Java Network Programming

Сокет

- Низкоуровневый API для пересылки байтов по сети
- Поддерживаются протоколы TCP и UDP
- Поддерживается адресация IPv4 (192.168.5.10) и IPv6 (2001:0db8:11a3:09d7:1f34:8a2e:07a0:765d)

java.net.DatagramSocket (клиент)

```
try ( DatagramSocket s = new DatagramSocket ())
{
     DatagramPacket p = new DatagramPacket (
          buf, buf.length , remoteAddress);
     s.send (p);
}
```

java.net.DatagramSocket (сервер)

```
try(DatagramSocket s = new DatagramSocket (port)){
     byte [] buf = new byte [1024];
     DatagramPacket p = new DatagramPacket (
          buf, buf.length);
     s.receive (p);
}
```

TCP. Схема работы

- Сервер создает «серверный сокет», слушающий конкретный порт
- Сервер вызывает метод accept(). Данный метод ожидает до тех пор пока клиент не подключится
- Клиент создает «клиентский сокет» и задает IP адрес и порт сервера.
- Конструктор сокета в процессе работы пытается установить соединение
- На стороне сервера метод accept() возвращает ссылку на Socket, по которому сервер может общаться с подключившимся клиентом.

java.net.Socket (TCP)

SN	Methods with Description
1	**public Socket(String host, int port) throws UnknownHostException, IOException.** This method attempts to connect to the specified server at the specified port. If this constructor does not throw an exception, the connection is successful and the client is connected to the server.
2	**public Socket(InetAddress host, int port) throws IOException** This method is identical to the previous constructor, except that the host is denoted by an InetAddress object.
3	**public Socket(String host, int port, InetAddress localAddress, int localPort) throws IOException.** Connects to the specified host and port, creating a socket on the local host at the specified address and port.
4	**public Socket(InetAddress host, int port, InetAddress localAddress, int localPort) throws IOException.** This method is identical to the previous constructor, except that the host is denoted by an InetAddress object instead of a String
5	**public Socket()** Creates an unconnected socket. Use the connect() method to connect this socket to a server.

java.net.Socket (TCP)

```
Socket socket = new Socket ("localhost", 11111);

OutputStream os = socket.getOutputStream();
os.write(requestBytes);
os.flush();

InputStream is = socket.getInputStream();
is.read(responseBytes);
```

java.net.ServerSocket

SN	Methods with Description
1	**public ServerSocket(int port) throws IOException** Attempts to create a server socket bound to the specified port. An exception occurs if the port is already bound by another application.
2	**public ServerSocket(int port, int backlog) throws IOException** Similar to the previous constructor, the backlog parameter specifies how many incoming clients to store in a wait queue.
3	**public ServerSocket(int port, int backlog, InetAddress address) throws IOException** Similar to the previous constructor, the InetAddress parameter specifies the local IP address to bind to. The InetAddress is used for servers that may have multiple IP addresses, allowing the server to specify which of its IP addresses to accept client requests on
4	**public ServerSocket() throws IOException** Creates an unbound server socket. When using this constructor, use the bind() method when you are ready to bind the server socket

java.net.ServerSocket

```
ServerSocket server = new ServerSocket(11111);
Socket socket = server.accept();

InputStream is = socket.getInputStream();
is.read(requestBytes);

OutputStream os = socket.getOutputStream();
os.write(responseBytes);
os.flush();
```

Обработка большого количества клиентов

```
while (true) {
    accept a connection;
    deal with the client;
}
```

Один поток…

Обработка большого количества клиентов

```
while (true) {
    accept a connection;
    create a thread to deal with the client;
}
```

Тратим время на создание потоков…

Стоит переиспользовать созданные потоки.

Обработка большого количества клиентов

```
while (true) {
    accept connection;
    create task which will deal with the client;
}
```