллаштириш ва киритилган маълумотларга эга бўлган ва пакетларни гуруҳнинг барча клиентларига юбориш учун DatagramSocket классининг send() методидан фойдаланувчи DatagramPacket синфининг packet объектини яратиш.

1) Сервер классини яратиш учун сичқончанинг ўнг тугмасини Package Explorer ойнасидаги каталог src устига босинг ва New/Class ни танланг.

2) Finish ни босинг. Пайдо бўлган ойнада пакет номи (Package) сифатида ru.ifmo.socket ни кўрсатинг, класс номи (Name) сифатида эса Server ни ёзинг.

Server классининг коди қуйида келтирилган:

```
package ru.ifmo.socket;
import java.io.*;
import java.net.*;
public class Server {
private BufferedReader in = null;
private String str = null;
private byte[] buffer;
private DatagramPacket packet;
private InetAddress address;
private DatagramSocket socket;
public Server() throws IOException {
System.out.println("Sending messages");
// клиент сўровларини қабул қилиш учун
//DatagramSocket объекти яратилади
socket = new DatagramSocket();
// Гуруҳда рўйхатга олинган барча клиентларга
// хабар юбориш учун transmit() методини чақириш
transmit();
}
public void transmit() {
try {
// консолдан маълумотларни қабул қилиш
//учун кириш оқими яратилади
in = new BufferedReader(new InputStreamReader(System.in));
while (true) {
System.out.println("клиентларга юбориш учун сатрни киритинг: ");
str = in.readLine();
buffer = str.getBytes();
address = InetAddress.getByName("233.0.0.1");
// 1502 рақамли портга датаграмма пакетини юбориш
packet = new DatagramPacket(buffer, buffer.length, address, 1502);
// гуруҳдаги барча клиентларга хабарларни юбориш
socket.send(packet);
}
} catch (Exception e) {
e.printStackTrace();
} finally {
try {
// оқим ва сокетни ёпиш
in.close();
socket.close();
} catch (Exception e) {
```

```
e.printStackTrace();
}
}
}
public static void main(String arg[]) throws Exception {
// серверни ишга тушириш
new Server();
}
}
```

Client классини яратиш

Client класси клиентга сервердан хабарлар қабул қилиш учун 233.0.0.1 гуруҳга кириш имконини беради. Client классини яратиш қуйидаги асосий босқичлардан иборат:

1. MulticastSocket класси ёрдамида гуруҳли хабарларни кўриш учун сокет яратиш. Клиент сокети хабарларни қабул қилиш вазифасини бажаради.

2. Сервер адресидан иборат бўлган InetAddress объектини яратиш ва ушбу сервернинг гуруҳига joinGroup сокет методи ёрдамида кириш.

3. Сокетдан датаграмма пакетлари (DatagramPacket) ни киритишни ташкиллаштириш ва экранда олинган маълумотларни акс эттириш.

1) Клиент классини яратиш учун Package Explorer ойнасидаги src каталогнинг ru.ifmo.socket пакетига сичқончанинг ўнг тугмасини босинг ва New/Class ни танланг.

2) Пайдо бўлган ойнада класс номи (Name) сифатида Client ни ёзинг. Finish ни босинг.

Client классининг коди қуйида келтирилган:

```
package ru.ifmo.socket;
import java.net.*;
public class Client {
private static InetAddress address;
private static byte[] buffer;
private static DatagramPacket packet;
private static String str;
private static MulticastSocket socket;
public static void main(String arg[]) throws Exception {
System.out.println("сервер хабарини кутиш ");
try {
// 1502 порт рақамидан фойдаланиб, гуруҳдан
//маълумотларни қабул қилиш учун MulticastSocket объектини яратиш
socket = new MulticastSocket(1502);
address = InetAddress.getByName("233.0.0.1");
```

```
// клиентни гуруҳда рўйхатга олиш
socket.joinGroup(address);
while (true) {
buffer = new byte[256];
packet = new DatagramPacket(buffer, buffer.length);
// сервердан маълумотларни қабул қилиш
socket.receive(packet);
str = new String(packet.getData());
System.out.println("хабар қабул қилинди: " + str.trim());
}
} catch (Exception e) {
e.printStackTrace();
} finally {
try {
// клиентни гуруҳдан ўчириш
socket.leaveGroup(address);
// сокетни ёпиш
socket.close();
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

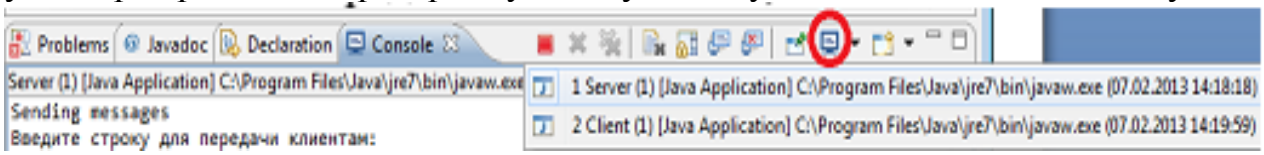
Ишга тушириш ва тестдан ўтказиш

Ҳар бир яратилган Client ва Server класслар main() методини ўз ичига олиб, тармоққа уланган алоҳида компьютерда ишга туширилиши мумкин бўлган алоҳида илова бўлиб ҳисобланади. Бунда гуруҳли маълумотларни (multicasting scope) узатиш соҳаси сервер тармоқости билан чегараланади. Бизнинг ҳолатимизда клиент ва сервер ролини битта компьютер бажаради.

1) Package Explorer ойнасидаги Client классга сичқончанинг ўнг тугмасини босинг ва контекст менюдан Run As/Java Application командасини танланг.

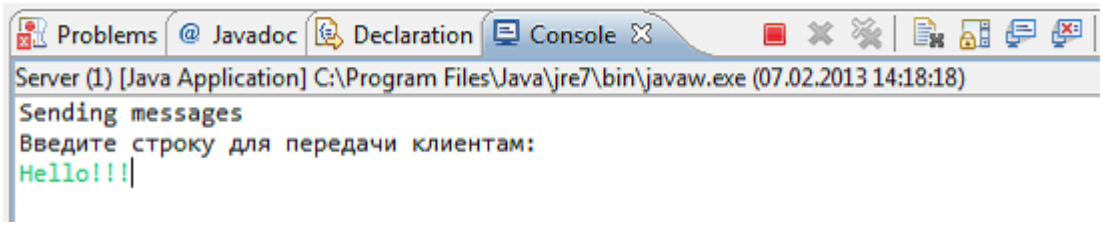
Худди шуни Server класси билан ҳам такрорланг.

1) Натижада, 23.2-расмдагидек, Display Selected Console рўйхат тугмалари ёрдамида бир-бирига ўтиш мумкин бўлган иккита илова ишга тушади.



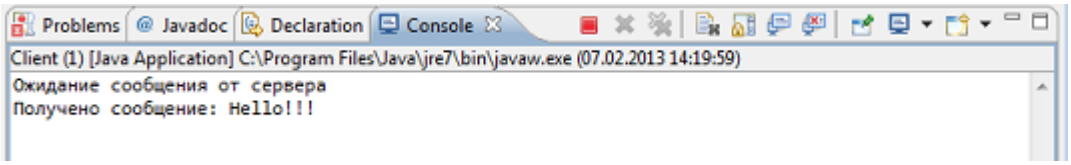
23.2-расм. Консолни танлаш

2) Сервер консолини танланг, "Hello!!!" сатрини киритинг ва юбориш учун Enter ни босинг, 23.3-расмдагидек.



23.3-расм. Сервер консолларига хабарни киритиш.

4) Клиент консолига ўтинг ва клиент хабарни қабул қилганлигини кўринг (23.4-расмдагидек).



23.4-расм. Клиент консолларига хабарни чиқариш.

5) Илова ишини тўхтатиш Terminate тугмасини босиш орқали амалга оширилади.

Назорат саволлари

1. java.net пакети.
2. DatagramPacket класси.
3. DatagramSocket класси.
4. MulticastSocket класси.
5. InetAddress класси.
6. ServerSocket класси.
7. Socket класси.

Адабиётлар ва интернет ресурслар

1. Computer networking : a top-down approach / James F. Kurose, Keith W. Ross.—6th ed. 2013. by Pearson Education, Inc., publishing as Addison-Wesley.
2. TCP/IP protocol suite / Behrouz A. Forouzan.—4th ed. Published by McGraw-Hill, a business unit of The McGraw-Hill Companies, Inc., 1221 Avenue of the Americas, New York, NY 10020. Copyright © 2010
3. Java Network Programming, Fourth Edition by Elliotte Rusty Harold. 2014. Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol.
4. The Definitive Guide to Linux Network Programming Copyright © 2004 by Keir Davis, John W. Turner, Nathan Yocom. 2011.