复习：

1. 线程：任务

线程的定义：继承Thread类，实现Runnable接口，实现Callable接口

多线程的并发安全问题：多个线程相互抢占，抢占是发生在了每一步，导致出现了一些不符合常理的数据。

同步锁机制：synchronized---锁对象：要求被所有的线程都认识---共享资源、类的字节码、this

死锁：多个线程的锁不同且形成了嵌套导致程序无法继续往下执行---统一锁对象、减少锁嵌套

等待唤醒机制：利用wait和notify、notifyAll

守护线程：被守护的线程结束，守护线程随之结束

线程的优先级：1-10

线程的状态：就绪、阻塞、运行

2. 单例模式：全局过程中只存在了一个实例。

## 网络编程

基于网络进行数据传输---IO

网络模型：

物理层 数据链路层 网络层--IP 传输层---UDP/TCP 会话层 表示层 应用层 --- http,ftp,POP3,SMTP...

IP地址：IPv4---用了4组数字表示一个地址，每一组数字的大小是0-255---IPv6

端口号：0-65535，0-1024在开发过程中不要使用

域名：每一级域名之间用 . 隔开

DNS解析服务器：将域名和地址进行对应

### UDP

基于流的。不建立连接，不可靠，传输速率相对较快。需要对数据进行封包，每个包不超过64K。---DatagramSocket, DatagramPacket --- 适用于要求传输速度不要求可靠性的场景， 视频聊天

发送端:

1. 创建套接字对象

2. 准备数据包，将数据放到数据包中并且要绑定发送的地址

3. 发送数据

4. 关流

接收端：

1. 创建套接字对象，绑定要监听的接收端口

2. 准备数据包

3. 接收数据

4. 关流

5. 将数据从数据包进行解析

练习：单人聊天---一个线程表示发送端，另一个线程表示接收端

255.255.255.255在UDP中表示的是一个广播地址

### TCP

基于流的。需要建立连接，经过三次握手，可靠，传输速率相对较低。理论上不对数据的大小进行限制。---适用于要求可靠性而不要求传输速率的场景 文件传输、文字聊天

客户端：---Socket

1. 创建客户端的套接字对象

2. 发起连接，绑定地址

3. 获取输出流对象，写出数据

4. 通知服务器端数据写完

5. 关流

服务器端：---ServerSocket

1. 创建服务器端的套接字对象，并且绑定监听的端口号

2. 接收连接，获取到一个Socket对象

3. 获取输入流，读取数据

4. 通知客户端数据已经读完

5. 关流

注意：receive/connect/accept/read/write都会产生阻塞

扩展：BIO --- Blocking IO --- 同步式阻塞式IO

NIO --- New IO --- NonBlocking IO --- 同步式非阻塞式IO --- JDK1.4

AIO --- Asynchronous IO --- 异步式非阻塞式IO --- jdk1.8

练习：文件的上传

网络编程.docx

扩展：如何传输一个目录？

扩展：利用TCP实现客户端之间的聊天

class Message implements Serializable {

String hostname; // 本机

String toAdrress; // 对方IP

byte type; // 确定请求类型

// 0 文字

// 1 语音

byte[] data; // 数据

int port; // 端口号

// 根据请求类型确定对应的端口号

// 文字 2001-3000

// 语音 3001-4000

}

## 注解

解释程序---给机器看的

用@interface来定义注解---jdk1.5出现的

@Override

@SuppressWarnings---压制警告

元注解：修饰注解的注解

@Target---限制注解的使用范围的--- 如果注解没有指定范围，则表示任意一个范围都可以使用

@Retention --- 限制注解出现的时间的

@Documented --- 表示当前这个注解能够生成到文档中

@Inherited --- 表示此注解能够限制子类