**第4讲 网络文件传输程序设计**

**教学与实践目的**：**学会基本的文件传输FTP程序设计技术**

## 一、介绍

**程序设计知识点：**字节流（网络字节流和文件字节流）的读写技术。

前两讲我们学会了使用TCP套接字(Socket)，能实现字符串的发送和接收功能，简单地做到了客户机和服务器的对话。

这次课，我们进一步学习TCP套接字，利用它的字节传输技术，实现网络文件传输。

文件传输协议规定（RFC 959），网络文件传输中用两个TCP端口来实现:

* 一个端口（21号）用来对话，传递控制信息，总是开启；
* 一个端口（20号）实现文件数据传递服务，有数据传输服务时开启。

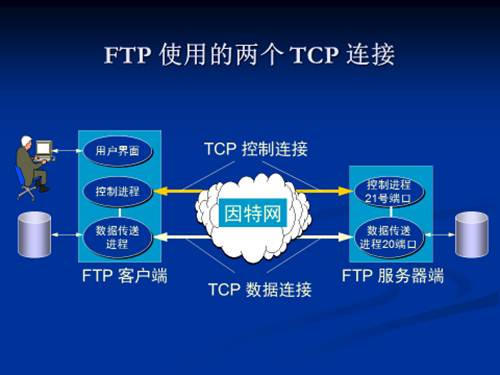


图4.1 FTP的两个端口

**网络对话和网络文件传输，使用TCP的socket编程，本质还是一样，对话过程，使用字符流来包装；而网络文件传输过程，则应该使用字节流来进行处理。**

本讲实现一个简单的远程文件传输系统，实现基本的下载功能。

我们用**2021端口**实现对话服务，如身份验证、文件列表信息浏览等，用**2020端口**传递数据文件（即文件下载）。

### 基于C/S的主要程序结构如下：

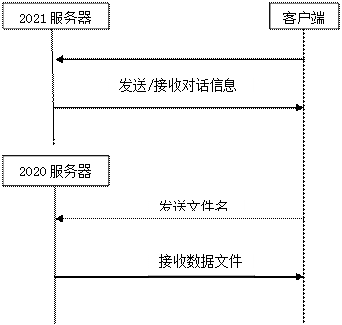


图4.2 C/S软件结构图

客户端程序：

主界面客户端程序FileClientFX.java；

文件对话客户端程序（控制进程）FileDialogClient.java；

文件数据客户端程序（数据传输进程）FileDataClient.java。

服务端程序：

文件对话服务器程序FileDialogServer.java，开启2021端口；

主要功能:身份验证、文件目录传送。

文件数据服务器程序FileDataServer.java，开启2020端口。

主要功能：传送文件名，接收文件。

## 二、程序设计第一步：创建客户端远程文件对话程序

该客户端程序主要功能：发送用户信息、实现和文件服务器的基本对话，文件浏览和下载。

**注意**：客户端发送信息后，能够看到什么内容，如何交互，这就和服务端的程序编写有关，客户端无法控制。例如，这一讲服务器就提供了文件列表浏览、传输文件等功能，也就有了更多约定：发送 **dir** 显示文件列表、发送 **help** 显示帮助信息、**bye**表示退出；发送文件名，服务器会回馈文件名是否正确、能否下载，再根据提示使用“下载”按钮完成下载动作。

1. 新建一个程序包：chapter04，这这个包下面创建client和server这两个程序包，本讲所有的客户端程序都创建在client包中。

2. 发送用户信息等功能和之前学习的程序要求基本一致，可以将上一讲的TCPClient.java程序借用过来，复制、重构、并命名为FileDialogClient.java，原有的方法保持不变。

3. 创建FileClientFX窗体程序。该窗体界面如图4.3所示。

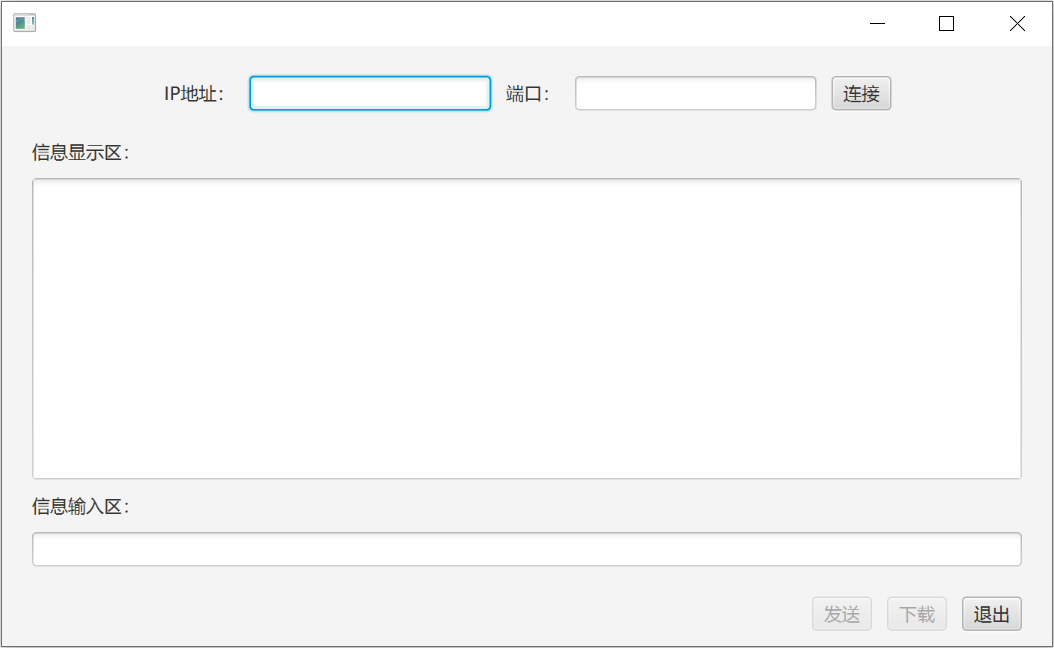


图4.3文件传输管理主界面

（1）程序界面和之前的任务大致相似，同样策略，可以复制、重构TCPClientThreadFX程序，重命名为FileClientFX，并在代码中增加“下载”按钮。

注意，使用idea重构此类以及前面的类，可能会提示有多处需要重构重命名，这是idea比较智能的地方，因为窗体类多处使用了TCPClient，这个类重命名后，窗体中对应的变量名也需要修改，这样比自己手动修改方便很多（重构时会将TCPClient的调用改为FileDialogClient的调用，TCPClient类型的实例变量名也会自动重命名为fileDialogClient，如果有报错，说明有地方没有修改完全，则自行仔细检查，手动完成）；

（2）在程序中增加ip和port两个String类型的成员变量，用来保存对应文本框的的输入信息；

（3）增加一个便捷功能：鼠标拖动加亮信息显示区的文字，拖动的同时把加亮的文字同步复制到信息输入框中，这个功能是为了方便下载文件（后续要实现点击下载按钮则读取信息输入框中的文件名，并下载该文件）；

这个功能的实现比较特别，这种情况一般会处理鼠标拖动事件来解决(mouse dragged)，例如，把文本框TextField的内容复制到文本域TextArea，只要setOnMouseDragged 处理鼠标拖动事件就可以。但从TextArea复制内容到TextFiled，这种方式对不适用。文本域是多行文本，鼠标拖动加亮文字不会触发鼠标拖动事件。解决方案是为文本域的选择属性selectionProperty添加监听器（相当于是为输入文本框和显示文本域做了有条件的属性绑定）：

*//信息显示区鼠标拖动高亮文字直接复制到信息输入框，方便选择文件名*

*//taDispaly为信息选择区的TextArea，tfSend为信息输入区的TextField*

*//为taDisplay的选择范围属性添加监听器，当该属性值变化(选择文字时)，会触发监听器中的代码*taDisplay.selectionProperty().addListener((observable, oldValue, newValue) -> {  
 *//只有当鼠标拖动选中了文字才复制内容* if(!taDisplay.getSelectedText().equals(""))  
 tfSend.setText(taDisplay.getSelectedText());  
});

## 三、程序设计第二步：创建客户端数据传送进程FileDataClient.java

主要功能：连接服务器数据端口、发送文件名、保存下载的文件，文件传输完成后关闭数据连接。

该程序有2个方法：

（1）构造方法，FileDataClient(String ip, String port)，主要功能是向服务器的数据端口请求连接；

（2）文件下载方法getFile(fileName)[[见附录代码]](#_1._FileDataClient.java中的getFile方法代码)。

主要功能是先在本地新建一个空文件，向服务器发送其文件名（基于字符串输出流操作），然后接收网络文件数据并保存为本地的这个文件（基于字节流操作），关闭数据套接字。

在窗体界面的**“下载”按钮**中调用该方法，例如：

btnDownload.setOnAction(event -> {  
 if(tfSend.getText().equals("")) *//没有输入文件名则返回* return;  
 String fName = tfSend.getText().trim();  
 tfSend.clear();  
  
 FileChooser fileChooser = new FileChooser();  
 fileChooser.setInitialFileName(fName);  
 File saveFile = fileChooser.showSaveDialog(null);  
 if (saveFile == null) {  
 return;*//用户放弃操作则返回* }   
 try {  
 *//数据端口是2020* new FileDataClient(ip, "2020").getFile(saveFile);  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setContentText(saveFile.getName() + " 下载完毕！");  
 alert.showAndWait();  
 *//通知服务器已经完成了下载动作，不发送的话，服务器不能提供有效反馈信息* fileDialogClient.send("客户端开启下载");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
});

**注意**，若下载文件为0字节，首先就要检查是否做了flush和关闭文件的操作。一些逻辑判断、ip地址、端口号等各种错误都可能造成错误，这种情况就善用idea的断点调试功能，慢慢观察各个变量值，找出bug。另外，为了避免不小心带入空白符，建议程序中获取文本信息都使用trim()方法消除空白。

下载完成后，第20行代码给服务器发送了“**客户端开启下载**”，这就类似bye一样都是服务端和客户端的约定，服务端的代码已经对这个字符串做了判断，接收到这个字符串就知道客户端开始了下载，这样就可以给客户端发送一些反馈信息。

## 四、程序设计第三步：文件对话服务器程序FileDialogServer.java

文件对话服务器程序开启2021端口，用来传送双方的对话信息，如文件列表等信息，本质上和第二讲的TCPServer类似，但要额外提供发送文件列表的功能，而且服务器要使用多线程模式，本讲不强制要求同学制作该程序（该程序运行在教师机172.16.229.253 : 2021上，同学们也可自行制作类似的服务器，提升程序设计能力），附录中有[部分代码](#_2._对话服务端程序中的文件列表服务模块（供参考）)。

## 五、程序设计第四步：文件数据服务器程序FileDataServer.java

文件数据服务器程序专门提供文件下载服务，开启2020端口，提供文件下载服务，不强制要求同学制作该程序（该程序运行在教师机172.16.229.253 : 2020上，同学们也可自行制作类似的服务器，提升程序设计能力），附录中有部分[下载文件的代码](#_3._文件数据服务器程序部分代码)，和客户端的getFile方法配合使用提供下载功能。

## 六、实验过程(课堂计分)

（1）启动客户端FileClientFX.java程序，连接172.16.229.253 : 2021服务器，关注提示信息，根据提示信息操作；

（2）根据信息显示区的文件列表，在信息输入区输入要下载文件的全名（如果实现了快捷功能， 可以使用鼠标加亮方式），发送此文件名，服务器会给出相应反馈提示信息；

（3）若能成功从服务器下载文件，你将**获得今天的平时成绩**，下次课可查询成绩。

## 附录

### 1. FileDataClient.java中的getFile方法代码

public void getFile(File saveFile) throws IOException {  
  
 if (dataSocket != null) { *// dataSocket是Socket类型的成员变量* FileOutputStream fileOut = new FileOutputStream(saveFile);*//新建本地空文件* byte[] buf = new byte[1024]; *// 用来缓存接收的字节数据  
 //网络字节输入流* InputStream socketIn = dataSocket.getInputStream();  
 *//网络字节输出流* OutputStream socketOut = dataSocket.getOutputStream();  
  
 *//(2)向服务器发送请求的文件名，字符串读写功能* PrintWriter pw = new PrintWriter(new OutputStreamWriter(socketOut, "utf-8"), true);  
 pw.println(saveFile.getName());  
  
 *//(3)接收服务器的数据文件，字节读写功能* int size = 0;  
 while ((size = socketIn.read(buf)) != -1) {*//读一块到缓存，读取结束返回-1* fileOut.write(buf, 0, size); *//写一块到文件* }  
 fileOut.flush();*//关闭前将缓存的数据全部推出  
 //文件传输完毕，关闭流* fileOut.close();  
 if (dataSocket != null) {  
 dataSocket.close();  
 }  
 } else {  
 System.*err*.println("连接ftp数据服务器失败");  
 }  
}

### 2. 文件对话服务器程序中的文件列表服务模块（供参考）

public void fileListPushToClient(PrintWriter pw) {  
 String path = "d:/temp/ftpserver"; *//给出下载目录路径* File filePath = new File(path);  
 if (!filePath.exists()) { *//路径不存在则返回* System.*out*.println("ftp下载目录不存在");  
 return;  
 }  
 *//如果不是一个目录就返回* if (!filePath.isDirectory()) {  
 System.*out*.println("不是一个目录");  
 return;  
 }  
 *//开始显示目录下的文件，不包括子目录* fileNames = filePath.list();  
 File tempFile;  
 *// 格式化文件大小输出，不保留小数，不用四舍五入，有小数位就进1* DecimalFormat formater = new DecimalFormat();  
 formater.setMaximumFractionDigits(0);  
 formater.setRoundingMode(RoundingMode.*CEILING*);  
  
 for (String fileName : fileNames) {  
 tempFile = new File(filePath + "/" + fileName);  
 if (tempFile.isFile()) {  
pw.println(" " +  
 fileName + " " + formater.format(tempFile.length() / 1024.0) + "KB");  
 }  
 }  
 }

### 3. 文件数据服务器程序部分代码

try {  
 InputStream socketIn = socket.getInputStream();  
 BufferedReader br = new BufferedReader(new InputStreamReader(socketIn, "utf-8"));  
  
 String downFileName = br.readLine();  
 System.*out*.println("要下载的文件为：" + downFileName);  
String filePath = "d:/ftpserver";  
 if (downFileName == null || !isValidFileName(downFileName, filePath)) {  
 socket.close(); *//文件名无效，关闭连接* return;  
 }  
 downFileName = filePath + "/" + downFileName;  
 *//读取ftp服务器上的文件，写出到网络字节流* OutputStream socketOut = socket.getOutputStream();  
 FileInputStream fileIn = new FileInputStream(downFileName);  
 byte[] buf = new byte[1024]; *//用来缓存字节数据* int size;  
 while ((size = fileIn.read(buf)) != -1) { *//读取结束返回-1* socketOut.write(buf, 0, size);  
 }  
 socketOut.flush();  
 socketOut.close();  
 fileIn.close();  
  
 System.*out*.println(downFileName + " 文件传输结束");  
} catch (IOException e) {  
 e.printStackTrace();  
} finally {  
 if (socket != null) {  
 try {  
 socket.close(); *//关闭socket及其关联的输入输出流* } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}