Socket通信

罗伟
苏州市职业大学

1 网络编程基础

- 网络通信协议
- 定义：通过计算机网络可以使多台计算机实现连接，但是位于同一个网络中的计算机在进行连接和通信时必须要遵守一定的规则。在计算机网络中，这些连接和通信的规则被称为网络通信协议。
- 作用：它对数据的传输格式、传输速率、传输步骤等做了统一规定，通信双方必须同时遵守才能完成数据交互。
- 分类：目前应用最广泛的有TCP/IP协议（Transmission Control Protocol/Internet Protocol，传输控制协议/英特网互联协议）、UDP协议（User Datagram Protocol，用户数据报协议）和其他一些协议的协议组。

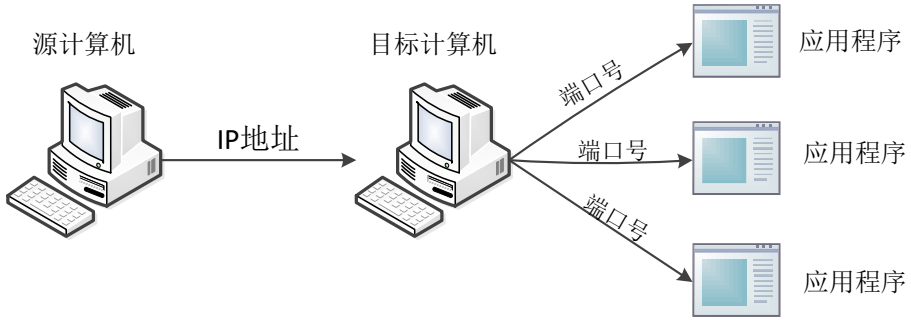
1 网络编程基础

- IP地址和端口号
- 要想使网络中的计算机能够进行通信，还必须为每台计算机指定一个标识号，通过这个标识号来指定接收数据的计算机或者发送数据的计算机。
- 在TCP/IP协议中，这个标识号就是IP地址，它可以唯一标识一台计算机，目前，IP地址广泛使用的版本是IPv4，它是由4个字节大小的二进制数来表示，如：00001010000000000000000000000001。
- 由于二进制形式表示的IP地址非常不便记忆和处理，因此通常会将IP地址写成十进制的形式，每个字节用一个十进制数字（0-255）表示，数字间用符号“.”分开表示4段数字，如“10.0.0.1”。

1 网络编程基础

- 端口号
- 通过IP地址可以连接到指定计算机，但如果想访问目标计算机中的某个应用程序，还需要指定端口号。
- 在计算机中，不同的应用程序是通过端口号区分的。
- 端口号是用两个字节（16位的二进制数）表示的，它的取值范围是0~65535；
- 其中，0~1023之间的端口号用于一些知名的网络服务和应用，用户的普通应用程序需要使用1024以上的端口号。

1 网络编程基础



- 位于网络中一台计算机可以通过IP地址去访问另一台计算机，并通过端口号访问目标计算机中的某个应用程序

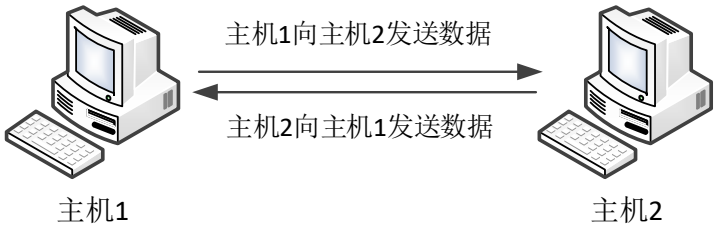
1 网络编程基础

- 在JDK中提供了一个与IP地址相关的InetAddress类，该类用于封装一个IP地址，并提供了一系列与IP地址相关的方法。

InetAddress——常用方法

方法声明	功能描述
InetAddress getByName(String host)	获取给定主机名的的IP地址，host参数表示指定主机
InetAddress getLocalHost()	获取本地主机地址
String getHostName()	获取本地IP地址的主机名
boolean isReachable(int timeout)	判断在限定时间内指定的IP地址是否可以访问
String getHostAddress()	获取字符串格式的原始IP地址

- UDP协议：UDP是无连接通信协议，即在数据传输时，数据的发送端和接收端不建立逻辑连接。
- 使用UDP协议消耗资源小、通信效率高、延迟小，所以通常都会用于音频、视频和普通数据的传输，例如视频会议都使用UDP协议。
- 使用UDP协议传送数据时，由于UDP的面向无连接性，不能保证数据的完整性，因此在传输重要数据时不建议使用UDP协议。



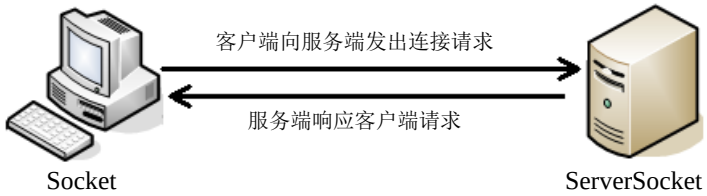
- TCP协议是面向连接的通信协议，即在传输数据前先在发送端和接收端建立逻辑连接，然后再传输数据，保证了两台计算机之间可靠无差错的数据传输。
- TCP协议传送速度较慢，但传送的数据比较可靠。
- 由于TCP协议的面向连接特性，它可以保证传输数据的安全性和完整性，所以是一个被广泛采用的协议，例如在下载文件时，如果数据接收不完整，将会导致文件数据丢失而不能被打开，因此，下载文件时必须采用TCP协议。

2 TCP通信

- TCP通信同UDP通信一样，都能实现两台计算机之间的通信，通信的两端则都需要创建Socket对象。
- TCP通信与UDP通信的其中一个主要区别在于，UDP中只有发送端和接收端，不区分客户端与服务器端，计算机之间可以任意地发送数据；
- 而TCP通信是严格区分客户端与服务器端的，在通信时，必须先由客户端去连接服务器端才能实现通信，服务器端不可以主动连接客户端。

2 TCP通信

- JDK中提供了ServerSocket类，表示服务器端；Socket类表示客户端。
- 通信时，首先要创建代表服务器端的ServerSocket对象，该对象相当于开启一个服务，并等待客户端的连接；然后创建代表客户端的Socket对象，并向服务器端发出连接请求，服务器端响应请求，两者建立连接后可以正式进行通信。



ServerSocket——常用构造方法

- ServerSocket()
  - 该构造方法在创建ServerSocket对象时并没有指定端口号，使用时还需要调用bind(SocketAddress endpoint)方法将其绑定到指定的端口号上。
- ServerSocket(int port)
  - 该构造方法在创建ServerSocket对象时，可以绑定到指定的端口号上。如果port参数值为0，此时系统就会分配一个未被其他程序占用的端口号。由于客户端需要根据指定的端口号来访问服务器端程序，因此端口号随机分配的情况并不常用，通常都会给服务器端指定一个端口号。

ServerSocket——常用方法

方法声明	功能描述
Socket accept()	该方法用于等待客户端的连接，在客户端连接之前一直处于阻塞状态，如果有客户端连接就会返回一个与之对应的Socket对象
InetAddress getInetAddress()	该方法用于返回一个InetAddress对象，该对象中封装了ServerSocket绑定的IP地址
boolean isClosed()	该方法用于判断ServerSocket对象是否为关闭状态，如果是关闭状态则返回true，反之则返回false
void bind(SocketAddress endpoint)	该方法用于将ServerSocket对象绑定到指定的IP地址和端口号，其中参数endpoint 封装了IP 地址和端口号

Socket——常用构造方法

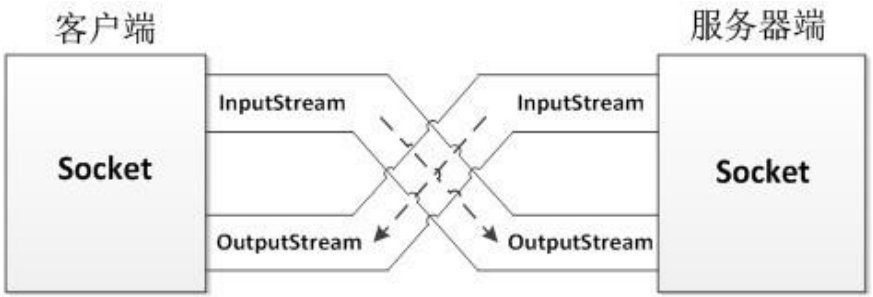
- Socket()
  - 该构造方法在创建Socket对象时，并没有指定IP地址和端口号，创建对象后还需调用connect(SocketAddress endpoint)方法，才能完成与指定服务器端的连接，其中参数endpoint用于封装IP地址和端口号。
- Socket(String host, int port)
  - 该构造方法在创建Socket对象时，会根据参数去连接在指定地址和端口上运行的服务器程序，其中参数host接收的是一个字符串类型的IP地址。
- Socket(InetAddress address, int port)
  - 该方法在使用上与第2个构造方法类似，参数address用于接收一个InetAddress类型的对象，该对象用于封装一个IP地址。

Socket——常用方法

方法声明	功能描述
int getPort()	该方法用于返回此Socket连接的远程服务端的端口号
InetAddress getLocalAddress()	该方法用于获取Socket对象绑定的本地IP地址，并将IP地址封装成InetAddress类型的对象返回
void close()	该方法用于关闭Socket连接，结束本次通信。在关闭Socket之前，应将与Socket相关的所有的输入/输出流全部关闭，这是因为一个良好的程序应该在执行完毕时释放所有的资源
InputStream getInputStream()	该方法返回一个InputStream类型的输入流对象。如果该对象是由服务器端的Socket返回，就用于读取客户端发送的数据；反之，用于读取服务器端发送的数据
OutputStream getOutputStream()	该方法返回一个OutputStream类型的输出流对象。如果该对象是由服务器端的Socket返回，就用于向客户端发送数据；反之，用于向服务器端发送数据

2 TCP通信

服务端与客户端通信图：



2 TCP通信

- 基于TCP协议的Socket通信的步骤：
- 服务器：
  - 1、创建ServerSocket，绑定一个监听端口
  - 2、通过accept()方法监听客户端请求
  - 3、建立连接后，通过输入流读取客户端数据，通过输出流向客户端发送数据
  - 4、关闭输入/输出流，关闭Socket
- 客户端：
  - 1、创建Socket，需要指明服务器的IP地址和端口号
  - 2、建立连接后，通过输出流向服务器发送数据，通过输入流读取服务器的响应信息
  - 3、关闭输入输出流，关闭Socket

Thank You