云南大学数学与统计学院  
《计算机网络实验》上机实践报告

|  |  |  |
| --- | --- | --- |
| 课程名称：计算机网络实验 | 年级：2015级 | 上机实践成绩： |
| 指导教师：陆正福 | 姓名：刘鹏 | 专业：信息与计算科学 |
| 上机实践名称：基于UDP协议与Socket接口的网络通信编程实验 | 学号：20151910042 | 上机实践日期：2018-10-26 |
| 上机实践编号：No.04 | 组号： |  |

# 实验目的

1. 熟悉基于UDP协议与Socket接口的网络通信编程实验
2. 熟悉教材第二章的基本概念

# 实验内容

1. 掌握基于UDP协议与Socket接口的网络通信编程的流程
2. 使用Java实现基于UDP协议与Socket接口的PC网络通信编程
3. 使用Java和Android实现基于UDP协议与Socket接口的移动网络通信编程（选做）
4. 使用Python实现基于UDP协议与Socket接口的网络通信编程（选做）

（说明：3和4至少选做1项；调试所用实例可以源自书本和网络，但是应有属于自己的修改，不可以照搬照抄）

# 实验平台

Windows 10 Pro 1803；

Cygwin GCC编译器。

JDK 11 for Windows；

# 程序代码

## 基于UDP协议与Socket接口的网络通信编程的流程

UDP，是User Datagram Protocol，即用户数据报协议。其关键在于“数据包”，主要就是把数据进行打包然后发送给目标，而不管目标是否接收到数据。

主要流程就是发送者打包数据（Datagram Packet），通过Datagram Socket发送，接收者收到数据进行拆解。下面简要地进行分析此过程。

### 服务端接收、发送流程



### 客户端发送、接受流程



## 使用Java实现基于UDP协议与Socket接口的PC网络通信编程

UDP与TCP在应用上的不同，主要体现在使用的Socket上面。TCP是创建Socket，UDP是创建DatagramSocket，这也恰恰反映了两者的区别，UDP基于数据报，TCP基于数据流。详见4.1。

## 使用Java和Android实现基于UDP协议与Socket接口的移动网络通信编程

本选做题继续选择使用Java语言。UDP协议的实现核心就是构造数据包。

### 应用设计

服务器开启一个端口号为2222的UDP进程，然后一直监听来自外部的访问，直到收到一个packet。这里的packet设置为1MB大小。手机App是客户端，仅仅向服务器发送一个字符串，然后接受数据并将之打印到屏幕上。

### 实验过程分析

Java客户端需要连接网络，所以需要赋予其网络权限。

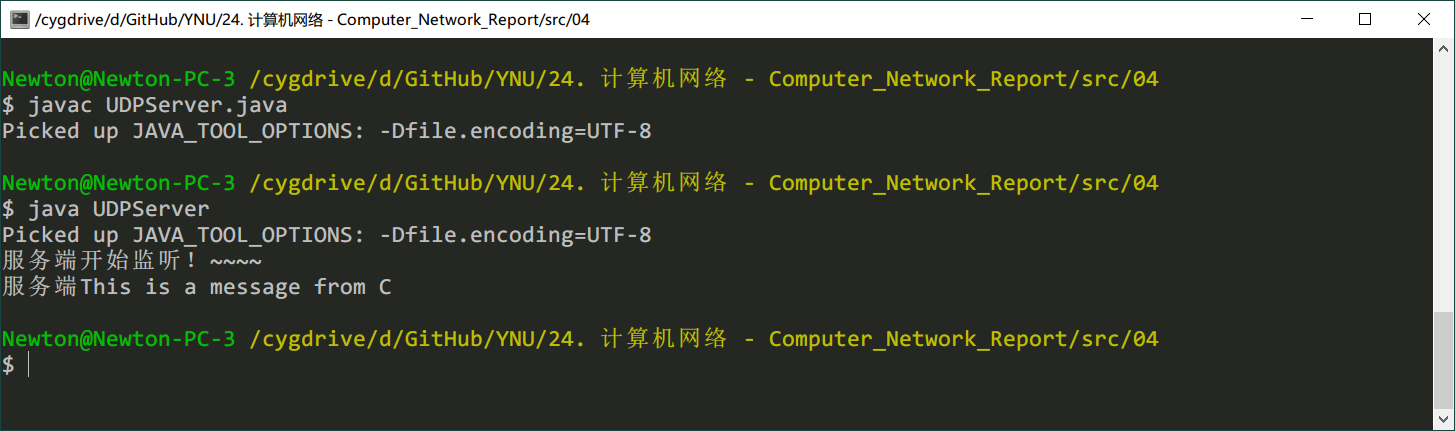
### Server端Java代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | **import** java**.**io**.**IOException**;**  **import** java**.**net**.**DatagramPacket**;**  **import** java**.**net**.**DatagramSocket**;**  **import** java**.**net**.**InetSocketAddress**;**  public class UDPServer **{**  public static void main**(**String**[]** args**)** **throws** IOException **{**  // 生成一个1MB的存储空间  byte**[]** buf **=** **new** byte**[**1024**];**  // 1. 接收数据  // (1) 创建接受数据的数据包  DatagramPacket packet **=** **new** DatagramPacket**(**buf**,** buf**.**length**);**  // (2) 创建UDP的Socket  DatagramSocket socket **=** **new** DatagramSocket**(**2222**);**  // (3) 接收数据  System**.**out**.**println**(**"服务端开始监听！~~~~"**);**  socket**.**receive**(**packet**);**  // (4) 处理数据  System**.**out**.**println**(**"服务端" **+** **new** String**(**buf**,** 0**,** buf**.**length**));**  // 2. 返回数据  DatagramPacket p **=** **new** DatagramPacket**(**buf**,** buf**.**length**,** packet**.**getAddress**(),** packet**.**getPort**());**  socket**.**send**(**p**);**  socket**.**close**();**  **}**  **}** |

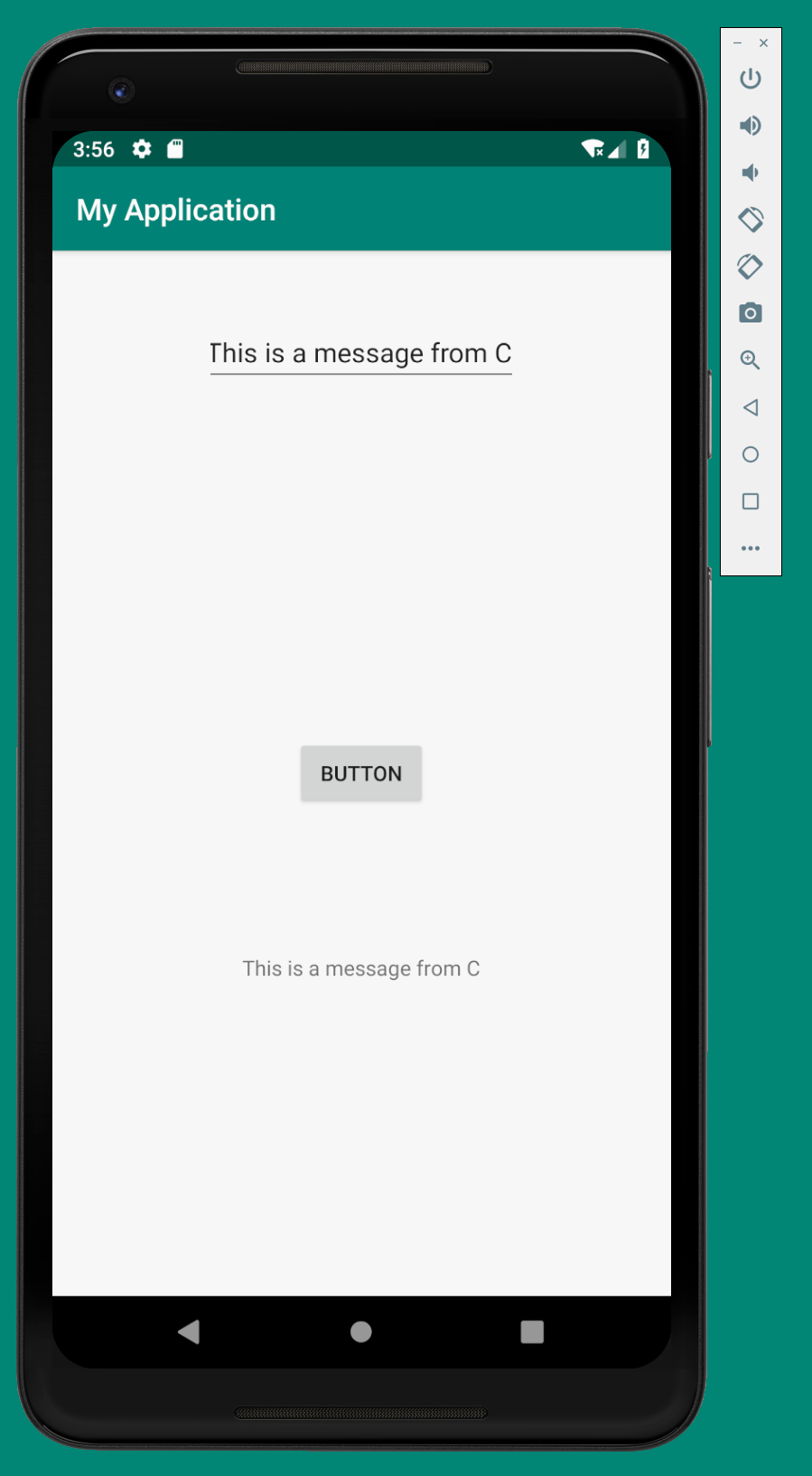
### Android App端Java代码

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76 | package com**.**example**.**newton**.**myapplication**;**  **import** android**.**support**.**v7**.**app**.**AppCompatActivity**;**  **import** android**.**os**.**Bundle**;**  **import** android**.**widget**.**EditText**;**  **import** android**.**widget**.**TextView**;**  **import** android**.**widget**.**Button**;**  **import** java**.**io**.**IOException**;**  **import** java**.**net**.**DatagramPacket**;**  **import** java**.**net**.**DatagramSocket**;**  **import** java**.**net**.**InetAddress**;**  **import** java**.**net**.**UnknownHostException**;**  **import** java**.**net**.**SocketException**;**  **import** android**.**view**.**View**;**  public class MainActivity **extends** AppCompatActivity **{**  EditText et**;**  TextView tv**;**  Button button**;**  @Override  protected void onCreate**(**Bundle savedInstanceState**)** **{**  **super.**onCreate**(**savedInstanceState**);**  setContentView**(**R**.**layout**.**activity\_main**);**  et **=** findViewById**(**R**.**id**.**et**);**  tv **=** findViewById**(**R**.**id**.**tv**);**  button **=** findViewById**(**R**.**id**.**button**);**  **}**  public void onClick**(**View view**)** **{**  **new** Thread**()** **{**  @Override  public void run**()** **{**  **super.**run**();**  byte**[]** bytes **=** et**.**getText**().**toString**().**getBytes**();**  **try** **{**  // 接收数据  InetAddress address **=** InetAddress**.**getByName**(**"192.168.1.75"**);**  // 1. 构造数据包  DatagramPacket packet **=** **new** DatagramPacket**(**bytes**,** bytes**.**length**,** address**,** 2222**);**  // 2. 创建数据报套接字并将其绑定到本地主机上的指定端口。  DatagramSocket socket **=** **new** DatagramSocket**();**  // 3. 从此套接字发送数据报包。  socket**.**send**(**packet**);**  // 接收数据  // 1. 构造 DatagramPacket，用来接收长度为 length 的数据包。  final byte**[]** bytes1 **=** **new** byte**[**1024**];**  DatagramPacket receiverPacket **=** **new** DatagramPacket**(**bytes1**,** bytes1**.**length**);**  socket**.**receive**(**receiverPacket**);**  runOnUiThread**(new** Runnable**()** **{**  @Override  public void run**()** **{**  tv**.**setText**(new** String**(**bytes1**,** 0**,** bytes1**.**length**));**  **}**  **});**  socket**.**close**();**  **}** **catch** **(**UnknownHostException e**)** **{**  e**.**printStackTrace**();**  **}** **catch** **(**SocketException e**)** **{**  e**.**printStackTrace**();**  **}** **catch** **(**IOException e**)** **{**  e**.**printStackTrace**();**  **}**  **}**  **}.**start**();**  **}**  **}** |

### Server端运行结果



### Android客户端运行结果



# 实验体会

因为之前做过有关TCP的Java编程，所以许多框架上的问题可以直接拿过来套用。重点是关心TCP与UDP的区别。从Java的实现来看，核心区别是采用的类不同，TCP采用Socket类，而UDP采用DatagramPacket Socket。这两种类的具体实现是有本质差异的。

下面引用之前曾经出现过的图进行对比：

|  |  |
| --- | --- |
| **TCP Server** | **UDP Server** |
| **TCP Client** | **UDP Client** |

具体的区别，是由TCP和UDP的传输协议的差异决定的。

# 参考文献

[1] **林锐**. 高质量 C++/C 编程指南 [M]. 1.0 ed., 2001.

[2] java IO: https://zhuanlan.zhihu.com/p/21444494

[3] java NIO: https://www.jianshu.com/p/093b7c408dba

[4] java NIO: https://docs.oracle.com/javase/7/docs/api/java/nio/package-summary.html

[5] java NET: https://docs.oracle.com/javase/7/docs/api/java/net/package-summary.html