

学号	0121710880415
----	---------------

武汉理工大学

《面向对象与多线程综合实验》报告

学 院	计算机科学与技术学院
专 业	软件工程
班 级	计算机类 m1704
姓 名	颜道江
指导教师	许毅

日期 2018 年 11 月 27 日

3.输入输出流

3.1 实验目的

了解 java 中 I/O 流的概念和种类；掌握字节流和字符流处理，包括 File 类，InputStream/OutputStream 及其子类，Reader/Writer 及其子类；熟练掌握文件的顺序处理，随机访问处理；熟悉对象串行化的概念和方法。

3.2 系统功能描述

在第三次的实验中利用流的方式进行文件的读写，首先本次的实验中增加了一个 Doc 类，主要作为获取文件的文件名，文件拥有者，文件的属性等等信息的初始化和文件信息的返回。

本次的实验中主要进行操作的就是操作员的上传和下载的模拟，另外就是要进行文件列表的显示，在执行相关的菜单操作的时候进行文件的相关的信息的输出。

档案模块的操作可以概括如下：

1. 档案查询:根据系统中已经有的数据，浏览已有的档案信息
2. 档案下载：根据相关用户的权限，实现文件的下载功能，同时将下载的文件保存到指定的目录中
3. 档案上传：档案录入人员专用，实现档案的上传，本次实验在本地进行数据库的模拟，从指定的目录实现文件的上传。

3.3 模块设计

本次的实验中增加了一个 Doc 类，这个类在教学资料中进行了给出，在实际编码的过程中我直接使用了给出的代码。本次设计到的几个操作如下：

- 文件上传：
 1. 本地模拟上传：在本地模拟文件的上传，将文件从系统的任意位置写入到指定的位置，即在指定的位置创建一个文件将要上传的文件保存到该目录下；
 2. 文件信息的写入：将上传的文件的信息保存到档案系统中，此处即将文件插入到哈希表中。
- 文件下载：
 1. 在系统（哈希表）中进行查找，如果再到输入的信息说明这个文件能够下

载，如果没有返回 False

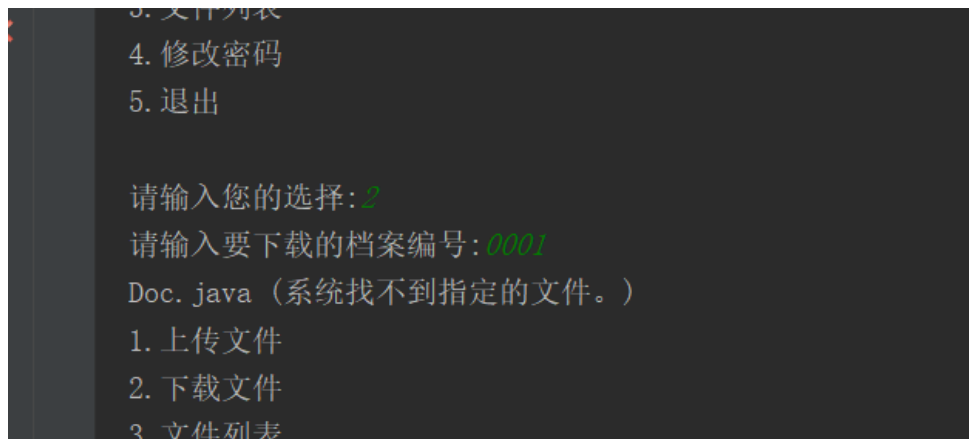
2. 文件下载：如果在哈希表中找到文件的相关信息就进行下载，下载过程及按照流的方式从系统中读入文件，然后将文件写到指定的位置

- 文件列表：

类似之前用户列表的操作，对系统中的文件信息利用哈希表的相关的函数进行文件信息的枚举，然后按照指定的格式进行文件信息的输出

3.4 开发难点与体会

- 错误信息



```
3. 文件列表
4. 修改密码
5. 退出

请输入您的选择: 2
请输入要下载的档案编号: 0001
Doc. java (系统找不到指定的文件。)
1. 上传文件
2. 下载文件
3. 文件列表
```

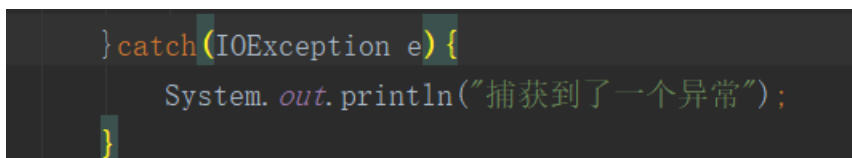
分析：

首先通过按照流的方式进行文件的读写，在这部分的操作中遇到的第一个问题就是通过输入文件的编号，只能确定在系统中（本次实验为哈希表）中好到了这个文件，但是不能正确的进行文件的下载。

处理办法：

要处理这个问题首先要定位到产生问题的地方。程序运行的过程中无法进行文件的下载，涉及到的类应该是父类中的下载函数的问题，或者是在操作员的位置产生了异常。

在处理的过程中我将 Operator 类中的一个异常处理部分进行了反馈信息的重写。



```
} catch (IOException e) {
    System.out.println("捕获到了一个异常");
}
```

然后通过运行确定了抛出异常的位置：

请输入您的选择:2
请输入要下载的档案编号:0001
捕获到了一个异常

通过本次的运行确定到了抛出异常的位置为 User 类中的文件下载出现了问题，因此在这个类的构建文件的位置设置了一个断点进行调试。

```
@ public FileInputStream( @NotNull File file) throws FileNotFoundException { file: "Doc.java"
    String name = (file != null ? file.getPath() : null); name: "Doc.java"
    SecurityManager security = System.getSecurityManager(); security: null
    if (security != null) {
        security.checkRead(name); security: null
    }
    if (name == null) { name: "Doc.java"
        throw new NullPointerException();
    }
    if (file.isInvalid()) { file: "Doc.java"
        throw new FileNotFoundException("Invalid file path");
    }
}

@ private FileNotFoundException(String path, String reason) { path: "Doc.java" reason: "系统找不到指定的文件。"
    super(path + ((reason == null) ? "" : " " + reason)); path: "Doc.java" reason: "系统找不到指定的文件。"
}
```

最终通过调试却定这是 FileNotFoundException 抛出的异常。然后我对中断的地方进行了检查，查到是文件的路径设置的问题。对保存文件的目录的绝对路径进行设置最后解决了问题。

体会：

这次在编写的过程中我遇到了一个小的问题就是文件路径的问题，虽然这个问题并不是特别的严重，但是在调试的过程中却花了我大量的时间才定位到了问题。

通过这次的调试我获得了以下的经验：

1. 如何分析产生异常的位置，然后设置断点逐步的进行调试
2. 另外的一个经验就是在编写相关的异常处理的时候要尽可能的将 try 语句中的部分变得更加的简单，这样有利于程序中出现异常的时候快速的查找到是程序的哪一位置产生的异常。然后进行调试解决，提高效率。

● 错误信息:

```
1. 修改密码
2. 删除文件
3. 上传文件
4. 下载文件
5. 退出

请输入您的选择: 2
请输入要下载的档案编号: 0001
D:\Java_Project\Lesson3\File\downloadfile (拒绝访问。)
```

原因:

在这里我自己通过调试并没有解决这个问题, 这里出现的问题我分析仍然是文件路径的问题, 但是检查仍然没有找到问题, 于是通过上网查找了一下

```
at org.springframework.web.servlet.FrameworkServlet.doPost(FrameworkServlet.java:718)
```

出现此问题原因在于:

在创建输出流的时候, 只写了文件的名称, 如果要实现上传则必须是文件的路径; 才能正常上传; 如果有遇到类似请看下自己的路径是否正确; 代码中丢了path, 因此报了“拒绝访问”的错误。

```
String path = request.getServletContext().getRealPath("/attr/");
File folder = new File(path);
if(!folder.exists()){
    folder.mkdirs();
}
String fileName = myfile.getOriginalFilename();
if(!fileName.isEmpty()){
    String ext = FilenameUtils.getExtension(fileName);
    String newFileName = new Date().getTime()+"_"+new Random().nextInt(1000000000)+ext;
    myfile.transferTo(new File(path+File.separator+newFileName));
}
```

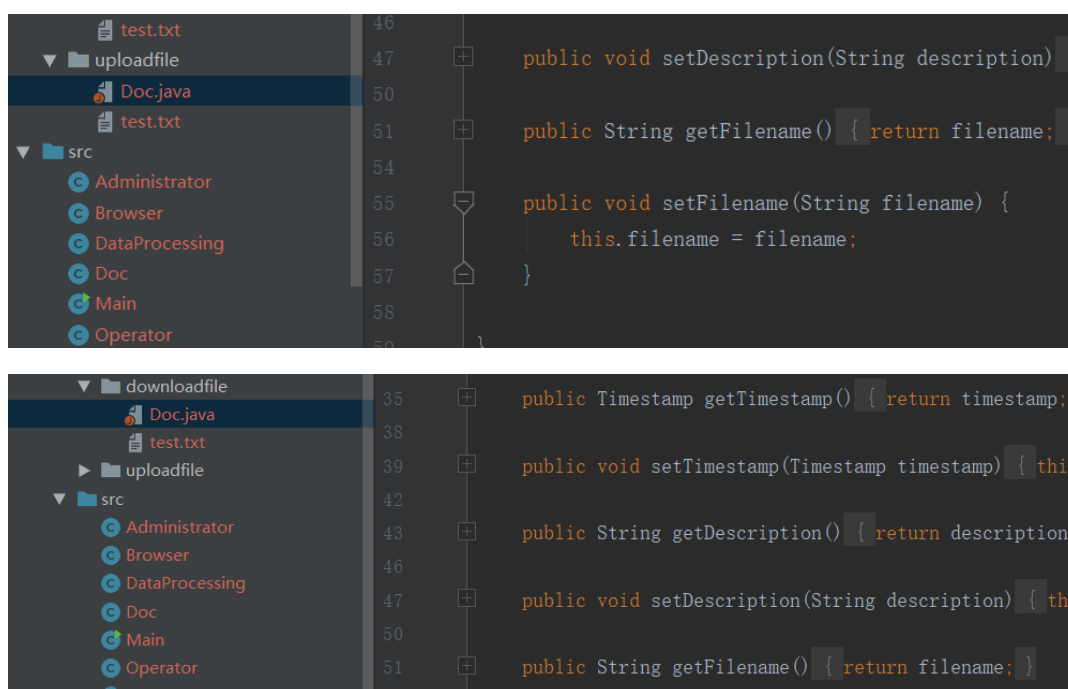
https://blog.csdn.net/Sky_rain

网上的一篇博客分析依然是路径的问题, 于是我直接将有关路径的部分代码删除进行了重写, 解决而问题。

体会:

虽然对于这个问题直到解决我也没弄清楚原因, 只能大致判断是文件的路径的问题, 但是经过重写还是解决了问题。这也算一种经验吧, 遇到了问题, 反复检查也不能解决, 如果代码较简单就直接进行重写。

● 错误信息



```
46
47 + public void setDescription(String description)
50
51 + public String getFilename() { return filename; }
54
55 - public void setFilename(String filename) {
56     this.filename = filename;
57 - }
58
59 }

35 + public Timestamp getTimestamp() { return timestamp; }
38
39 + public void setTimestamp(Timestamp timestamp) { this.timestamp = timestamp; }
42
43 + public String getDescription() { return description; }
46
47 + public void setDescription(String description) { this.description = description; }
50
51 + public String getFilename() { return filename; }
```

这里我实现了文件的下载但是通过检查下载的文件我发现的问题是，我下载保存的文件不是完整的，这是部分文件。从文件的行号来看是缺少了 9 行，更加具体的是文件从 2k 变成了 1k。

原因：

通过对文件大小的具体的变化我查找到死在进行文件的读写的时候使用的字节数组的大小为 `byte[1024]`，所以文件下载下来就变成了 1k。

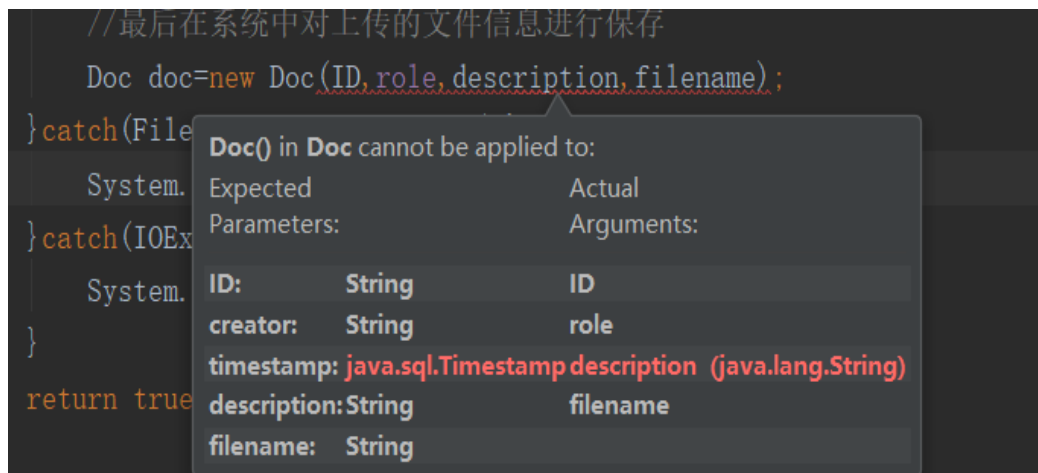
解决办法：



```
public class FileInputStreamDemo {
    public static void main(String args[]) {
        try {
            FileInputStream fin=new FileInputStream("D:\\test.txt");
            int i,j=0;
            byte[] content = new byte[fin.available()];
            while((i=fin.read())!=-1){
                content[j]=(byte)i;
                j++;
            }
            //fin.read(content);
            fin.close();
            System.out.println(new String(content));
        } catch(IOException e){
            System.out.println(e);
        }
    }
}
```

最后我的解决办法是根据课件上提供的方法，在文件进行读取的时候，对指定的文件读取文件的全部字节。

- 错误信息:



在创建一个新的文件对象时，他的构造函数中有一个参数为时间戳，这是我在之前并没有碰到过的，于是查阅了网上 Java 时间戳的相关信息。

获取当前时间戳

```
//方法 一
System.currentTimeMillis();

//方法 二
Calendar.getInstance().getTimeInMillis();

//方法 三
new Date().getTime();
```

https://www.cnblogs.com/zhujiabin/p/6168671.html?utm_source=itdadao&utm_medium=referral

通过上面地址的博客的阅读，我通过创建了一个 Date 对象完成了对时间信息的写入

```
//最后在系统中对上传的文件信息进行保存
Date date=new Date();
Timestamp timestamp=new Timestamp(date.getTime());
DataProcessing.insertDoc(ID,name,timestamp,description,uploadfile.getName());
```

3.5 实验总结

本次的实验是关于输入输出流相关的内容，关于这部分的内容在 Java 程序课上并没有进行太多的讲述，只是在大一的 C++面向对象中对流的概念有一定的了解。虽然本次的实验并不是太难，但是还是遇到了不小的困难。首先是在编写程序的时候犯了几个低级的错误就是关于文件路径的问题，实验中我遇到了无法找到文件的异常和文件拒绝访问的异常。并且有一个问题我经过调试并没有解决直接通过相关部分的代码的重写得到的实现。同时在本次的实验中文件的读写中我使用的是课件上提供的一个办法来获取到文件中的全部的内容并将其进行模拟下载相关的操作。

最后通过本次的实验获得几个方面的收获就是程序的调试和程序相关异常的查找定位。另外，通过本次的实验我知道了 java 中的时间戳的用法，以及如何获取到时间并且进行时间的转换，让时间按照指定的格式进行输出。