# 网络编程

## 网络编程入门

网络编程:

在网络通信协议下,不同计算机上运行的程序,可以进行数据传输.

### 三要素概述

**IP地址**

设备在网络中的地址,时唯一标识;

**端口**

应用程序在设备中的唯一标识;

**协议**

数据在网络中传输的规则,常见的协议有UDP协议和TCP协议.

### 三要素-IP

IP : 全称互联网协议地址,也称IP地址.是分配给上网设备的数字标签.常见的IP分类为ipv4和ipv6.

Ipv4 : 4字节4\*8=32bit位,点分十进制表示法,如192.168.1.66

Ipv6 : 16字节16\*8 = 128bit位,冒分十六进制表示法,两个字节一组共8组,开头为0可以省略,全部是0都可以省略,如fe80::f52d:8c6:f609:4bca%19

### 常见命令

ipconfig : 查看本机IP地址;

ping IP地址 : 检查网络是否连通;

特殊IP地址 :

127.0.0.1 : 是回送地址也称本机回环地址,可以代表本地的IP地址,一般用来测试使用的

### Inetaddress类

对主机的IP地址等进行获取和操作.

**1.获取**

没有构造方法,使用静态方法获取类的对象 :

Inetaddress.getByName(String host) : 传入主机名或者IP地址;

**2.成员方法**

getHostName() : 获取主机名;

getHostAddress() : 获取主机地址;

### 三要素-端口

端口:应用程序在设备中的唯一标识;

端口号: 用两个字节表示的整数,他的取值范围是0-65535;

其中0-1023之间的端口号用于一些知名的网络服务或者应用;

我们自己是用1024以上的端口号就可以了

注意 : 一个端口号只能被一个应用程序使用.

### 三要素-协议

协议 : 计算机网络中,连接和通信的规则被称为网络通信协议;

UDP协议

1.用户数据报协议

2.UDP是面向无连接通信协议;

特点:速度快,有大小限制一次最多发送64K,数据不安全,易丢失数据;

TCP协议

1.传输控制协议;

2.TCP协议是面向连接的通信协议;

特点:速度慢,没有大小限制,数据安全.

## UDP通信程序

### 发送端

步骤:

1.创建发送端DatagramSocket对象;

DatagramSocket ds = new DatagramSocket();

抛出SocketException,空参:表示随机端口发出

2.创建数据,并把数据打包(DatagramPacket);

DatagramPacket dp = new DatagramPacket(byte[] buf, int length, InetAddress address, int port);

也是创建对象,传入4个参数:

byte[] buf : 要发送的数据;

int length : 要发送的数组的长度;(不一定是整个数组发送)

InetAddress address : 接收端IP地址

int port : 接收端端口号

3.调用DatagramSocket对象的send()方法发送数据;

DatagramSocket对象.send(DatagramPacket对象);

4.释放资源;

DatagramSocket对象.close();

### 接收端

**步骤:**

1.创建接收端的DatagramSocket对象

DatagramSocket ds = new DatagramSocket();

注意:这里的参数是发送端的端口号,发送端发送数据参数里有写明

2.创建一个箱子,用于接收数据

创建一个数组用于接收数据.

再创建DatagramPacket对象.用DatagramPacket的方法接收数据.参数是创建的数组容器以及接受数据的长度.

**示例:**

byte[] bytes = new byte[1024];

DatagramPacket dp = new DatagramPacket(bytes, bytes.length);

3.调用DatagramSocket的方法接收数据,并将数据放入箱子中

DatagramSocket对象.receive(DatagramPacket对象);

**示例:**

ds.receive(dp);

4.解析数据包,并把数据在控制台显示

数据操作,比如将字节数组转换成字符串,增加可读性

**示例:**

int length = dp.getLength();

String s = new String(bytes, 0, length);

System.out.println(s);

5.释放资源

DatagramSocket对象.close();

### 练习

在本机上循环发送接收数据

### 三种通讯方式

单波;

组播;

广播;

### 组播代码实现

在发送端指定组播地址,其他步骤不变

在接收端需要加入发送端指定组播地址的组中,使用 :

**DatagramSocket对象. joinGroup(地址)**

**示例 :**

ms.joinGroup(InetAddress.getByName("225.0.1.0"));

### 广播代码实现

和组播一个不同的地方就是发送端指定的地址为(255.255.255.255),其他不变.

## TCP通信程序

TCP通信协议是一种可靠的网络协议,它在通信的两端各建立一个Socket对象.通信之前要保证连接已经建立.

客户端 : Socket

服务器 : ServerSocket

通信之前保证保证连接已经建立

### 客户端

客户端发送数据的步骤

1.创建客户端Socket对象(Socket)与指定服务端连接;

Socket socket = new Socket("127.0.0.1", 10000);

2.获取输出流,写数据;

OutputStream os = socket.getOutputStream();

os.write("hello".getBytes());

3.释放资源;

os.close();

socket.close();

### 服务器

TCP接收数据

**步骤:**

**1.创建服务器端的Socket对象(ServerSocket)**

ServerSocket(int port)

**示例 :**

ServerSocket ss = new ServerSocket(10000);

**2.监听客户端连接,返回一个Socket对象**

Socket accept()

**示例 :**

Socket accept = ss.accept();

**3.获取输入流,读数据,并把数据显示在控制台**

getInputStream()

**示例 :**

InputStream is = accept.getInputStream();

int len;

while ((len = is.read()) != -1) {

System.out.print((char) len);

}

**4.释放资源**

void close()

### 原理分析

1.accept方法是阻塞的,作用就是等待客户端连接;

2.客户端创建对象并连接服务器,此时是通过三次握手协议保证跟服务器之间的连接;

3.针对客户端来讲,是往外写的,所以是输出流;正对服务器来讲,是往里读的,所以是输入流;

4.read方法也是阻塞的;

5.在关流的时候,还多了一个往服务器写结束标记的动作;

6.最后一步断开连接,通过四次挥手协议保证连接终止.

### 三次握手

### 四次握手

### 练习

## 服务器端优化

### 循环

### UUID

### 多线程

### 线程池