Java 网络编程

- What?Why?How?

本章概述

- 基本概念
- 网络分层
- 数据封装拆分
- 网络爬虫原理
- TCP编程
- UDP编程

网络的概念

- 网络：一组相互连接的计算机
  - 多台计算机组成
  - 使用物理线路进行连接

相互连接的计算机

交换数据

共享资源

网络编程的三要素

【1】IP地址:唯一标识网络上的每一台计算机

两台计算机之间通信的必备要素

【2】端口号:计算机中应用的标号(代表一个应用程序)

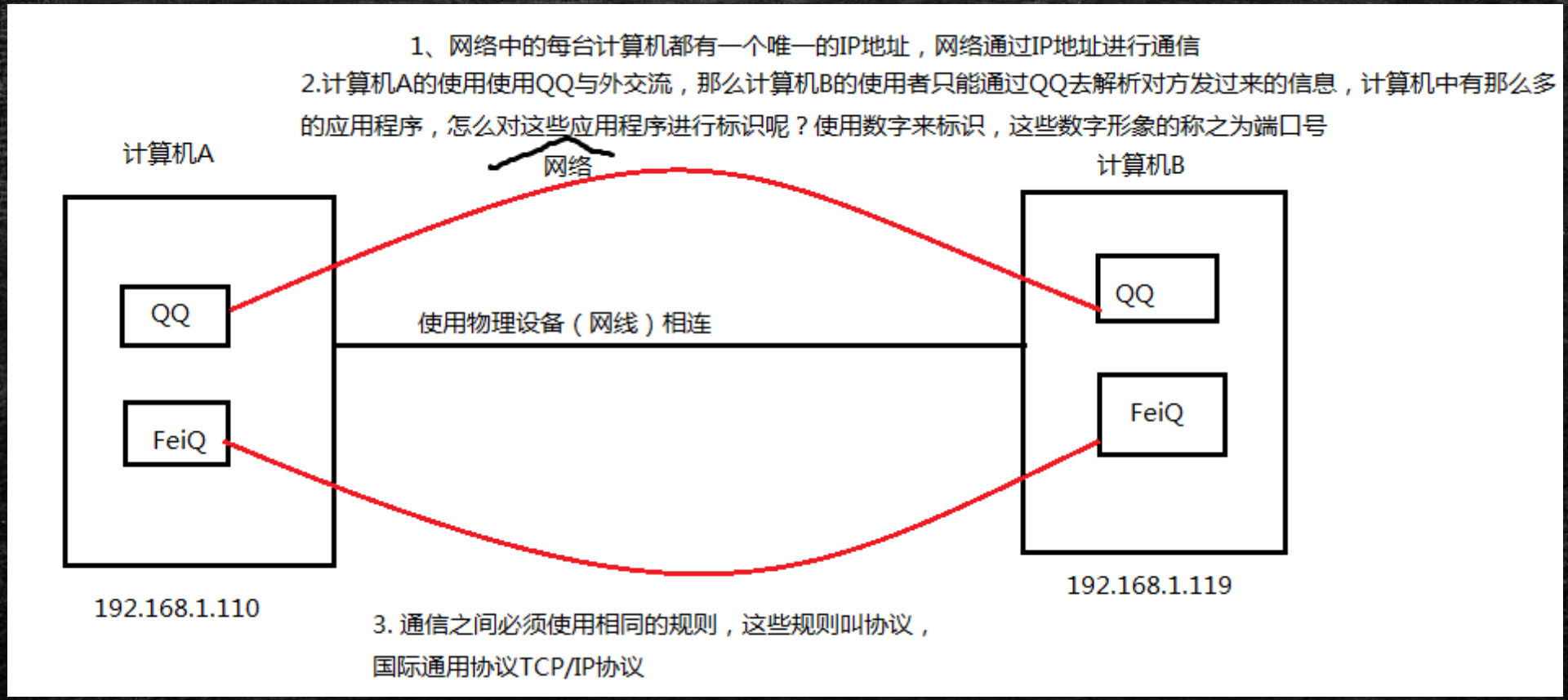
0-1024系统使用或保留端口，

有效端口0-65536

【3】通信协议:通信的规则

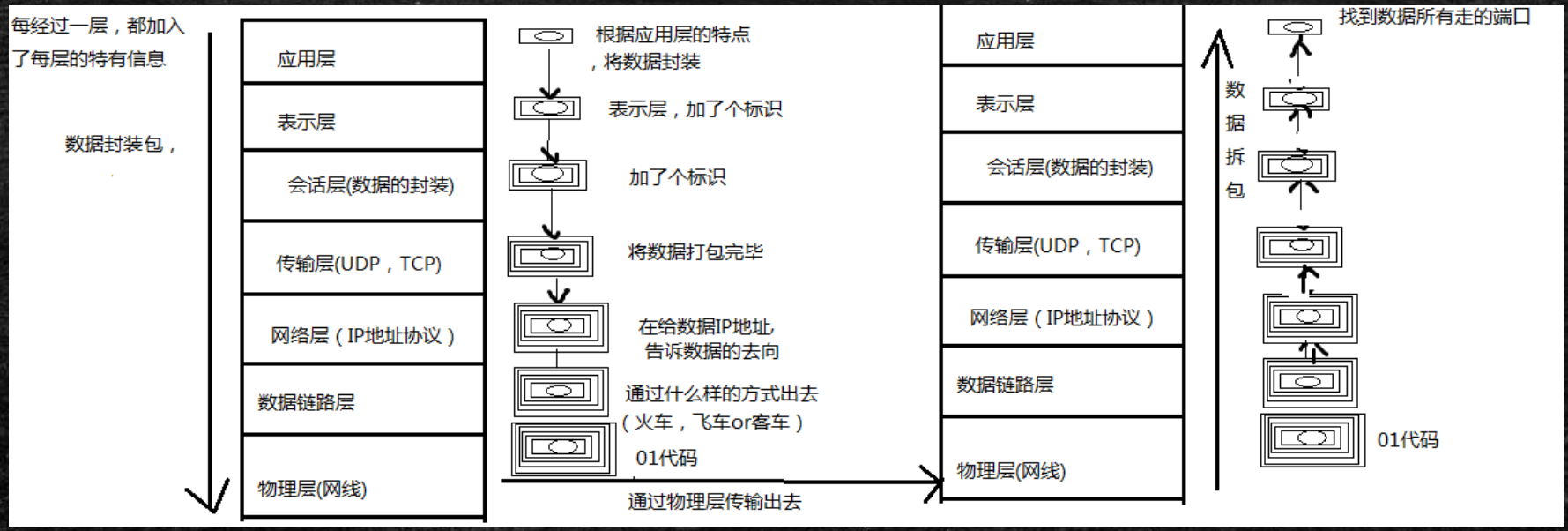
TCP,UDP

网络编程的三要素


1、网络中的每台计算机都有一个唯一的IP地址，网络通过IP地址进行通信
2.计算机A的使用使用QQ与外交流，那么计算机B的使用者只能通过QQ去解析对方发过来的信息，计算机中有那么多的应用程序，怎么对这些应用程序进行标识呢？使用数字来标识，这些数字形象的称之为端口号
计算机A
网络
计算机B
QQ
使用物理设备（网线）相连
QQ
FeiQ
FeiQ
192.168.1.110
192.168.1.119
3. 通信之间必须使用相同的规则，这些规则叫协议，
国际通用协议TCP/IP协议

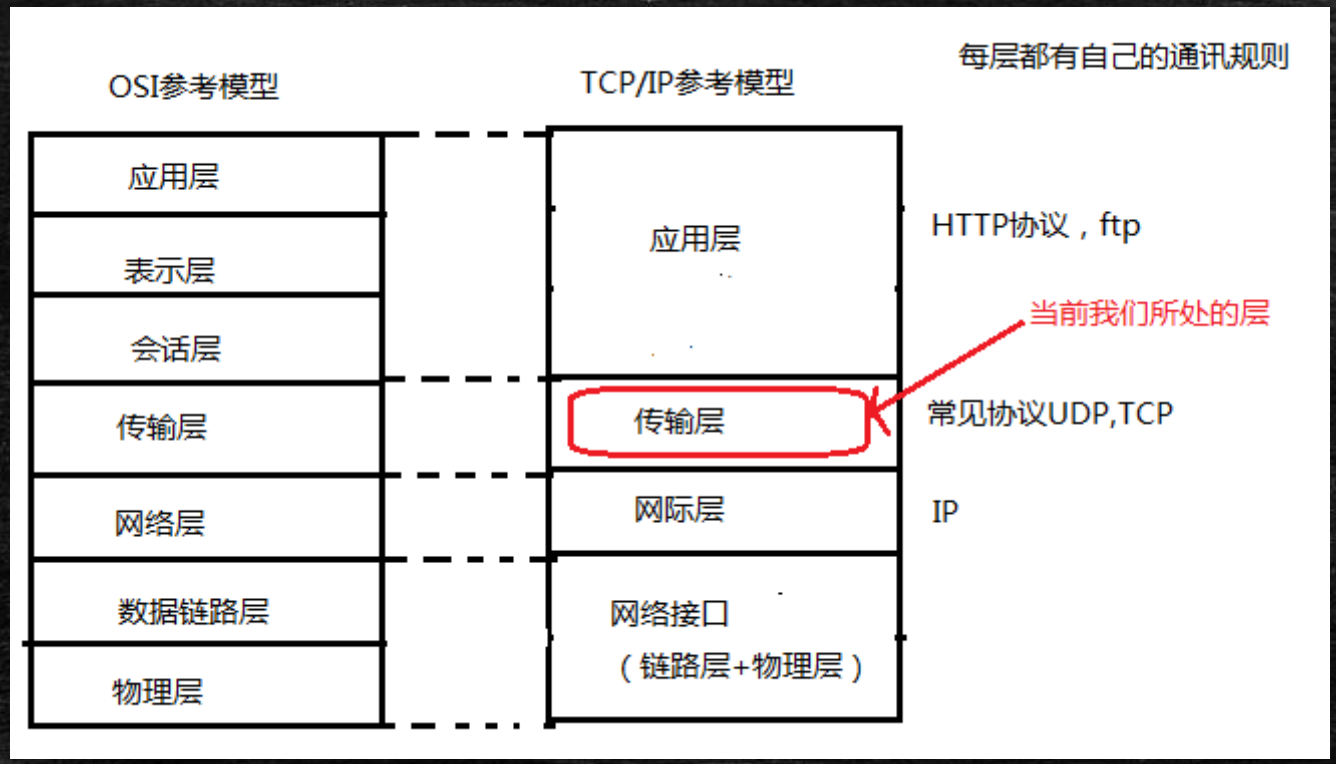
网络模型一

- **OSI参考模式:开放系统互连参考模型（Open System Interconnect）**



网络模型二

- **TCP/IP参考模型:传输控制/网际协议 Transfer Controln Protocol/Internet Protocol**



IP地址的表示方法

- IP 地址：32位，由4个8位二进制数组成
- IP表示方法：点分十进制

192.168.1.200 = 11000000.10101000.00000001.11001000

十进制表示　　二进制表示

- IP地址 = 网络ID +主机ID
  - 网络ID：标识计算机或网络设备所在的网段
  - 主机ID：标识特定主机或网络设备

IP地址的分类

- 地址类用于指定网络 ID 并在网络 ID 和主机 ID 之间提供分隔方法
- IANA负责分配A、B、C类网络地址，具体主机地址由机构组织自行分配
- IP 地址类包括：
  - A 类：
  - B 类：
  - C 类：
  - D 类：
  - E 类：

特殊的IP地址

- 0.0.0.0：本机
- 127.0.0.1：本机回环地址，用于本机测试
- 255.255.255.255：当前子网，一般用于向当前子网广播信息

IP地址所对应的对象→InetAddress

java.net
类 InetAddress

java.lang.Object
  └ java.net.InetAddress

所有已实现的接口：
Serializable

直接已知子类：
Inet4Address, Inet6Address

---

public class **InetAddress**
extends Object
implements Serializable

此类表示互联网协议（IP）地址。

IP地址所对应的对象→InetAddress

序号	方法	描述
1	public static InetAddress **getLocalHost**()	获得主机名和IP地址
2	public String **getHostAddress**()	获取IP地址
3	public String **getHostName**()	获取主机名
4	public static InetAddress **getByName**(String host)	根据主机名获得IP地址

获得百度的主机名

```
InetAddress ia2=InetAddress.getByName("www.baidu.com");
System.out.println("其它主机名称:"+ia2.getHostAddress());
```

注意事项：有可能返回的主机ip有很多,只是显示了中的一个

端口

- 端口:port

  端口是虚拟的概念，并不是说在主机上真的有若干个端口。通过端口，可以在一个主机上运行多个网络应用程序。

传输协议

UDP:相当于发短信（有字数限制），

不需要建立连接，

数据报的大小限制在64k内，

效率较高,不安全，容易丢包

TCP:相当于打电话，需要建立连接，

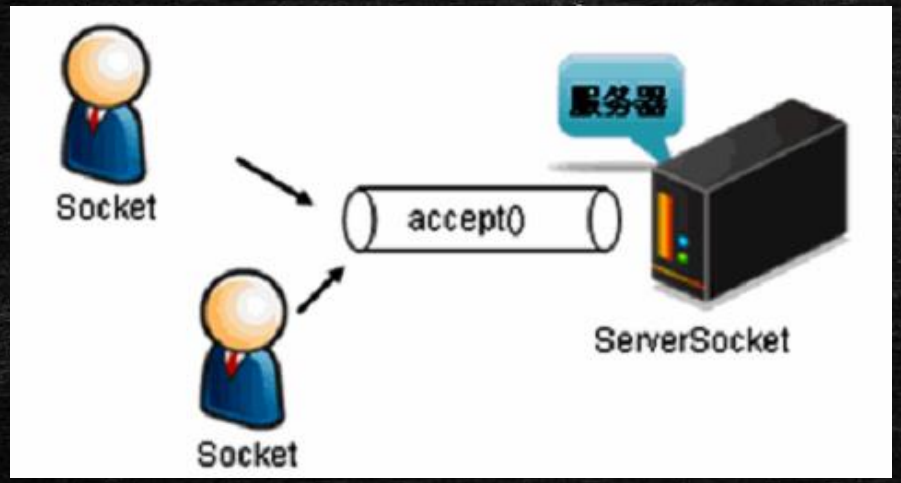
效率相对比较低，数据传输安全，

三次握手完成。

(点名→答到→确认)

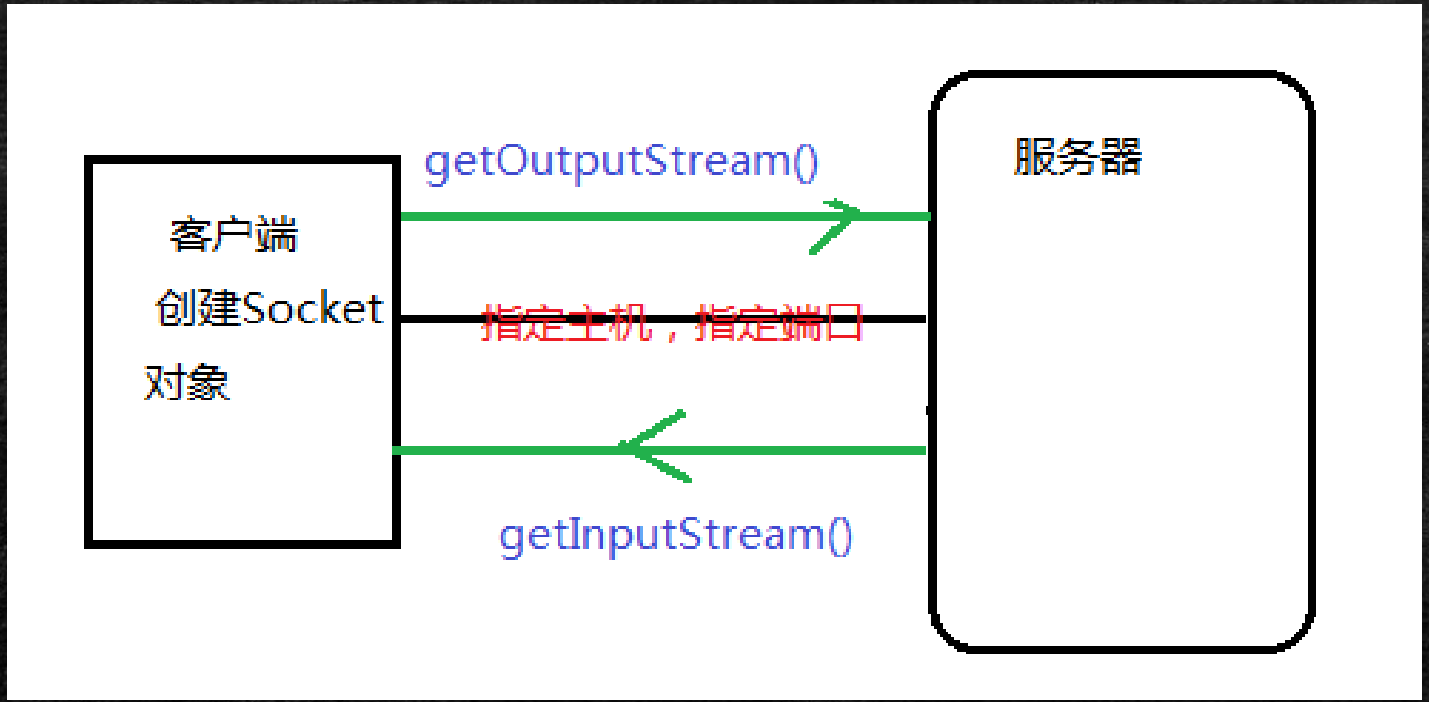
Socket套接字

- 网络上的两个程序通过一个双向的通信连接实现数据的交换，
- 这个连接的一端称为一个socket。
- Java中使用Socket完成TCP程序的开发，使用此类可以方便的建立**可靠的、双向的、持续性的、点对点**的通讯连接
- 在Socket的程序开发中，服务器端使用ServerSocket等待客户端的连接，
- 对于java的网络程序来讲，每一个客户端都使用一个Socket对象表示



基于TCP协议的Socket编程

- 进行网络通信时，Socket需要借助数据流来完成数据的传递工作



基于TCP协议的Socket编程

客户端

建立连接

打开Socket关联的输入输出流

数据流中读写信息

关闭所有的数据流和Socket

实现单用户登录

```
Socket socket=new Socket("localhost",8800);
```

```
OutputStream os=socket.getOutputStream();
```

```
String info="用户名：Tom;用户密码：123456";
os.write(info.getBytes());
socket.shutdownOutput();
```

```
os.close();
socket.close();
```

基于TCP协议的Socket编程

服务端

建立连接,处理发送到指定端口的数据

```
ServerSocket server=new ServerSocket(8800);
```

获取客户端对象

```
Socket socket=server.accept();
```

数据流中读写信息

```
InputStream is=socket.getInputStream();
 byte[] buf=new byte[1024];
 int len=is.read(buf);
 syso(new String(buf,0,len))
socket.shutdownInput();
```

关闭所有的数据流和Socket

```
is.close();
socket.close();
server.close();
```

上机练习

- 编程实现客户端与服务器的一次会话

运行效果:

服务器端

```
服务器端已启动.........
127.0.0.1:说你好!
```

客户端

```
服务器端说:收到!
```

上机练习

- 将客户端的图片上传到服务器
- 思路:
- 客户端输出到服务器
- 服务器读取数据写入文件

Socket中实现对象的传递

- 如何传递对象信息呢?

```
String info="用户名：Tom;用户密码：123456";
os.write(info.getBytes());
```

序列化

```
User user=new User();//User是用户类
user.setLoginName("Tom");
user.setPwd("123456");
oos.writeObject(user);
```

上机练习

- 实现用户登录

- **需求说明：**
- 客户端序列化对象
- 服务器端反序列化对象

上机练习

- 多线程实现用户登录

- **需求说明：**
- 在上机练习三的基础上实现

基于UDP的网络编程

基于TCP协议的Socket编程	基于UDP协议的Socket编程
通信双发需要建立连接	通信双发不需要建立连接
连接建立时双方存在主次之分	通信双方完全平等
114查号台	QQ聊天模式

基于UDP的网络编程

咨询者

咨询问题

客服人员

客服人员

答复问题

接收方

发送方