第二十天

1、 RandomAccessFile

不属于流，是 Object 类的子类，但它融合了 InputStream 和 OutputStream 的功能，支持对文件的读取和写入随机访问。

RandomAccessFile 的访问模式有如下四个值：

r：以只读方式打开指定文件。如果试图对该 RandomAccessFile 指定的文件执行写入方法则会抛出 IOException

rw：以读取、写入方式打开指定文件。如果该文件不存在，则尝试创建文件

rws：以读取、写入方式打开指定文件。相对于 rw 模式，还要求对文件的内容或元数据的每个更新都同步写入到底层存储设备，默认情形下(rw 模式下),是使用 buffer 的,只有 cache 满的或者使用 RandomAccessFile.close()关闭流的时候儿才真正的写到文件

rwd：与 rws 类似，只是仅对文件的内容同步更新到磁盘，而不修改文件的元数据

RandomAccessFile 对象包含一个记录指针，用以标识当前读写处的位置

long getFilePointer()：返回文件记录指针的当前位置

void seek(long pos)：将文件记录指针定位到 pos 位置

-->RandomAccessFileTest.java

作业：多线程拷贝（读懂）-->CopyThreadTest.java,ExpressCopy.java

2、 对象持久化

ObjectInputStream/ObjectOutputStream：可以读取、写出对象

readObject()/writeObject()

读写两个对象-->ObjectStreamTest.java

java.io.NotSerializableException

Java 序列化是指把 Java 对象转换为字节序列的过程；而 Java 反序列化是指把字节序列恢复为 Java 对象的过程

只有实现了 Serializable 接口的类的对象才能被序列化

transient（短暂的、临时的）修饰的属性不会被序列化

Java 的序列化机制是通过在运行时判断类的 serialVersionUID 来验证版本一致性的。在进行反序列化时，JVM 会把传来的字节流中的 serialVersionUID 与本地相应实体（类）的 serialVersionUID 进行比较，如果相同就认为是一致的，可以进行反序列化，否则就会出现序列化版本不一致的异常(java.io.InvalidClassException)

选择流的步骤：

1）以自己的 java 程序为参照，选择输入或输出流

2）处理的数据是否需要让人读懂，选择字节流或字符流

3）根据业务特点选择具体字节、字符流的子类

3、 jdk7 中资源释放：try(打开资源){可能出现异常的代码}catch(){}，资源自动释放

```
try (创建流对象语句，如果多个,使用';'隔开) {
   // 读写数据
} catch (IOException e) {
```

```
    e.printStackTrace();
}
 try ( FileWriter fw = new FileWriter("a.txt"); ) {
        fw.write("Hello!");
  } catch (IOException e) {
        e.printStackTrace();
  }
```

注意：实现 java.lang.AutoCloseable 接口的对象才可以自动关闭

4、 网络编程概念

计算机网络：是相互连接的独立自主的计算机的集合，

网络编程:实现网络中计算机上运行程序间的数据交换

使用的包：java.net/java.io/java.lang.Thread

OSI(Open System Interconnection)七层参考模型

物理层：二进制传输，确定如何在通信信道上传递比特流；

数据链路层：加强物理层的传输功能，建立一条无差错的传输线路；

网络层：在网络中数据到达目的地有很多线路，网络层就是负责找出最佳的传输线路；

传输层：传输层为源端计算机到目的端计算机提供可靠的数据传输服务，隔离网络的上下层协议，使得上层网络应用的协议与下层无关；

会话层：在两个相互通信的应用进程之间建立、组织和协调其相互之间的通信；

表示层：处理被传送数据的表示问题，也就是信息的语法和语义，如有必要将使用一种通用的格式在多种格式中进行转换；

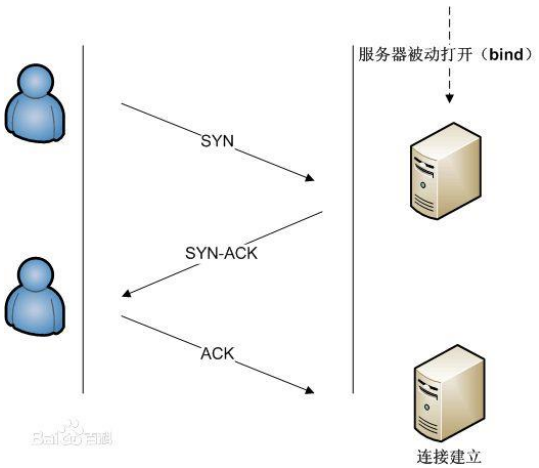
应用层：为用户的应用程序提供网络通信服务；

网络协议：为计算机网络中进行数据交换而建立的规则、标准或约定的集合

1）应用层：远程登录协议 Telnet、文件传输协议 FTP（网上下载一个软件或者资料的时候就会使用该协议）、 超文本传输协议 HTTP（使用较多，通过 IE 浏览一个网页的时候就使用该协议）、域名服务 DNS（使用较多，通过网络访问一个计算机一般不使用该主机的 IP 地址，而是通过该主机的域名访问）、简单邮件传输协议 SMTP（通过 Foxmail 发送邮件）、邮局协议 POP3 等（通过 Foxmail 收邮件）；

2）传输层：传输控制协议 TCP、用户数据报协议 UDP；

TCP：传输控制协议，面向连接（三次握手）、可靠的数据传输；占用带宽，效率低；
提供数据确认和重传的机制，保证数据一定能够到达数据接收端。



UDP：用户数据报协议，无连接、不可靠数据传输；占用带宽小，效率高；

不需要建立连接，可以直接向一个 IP 地址发送数据，至于是不是能够收到不能保证，发送过程中数据有可能丢失、IP 地址可能不存在、IP 地址代表的主机没有运行等原因都可能导致不能接收到数据。

IP 地址：唯一标识网络中的计算机
172.16.7.250-->32 个位
点分四段表示法，每段取值范围[0,255]
32 位包含：网络地址+主机地址
255.255.255.0
子网掩码：网络地址
127.0.0.1-->代表本机 IP，localhost

端口号：唯一识别同一计算机上的不同网络程序
编写网络程序就需要绑定一个端口号，端口号范围从 0-65535，尽量使用 1024 以上的，1024 以下的留给系统程序使用
常用端口：

ftp：21
telnet：23
http : 80
mysql: 3306
oracle: 1521
tomcat: 8080
QQ: 4000

java.io.File
Socket：套接字，网络程序进行双向通信的链路节点，包含 IP、端口等信息，可以向网络发送请求，也可以响应其它网络程序的请求
是对网络设备的抽象

5、 基于 TCP 协议网络服务器端编程步骤：
1）创建服务器端 Socket，并绑定在某一端口上
ServerSocket ss = new ServerSocket(port);
2）接收客户请求，返回客户端 Socket
Socket s = ss.accept();
3）获取客户端的输入、输出流
s.getInputStream();
s.getOutputStream();
4）封装输入、输出流
5）执行读、写操作
6）释放资源
close();

基于 TCP 协议网络客户端编程步骤：

1）创建 Socket，并指定服务器的 IP 和端口信息
Socket s = new Socket(serverAddress,serverPort);
2）获取服务器端的输入、输出流
s.getInputStream();
s.getOutputStream();
3）封装输入、输出流
4）执行读、写操作
5）释放资源
close();

基于 TCP 协议单线程服务器、客户端-->TimeTcpServer.java、TimeTcpClient.java
基于 TCP 协议多线程服务器、客户端-->TimeTcpThreadServer.java

端口号被占用解决方案:
win7 解决方案:（如果提示权限不够那么可以使用管理员权限打开 cmd 命令提示符,在开始附件中鼠标右键选择管理员运行即可)
netstat -aon|findstr 8081
找到进程的 PID　xxxx
然后杀死这个进程
taskkill /pid xxxx

ubuntu 解决方案
netstat -anp | grep 6666
找到进程的 PID　xxxx
然后杀死这个进程
kill -9 xxxx