

Lập trình socket

Nội dung

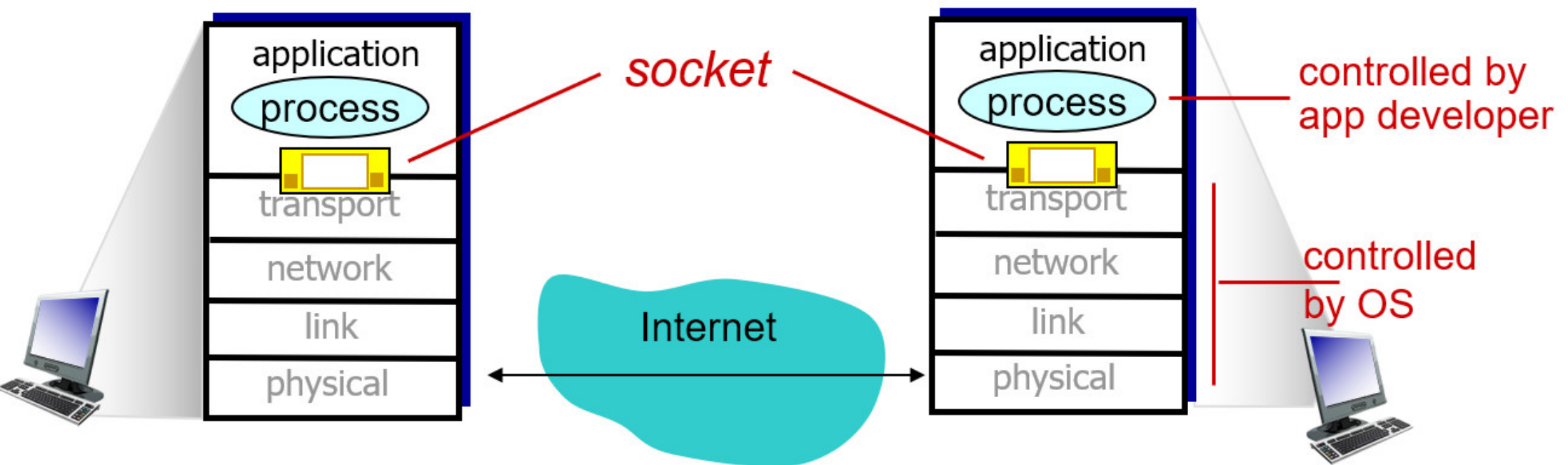
- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
- ❑ Bài tập

Nội dung

- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
- ❑ Bài tập

Khái niệm socket

- ❑ Socket: là điểm cuối (end point) của một liên kết truyền thông 2 chiều giữa 2 chương trình chạy trên môi trường mạng internet.
- ❑ Socket: “cửa” nằm giữa process ứng dụng và end-end transport protocol (TCP, UDP)



Khái niệm socket

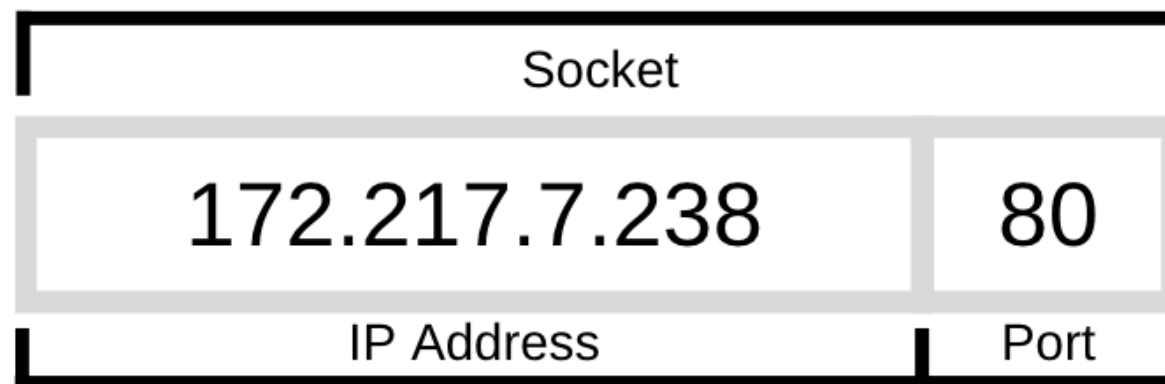
- ❑ **Góc độ mạng:** Socket là 1 trong 2 điểm cuối của đường nối kết 2 chiều giữa 2 chương trình thực thi trên mạng
- ❑ **Góc độ người lập trình:** Socket là giao diện lập trình ứng dụng (API) hay bộ thư viện hàm hỗ trợ, dùng để nối kết chương trình ứng dụng với lớp mạng trong hệ thống mạng TCP/IP.

Khái niệm socket

- ❑ Mỗi tiến trình khi muốn truyền thông bằng socket phải tạo ra một socket và socket đó phải được gán một định danh duy nhất được gọi là địa chỉ socket.

$$\text{Socket} = \text{IP} + \text{Port}$$

=> Địa chỉ socket xác định một đầu mút cuối truyền thông. Nó chỉ ra tiến trình truyền thông nào(port) và chạy trên trên máy nào(IP) sẽ thực hiện truyền thông.

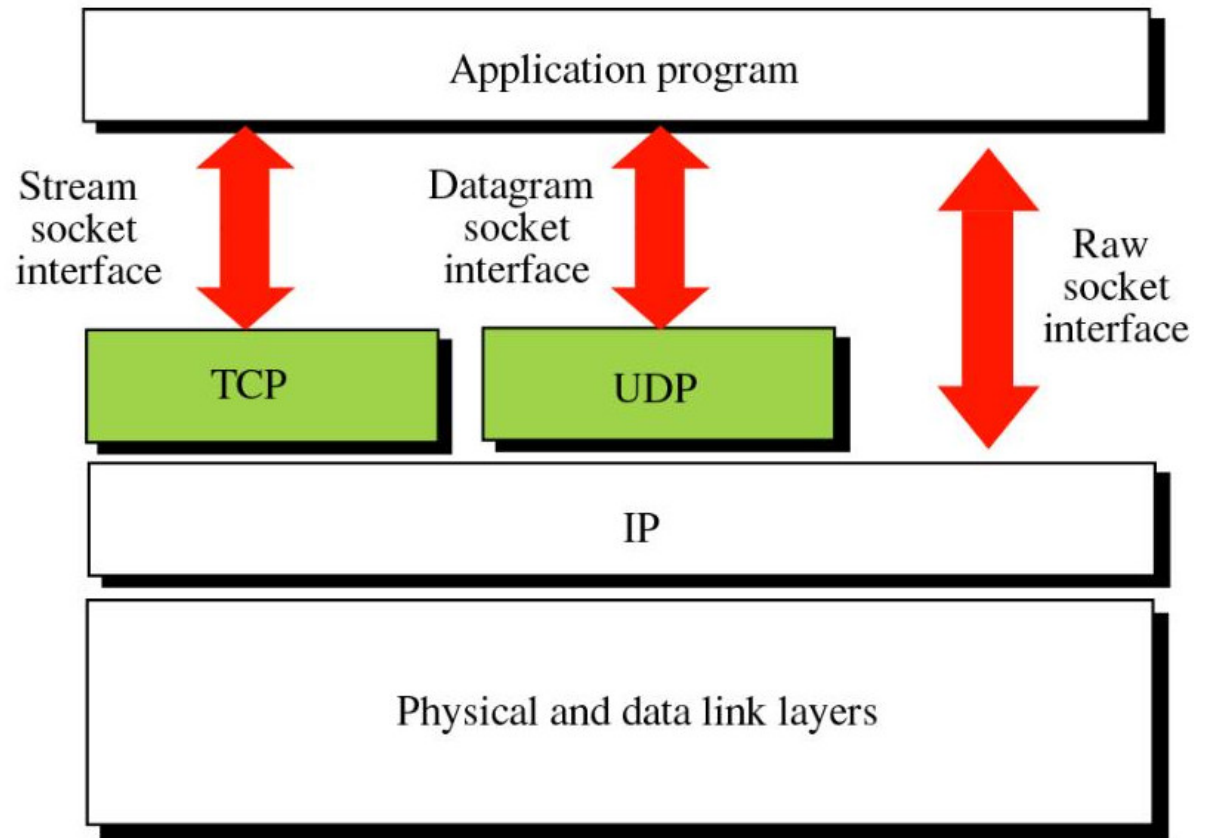


Nội dung

- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
- ❑ Bài tập

Phân loại socket

- ❑ Stream socket
- ❑ Datagram socket
- ❑ Raw socket: cho phép truyền thông đến các giao thức ở tầng mạng thấp hơn cả tầng transport (VD: giao thức ICMP của tầng Internet)



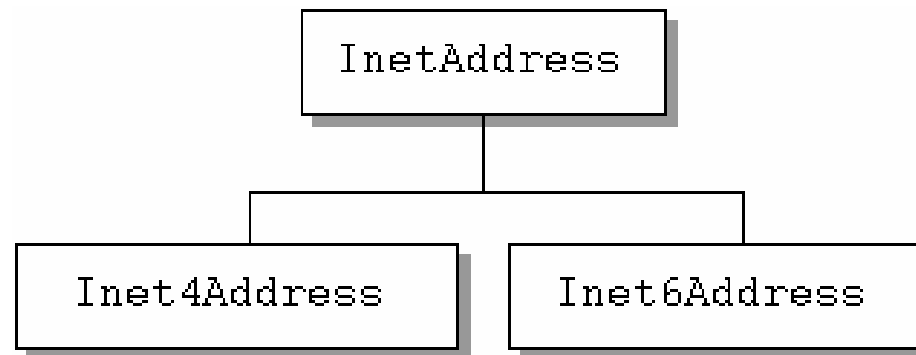
Nội dung

- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
- ❑ Bài tập

Lập trình thao tác với địa chỉ IP

❑ Lớp InetAddress

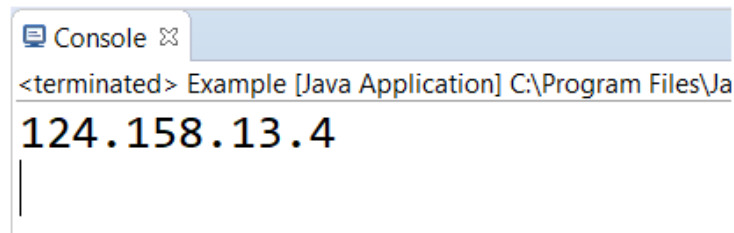
- Lớp mô tả về địa chỉ IP
- Không có constructor → không tạo đối tượng InetAddress bằng toán tử new
- Phương thức `getLocalHost`, `getByName`, hay `getAllByName` để tạo một InetAddress instance
 - ✓ *`public static InetAddress InetAddress.getByName(String hostname)`*
 - ✓ *`public static InetAddress [] InetAddress.getAllByName(String hostname)`*
 - ✓ *`public static InetAddress InetAddress.getLocalHost()`*



Lập trình thao tác với địa chỉ IP

❑ Lớp InetAddress

```
try {  
    InetAddress add = InetAddress.getByName("sgu.edu.vn");  
    System.out.println(add.getHostAddress());  
} catch (UnknownHostException e) {  
    System.out.println("Could not find sgu.edu.vn");  
}
```



Lấy địa chỉ IP tương ứng với domain sgu.edu.vn

Lập trình thao tác với địa chỉ IP

❑ Lớp InetAddress

```
try {  
    InetAddress[] add = InetAddress.getAllByName("google.com");  
    for(int i=0; i<add.length; i++)  
        System.out.println(add[i]);  
} catch (UnknownHostException e) {  
    System.out.println("Could not find sgu.edu.vn");  
}
```



The screenshot shows a console window titled "Console" with a tab icon. The output text is as follows:

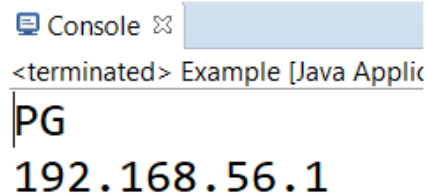
```
<terminated> Example [Java Application] C:\Program Files\J  
google.com/74.125.24.102  
google.com/74.125.24.100  
google.com/74.125.24.113  
google.com/74.125.24.101  
google.com/74.125.24.139  
google.com/74.125.24.138
```

Lấy tất cả địa chỉ IP tương ứng với domain google.com

Lập trình thao tác với địa chỉ IP

❑ Lớp InetAddress

```
try {  
    InetAddress myAdd = InetAddress.getLocalHost();  
    System.out.println(myAdd.getHostName());  
    System.out.println(myAdd.getHostAddress());  
} catch (UnknownHostException e) {  
    System.out.println("Could not find localhost");  
}
```



Console ✖

<terminated> Example [Java Applic

PG

192.168.56.1

Lấy tất cả địa chỉ IP tương ứng với domain google.com

Lập trình thao tác với địa chỉ IP

- ❑ Bài 1: Viết chương trình nhận vào một domain và trả về IP tương ứng với domain đó hoặc thông báo lỗi nếu domain không tồn tại. Chương trình kết thúc khi người dùng nhập từ khóa exit.
- ❑ Bài 2: Phát triển chương trình ở bài 1, đọc danh sách các domain từ 1 file text và xuất kết quả ra console.
- ❑ Bài 3: Viết chương trình đọc danh sách IP từ 1 file text và cho biết những IP nào kết nối/không kết nối được.

Nội dung

- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
 - Chương trình phía Client
 - Chương trình phía Server
- ❑ Bài tập

Nội dung

- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
 - Chương trình phía Client
 - Chương trình phía Server
- ❑ Bài tập

Lập trình TCPSocket

❑ Lớp Socket: chương trình phía client

- Dùng để tạo đối tượng socket cho phép truyền thông với giao thức TCP hoặc UDP. (giao thức UDP thường sử dụng lớp DatagramSocket thay vì lớp Socket)
- Constructor:

✓ `public Socket(String host, int port) throws UnknownHostException, IOException`

```
try {  
    Socket s = new Socket("sgu.edu.vn", 443);  
    if(s.isConnected())  
        System.out.println("Connected");  
} catch (UnknownHostException e) {  
    System.err.println(e);  
} catch (IOException e) {  
    System.err.println(e);  
}
```

Lập trình TCPSocket

❑ Lớp Socket:

- Dùng để tạo đối tượng socket cho phép truyền thông với giao thức TCP hoặc UDP. (giao thức UDP thường sử dụng lớp DatagramSocket thay vì lớp Socket)
- Constructor:
 - ✓ *public Socket(String host, int port) throws UnknownHostException, IOException*
 - ✓ *public Socket(InetAddress host, int port) throws IOException*
 - ✓ *public Socket(String host, int port, InetAddress interface, int localPort) throws IOException, UnknownHostException*
 - ✓ *public Socket(InetAddress host, int port, InetAddress interface, int localPort) throws IOException*

Lập trình TCPSocket

❑ Lớp Socket:

- *public InetAddress getInetAddress()*: trả về địa chỉ mà socket kết nối đến
- *public int getPort()*: port number trên máy trạm từ xa đang kết nối với socket.
- *public int getLocalPort()*: port number trên máy cục bộ
- *public InputStream getInputStream() throws IOException*: trả về luồng nhập của socket là đối tượng InputStream cho việc đọc byte từ socket này
- *public OutputStream getOutputStream() throws IOException*: trả về luồng xuất của socket là đối tượng OutputStream.
- *public void close() throws IOException*: Đóng socket

Lập trình TCPSocket

❑ Lớp Socket:

```
try {
    Socket s = new Socket("sgu.edu.vn", 443);
    System.out.println("Connected to " + s.getInetAddress() +
        " on port " + s.getPort() + " from port " + s.getLocalPort() +
        " of " + s.getLocalAddress());
    s.close();
} catch (UnknownHostException e) {
    System.err.println(e);
} catch (IOException e) {
    System.err.println(e);
}
```

Connected to sgu.edu.vn/124.158.13.4 on port 443 from port 58619 of /192.168.43.114

Lập trình TCPSocket

- ❑ Ví dụ 1: chương trình lấy ngày giờ hiện tại từ time.nist.gov

```
10 public static void main(String[] args) {
11     String hostname = "time.nist.gov";
12     int port = 13;
13     BufferedReader in;
14     int c;
15     try (Socket s = new Socket(hostname, port)) {
16         in = new BufferedReader(new InputStreamReader(s.getInputStream()));
17         StringBuilder data = new StringBuilder();
18         while ((c=in.read())!=-1)
19             data.append((char) c);
20         System.out.println(data);
21         in.close();
22         s.close();
23     } catch (IOException e) {
24         System.err.println(e);
25     }
26 }
```

```
59113 20-09-21 04:32:30 50 0 0 623.3 UTC(NIST) *
```

Lập trình TCPSocket

❑ Ví dụ 2: chương trình WHOIS domain name

```
12 public static void main(String[] args) {
13     String hostname = "whois.internic.net";
14     int port = 43;
15     try (Socket s = new Socket(hostname, port)){
16         BufferedReader in = new BufferedReader(new InputStreamReader(s.getInputStream()));
17         BufferedWriter out = new BufferedWriter(new OutputStreamWriter(s.getOutputStream()));
18         out.write("baomoi.com");
19         out.newLine();
20         out.flush();
21         // Response
22         String line;
23         while ((line = in.readLine()) != null) {
24             System.out.println(line);
25         }
26     } catch (IOException e) {
27         System.err.println(e);
28     }
29 }
```

```
Domain Name: BAOMOI.COM
Registry Domain ID: 148277349_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.namecheap.com
```

Lập trình TCPSocket

❑ Ví dụ 3: chương trình mô phỏng HTTPClient

```
12 public static void main(String[] args) {
13     String hostname = "thongtindaotao.sgu.edu.vn";
14     int port = 80;
15     try (Socket s = new Socket(hostname, port)){
16         BufferedReader in = new BufferedReader(new InputStreamReader(s.getInputStream()));
17         BufferedWriter out = new BufferedWriter(new OutputStreamWriter(s.getOutputStream()));
18         // send an HTTP GET method to the web server
19         String path = "/Default.aspx?page=thoikhoabieu&sta=1&id=10987";
20         out.write("GET " + path + " HTTP/1.1\r\n");
21         out.write("Host: " + hostname + "\r\n");
22         out.write("User-Agent: DemoData\r\n");
23         out.write("Accept: text/html\r\n");
24         out.write("Accept-Language: en-US\r\n");
25         out.write("Connection: Close\r\n");
26         out.newLine();
27         out.flush();
28         // response
29         String resp;
30         while((resp=in.readLine())!=null)
31             System.out.println(resp);
32     } catch (IOException e) {
33         System.err.println(e);
34     }
35 }
```


Lập trình TCPSocket

❑ Ví dụ 4: chương trình scan port

```
8 public static void main(String[] args) {
9     String hostname = "thongtindaotao.sgu.edu.vn";
10    int beginPort = 400;
11    int endPort = 500;
12    int timeout = 200;
13    for(int port=beginPort; port<=endPort; port++) {
14        try {
15            Socket socket = new Socket();
16            socket.connect(new InetSocketAddress(hostname, port), timeout);
17            System.out.println("# Port " + port + " is opened");
18            socket.close();
19        } catch (IOException e) {
20            System.out.println("Port " + port + " is closed");
21        }
22    }
23 }
```


Nội dung

- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
 - Chương trình phía Client
 - Chương trình phía Server
- ❑ Bài tập

Lập trình TCPSocket

❑ Lớp ServerSocket: chương trình phía server

- Cho phép tạo đối tượng socket phía server và truyền thông với giao thức TCP.
- Đối tượng sau khi tạo được đặt ở trạng thái lắng nghe (thụ động) chờ tín hiệu kết nối gửi từ client.
- Constructor:
 - ✓ `public ServerSocket(int port) throws BindException, IOException`
 - Tạo ra đối tượng ServerSocket với số cổng xác định được chỉ ra bởi tham số port.
 - Port=0: cho phép sử dụng một số cổng cho phép nào đó (anonymous port).
 - Trả về ngoại lệ khi socket không thể tạo ra được.
 - Cho phép đáp ứng cực đại tới 50 kết nối đồng thời.

Lập trình TCPSocket

❑ Lớp ServerSocket:

■ Constructor:

✓ `public ServerSocket(int port, int queueLength) throws IOException, BindException`

➤ Chỉ ra số kết nối cực đại mà socket có thể đáp ứng đồng thời bởi tham số queueLength

✓ `public ServerSocket() throws IOException`

➤ Tạo đối tượng ServerSocket nhưng không gắn kết thực sự socket với một port cụ thể.

➤ Không thể chấp nhận bất cứ kết nối nào gửi tới nếu chưa bind().

```
ServerSocket ss = new ServerSocket();  
SocketAddress http = new InetSocketAddress(80);  
ss.bind(http);
```

Lập trình TCPSocket

❑ Lớp ServerSocket:

■ Phương thức accept()

- ✓ **public Socket accept() throws IOException**
- ✓ Đặt đối tượng ServerSocket ở trạng thái “nghe” tại số cổng xác định chờ tín hiệu kết nối gửi đến từ client.
- ✓ Khi có tín hiệu kết nối gửi tới phương thức sẽ trả về đối tượng Socket mới để phục vụ kết nối đó.
- ✓ Khi xảy ra lỗi nhập/xuất, phương thức sẽ ném trả về ngoại lệ IOException

```
ServerSocket server = new ServerSocket(5678);  
while (true) {  
    Socket conn = server.accept();  
    OutputStreamWriter out = new OutputStreamWriter(conn.getOutputStream( ));  
    out.write("You've connected to this server. Bye-bye now.\r\n");  
    conn.close( );  
}
```

Lập trình TCPSocket

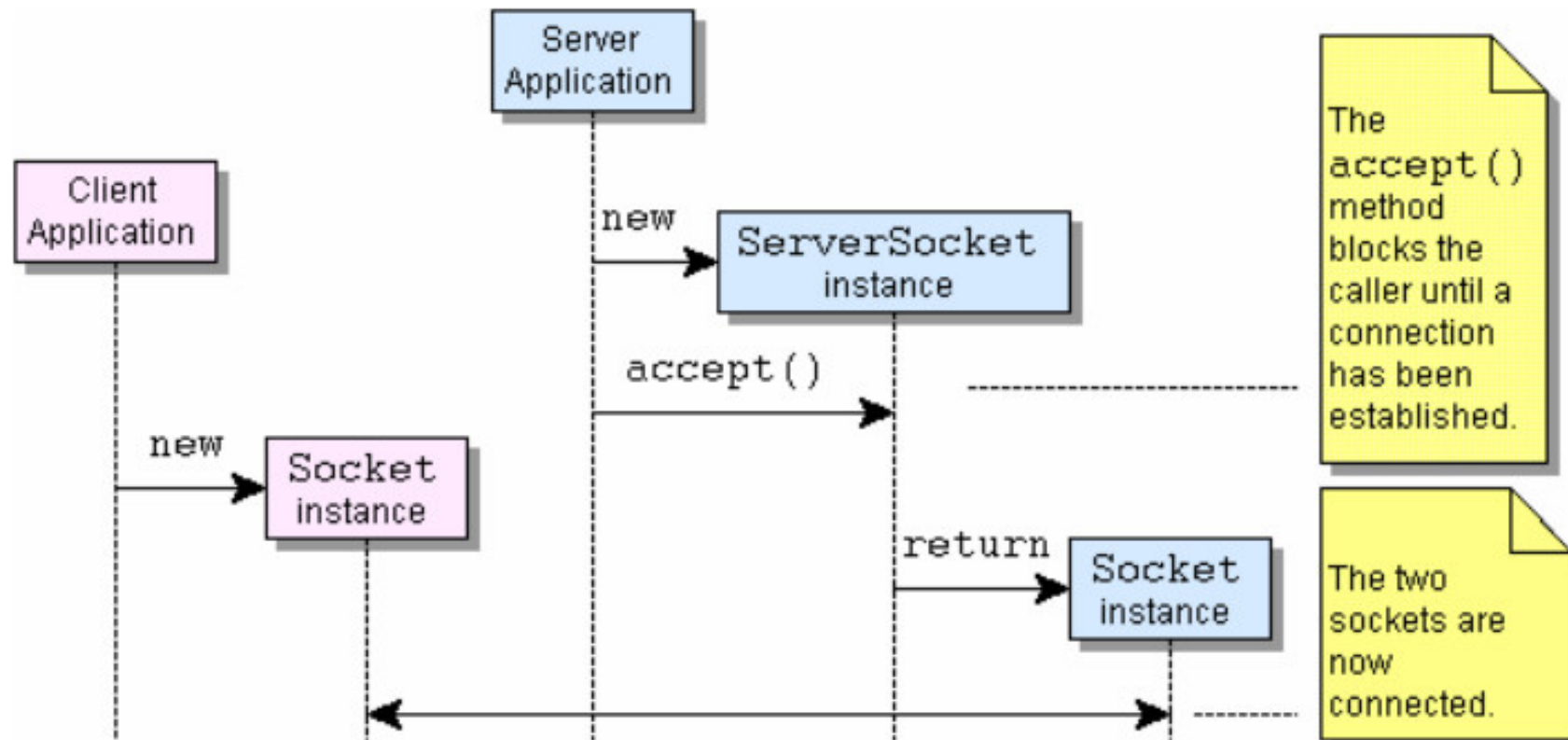
❑ Lớp ServerSocket:

■ Phương thức close()

- ✓ `public void close() throws IOException`
- ✓ Đóng socket và giải phóng tài nguyên.

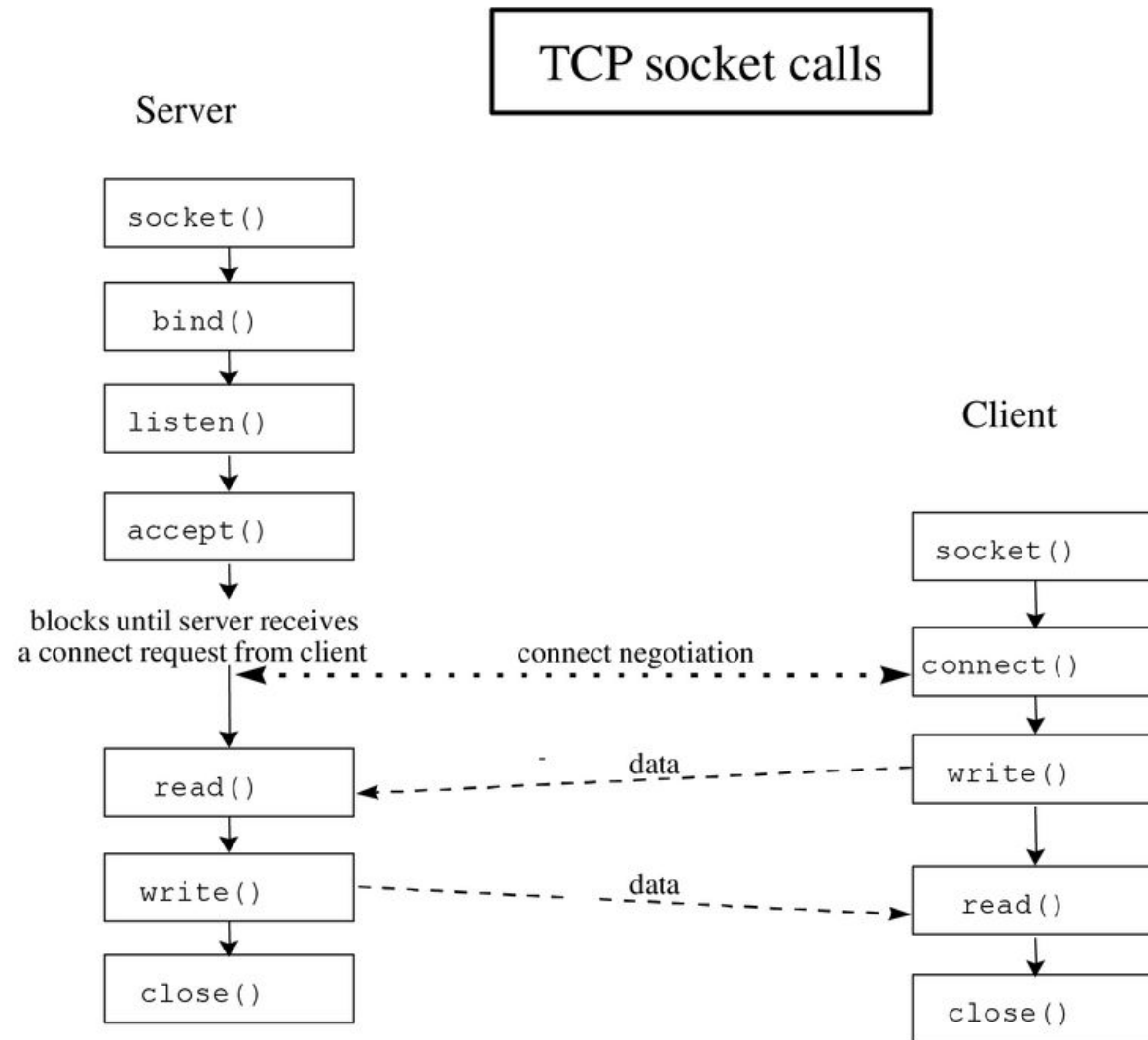
Lập trình TCPSocket

❑ Lập trình với TCP:



Lập trình TCPSocket

❑ Lập trình với TCP:



Lập trình TCPSocket

❑ Lập trình với TCP: chương trình phía server

- Tạo đối tượng server socket và gán port number
- Lắng nghe yêu cầu kết nối bằng phương thức `accept()`, với mỗi yêu cầu được chấp thuận, tạo ra đối tượng socket mới để phục vụ:
 - ✓ Lấy `InputStream` và `OutputStream` gắn với socket kênh ảo vừa hình thành.
 - ✓ Lặp lại công việc sau:
 - Chờ nhận các yêu cầu
 - Phân tích & thực hiện yêu cầu
 - Tạo thông điệp trả lời
 - Gửi thông điệp trả lời → client
 - Không còn yêu cầu hoặc client kết thúc → đóng socket và quay lại lắng nghe yêu cầu kết nối

Lập trình TCPSocket

❑ Lập trình với TCP: chương trình phía client

- Tạo đối tượng socket và thiết lập kết nối đến server (phải chỉ rõ tham số server)
- Khai báo luồng nhập/xuất (kiểu byte hoặc char)
- Truyền thông qua luồng nhập/xuất đã khai báo
- Đóng socket và giải phóng tài nguyên khi không cần.

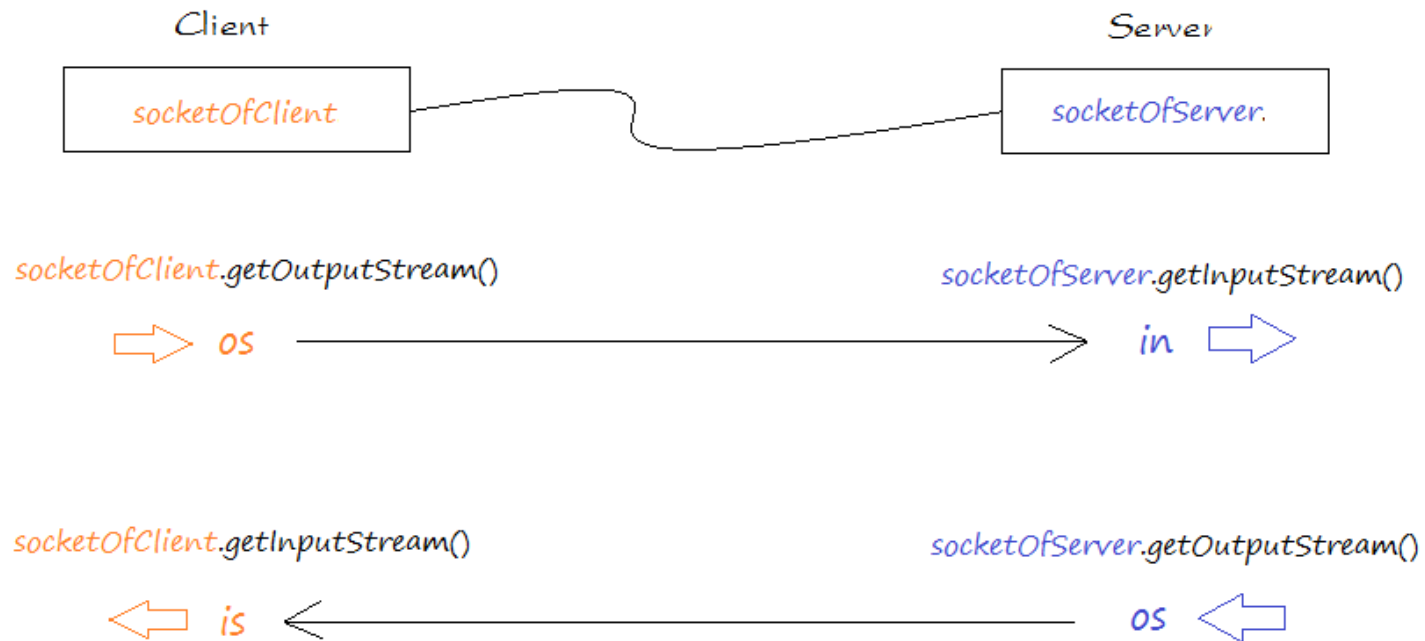
❑ Lưu ý:

- Chương trình phía server luôn chạy trước client
- Chương trình server có thể phục vụ nhiều client đồng thời hoặc lặp.

Lập trình TCPSocket

- ❑ Ví dụ minh họa: chương trình gửi dữ liệu qua lại giữa client và server.
 - Client gửi một chuỗi dữ liệu → server
 - Server nhận và in chuỗi ra màn hình
 - Chương trình lặp lại liên tục cho đến khi client gửi Over thì server đóng socket
- ❑ Phần code phía server:

Lập trình TCPSocket



Lập trình TCPSocket

❑ Code chương trình server

```
1 // A Java program for a Server
2 import java.net.*;
3 import java.io.*;
4
5 public class Server
6 {
7     private Socket      socket    = null;
8     private ServerSocket server    = null;
9     BufferedWriter out = null;
10    BufferedReader in = null;
11
12    public Server(int port)
13    {
14        try
15        {
16            server = new ServerSocket(port);
17            System.out.println("Server started");
18            System.out.println("Waiting for a client ...");
19            socket = server.accept();
20            System.out.println("Client accepted");
21            out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
22            in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
23            String line = "";
```

Lập trình TCPSocket

```
24 while (!line.equals("Over"))
25 {
26     try
27     {
28         line = in.readLine();
29         System.out.println("Server received: " + line);
30     }
31     catch(IOException i)
32     {
33         System.err.println(i);
34     }
35 }
36 System.out.println("Closing connection");
37
38 in.close();
39 out.close();
40 socket.close();
41 server.close();
42 }
43 catch(IOException i)
44 {
45     System.out.println(i);
46 }
47 }
```

```
49 public static void main(String args[])
50 {
51     Server server = new Server(5000);
52 }
53 }
```

Lập trình TCPSocket

❑ Phần code phía client:

```
1 // A Java program for a Client
2 import java.net.*;
3 import java.io.*;
4
5 public class Client
6 {
7     private Socket socket          = null;
8     BufferedWriter out = null;
9     BufferedReader in = null;
10    BufferedReader stdIn = null;
11
12    public Client(String address, int port) throws UnknownHostException, IOException
13    {
14
15        socket = new Socket(address, port);
16        System.out.println("Connected");
17        out = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
18        in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
19        stdIn = new BufferedReader(new InputStreamReader(System.in));
20
21        String line = "";
```

Lập trình TCPSocket

```
22
23     while (!line.equals("Over"))
24     {
25
26         line = stdIn.readLine();
27         System.out.println("Client sent: " + line);
28         out.write(line);
29         out.newLine();
30         out.flush();
31     }
32
33     in.close();
34     out.close();
35     socket.close();
36
37 }
38
39 public static void main(String args[]) throws UnknownHostException, IOException
40 {
41     Client client = new Client("127.0.0.1", 5000);
42 }
43 }
```

Lập trình TCPSocket

❑ Lưu ý khi chạy chương trình:

- Server cần khởi động trước
- Nên phân chia công việc → method, không nên gom trong Main hay constructor
- Lỗi Address already in use xảy ra do port muốn sử dụng bị chiếm, cần kill tiến trình đang chiếm port hoặc sử dụng port khác.
- Lệnh netstat cho biết thông tin về kết nối trên máy tính

```
C:\Users\Giang Nguyen>netstat -ano | findstr 5000
```

TCP	0.0.0.0:5000	0.0.0.0:0	LISTENING	17912
TCP	127.0.0.1:50000	0.0.0.0:0	LISTENING	19252
TCP	127.0.0.1:50000	127.0.0.1:59483	ESTABLISHED	19252
TCP	127.0.0.1:59483	127.0.0.1:50000	ESTABLISHED	16656
TCP	:::5000	:::0	LISTENING	17912

Nội dung

- ❑ Khái niệm socket
- ❑ Phân loại socket
- ❑ Lập trình với địa chỉ IP (InetAddress)
- ❑ Lập trình TCPSocket
- ❑ Bài tập

Bài tập

- ❑ Bài 1: Viết chương trình gửi tin nhắn qua lại giữa Client-Server:
 - Client gửi một chuỗi ký tự bất kỳ đến server
 - Server nhận và gửi chuỗi đảo ngược về client
 - Client xuất kết quả ra console, chương trình kết thúc khi client gửi chuỗi **bye**.
- ❑ Bài 2: Viết chương trình tìm số hoàn hảo theo mô hình Client-Server
 - Client gửi 1 số n nguyên dương đến Server
 - Server kiểm tra n , nếu:
 - ✓ Là số hoàn hảo: trả kết quả về client và xuất ra màn hình
 - ✓ Không là số hoàn hảo: trả về client số hoàn hảo lớn hơn và gần n nhất.

Bài tập

- ❑ Bài 3: Viết chương trình phân tích số hoạt động theo mô hình Client – Server:
 - Client gửi số nguyên dương $n \geq 10$ đến server.
 - Server phân tích n thành tích các số nguyên tố và gửi trả ngược lại client
 - Client xuất kết quả ra console
- ❑ Bài 4: Viết chương trình đoán số
 - Khi client kết nối, server tạo sẵn 1 số nguyên ngẫu nhiên $n \leq 100$
 - Client đoán số do server tạo, nếu không đúng, server cần gợi ý bằng cách cho biết số client gửi lớn hơn hay nhỏ hơn n .
 - Quá trình lặp liên tục cho tới khi client gửi đúng số $= n$. Server xuất các thống kê: số lần client đoán, tổng thời gian đoán.

Tài liệu tham khảo

- ❑ Bài giảng Lập trình mạng – Phạm Trần Vũ – ĐH Bách Khoa TP.HCM
- ❑ Giáo trình Lập trình mạng – Hà Mạnh Đào – PTIT
- ❑ <https://www.codejava.net/>
- ❑ <https://www.geeksforgeeks.org/socket-in-computer-network/>
- ❑ <https://o7planning.org/vi/10393/huong-dan-lap-trinh-java-socket>
- ❑ <https://www.baeldung.com/a-guide-to-java-sockets>
- ❑ <https://o7planning.org/vi/10173/huong-dan-su-dung-luong-vao-ra-ky-tu-trong-java>