Java 核心技术(进阶)

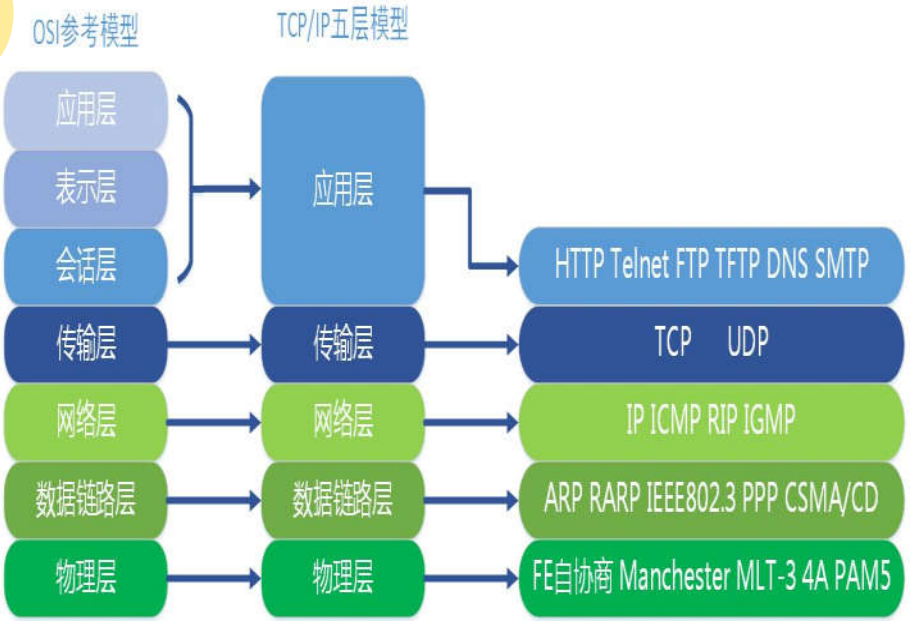
第六章 Java网络编程

第三节 Java TCP 编程

华东师范大学 陈良育

网络通讯协议

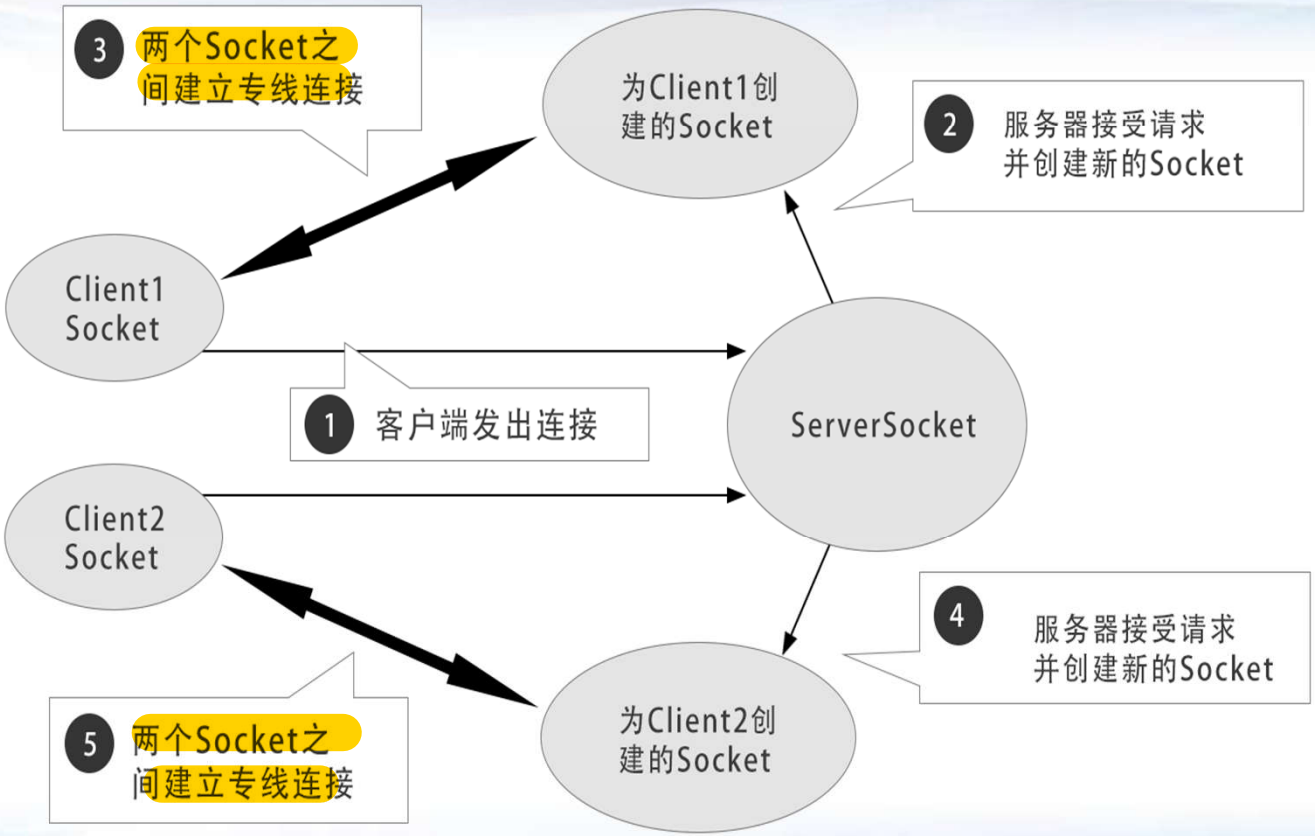
- 通讯协议：TCP和UDP
- TCP：Transmission Control Protocol
  - 传输控制协议，面向连接的协议
  - 两台机器的可靠无差错的数据传输
  - 双向字节流传递
- UDP: User Datagram Protocol
  - 用户数据报协议，面向无连接协议
  - 不保证可靠的数据传输
  - 速度快，也可以在较差网络下使用



TCP(1)

- TCP协议：有链接、保证可靠的无误差通讯
  - ①服务器：创建一个ServerSocket，等待连接
  - ②客户机：创建一个Socket，连接到服务器
  - ③服务器：ServerSocket接收到连接，创建一个Socket和客户的Socket建立专线连接，后续服务器和客户机的对话(这一对Socket)会在一个单独的线程（服务器端）上运行
  - ④服务器的ServerSocket继续等待连接，返回①

TCP(2)


3 两个Socket之间建立专线连接
为Client1创建的Socket
2 服务器接受请求并创建新的Socket
Client1 Socket
1 客户端发出连接
ServerSocket
Client2 Socket
4 服务器接受请求并创建新的Socket
为Client2创建的Socket
5 两个Socket之间建立专线连接

TCP(3)

- ServerSocket: 服务器码头
  - 需要绑定port
  - 如果有多块网卡，需要绑定一个IP地址
- Socket: 运输通道
  - 客户端需要绑定服务器的地址和Port
  - 客户端往Socket输入流写入数据，送到服务端
  - 客户端从Socket输出流取服务器端过来的数据
  - 服务端反之亦然

TCP(4)

- 服务端等待响应时，处于阻塞状态
- 服务端可以同时响应多个客户端
- 服务端每接受一个客户端，就启动一个独立的线程与之对应
- 客户端或者服务端都可以选择关闭这对Socket的通道
- 实例
  - 服务端先启动，且一直保留
  - 客户端后启动，可以先退出

总结

- 了解TCP基础概念
- 了解Java的TCP编程

代码(1) TcpServer.java


```
public class TcpServer
{
    public static void main(String [] args)
    {
        try
        {
            ServerSocket ss=new ServerSocket(8001); //驻守在8001端口
            Socket s=ss.accept();                   //阻塞，等到有客户端连接上来
            System.out.println("welcome to the java world");
            InputStream ips=s.getInputStream();     //有人连上来，打开输入流
            OutputStream ops=s.getOutputStream();   //打开输出流
            //同一个通道，服务端的输出流就是客户端的输入流；服务端的输入流就是客户端的输出流

            ops.write("Hello, Client!".getBytes());  //输出一句话给客户端
```

代码(2) TcpServer.java


```
            BufferedReader br = new BufferedReader(new InputStreamReader(ips));
            //从客户端读取一句话
            System.out.println("Client said: " + br.readLine());


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

代码(3) TcpClient.java


```
public class TcpClient {
    public static void main(String[] args) {
        try {
            Socket s = new Socket(InetAddress.getByName("127.0.0.1"), 8001); //需要服务端先开启

            //同一个通道，服务端的输出流就是客户端的输入流；服务端的输入流就是客户端的输出流
            InputStream ips = s.getInputStream();     //开启通道的输入流
            BufferedReader brNet = new BufferedReader(new InputStreamReader(ips));

            OutputStream ops = s.getOutputStream();  //开启通道的输出流
            DataOutputStream dos = new DataOutputStream(ops);

            BufferedReader brKey = new BufferedReader(new InputStreamReader(System.in));
```

代码(4) TcpClient.java


```
            while (true)
            {
                String strWord = brKey.readLine();
                if (strWord.equalsIgnoreCase("quit"))
                {
                    break;
                }
                else
                {
                    System.out.println("I want to send: " + strWord);
                    dos.writeBytes(strWord + System.getProperty("line.separator"));

                    System.out.println("Server said: " + brNet.readLine());
                }

            }

            dos.close();
            brNet.close();
            brKey.close();
            s.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

代码(5) TcpServer2.java


```
public class TcpServer2
{
    public static void main(String [] args)
    {
        try
        {
            ServerSocket ss=new ServerSocket(8001);
            while(true)
            {
                Socket s=ss.accept();
                System.out.println("来了一个client");
                new Thread(new Worker(s)).start();
            }
            //ss.close();
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

代码(6) Worker.java


```
class Worker implements Runnable {
    Socket s;

    public Worker(Socket s) {
        this.s = s;
    }

    public void run() {
        try {
            System.out.println("服务人员已经启动");
            InputStream ips = s.getInputStream();
            OutputStream ops = s.getOutputStream();

            BufferedReader br = new BufferedReader(new InputStreamReader(ips));
            DataOutputStream dos = new DataOutputStream(ops);
```

代码(7) Worker.java


```
            while (true) {
                String strWord = br.readLine();
                System.out.println("client said:" + strWord +":" + strWord.length());
                if (strWord.equalsIgnoreCase("quit"))
                    break;
                String strEcho = strWord + " 666";
                // dos.writeBytes(strWord +"---->"+ strEcho +"\r\n");
                System.out.println("server said:" + strWord + "---->" + strEcho);
                dos.writeBytes(strWord + "----> " + strEcho + System.getProperty("line.separator"));
            }
            br.close();
            // 关闭包装类，会自动关闭包装类中所包装的底层类。所以不用调用ips.close()
            dos.close();
            s.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

谢 谢！