复习：

1. 线程---三种定义方式，多线程并发安全问题---同步锁---锁对象，线程等待唤醒机制，守护线程

2. 单例模式---要求全局过程中存在一个唯一的实例

练习：生产消费模型

一个线程表示生产者，另一个线程表示消费者。商品数量总数不超过1000

0 300 140 160

160 400 380 180

180 820 0 1000

1000 0 630 370

## 套接字

实际上是一套用于网络通信的API---本质上是一套基于网络传输数据的流

IP地址---IPv4---由四组数组成的IP地址，每组数的范围在0~255之间---IPv6

端口---用于和外界进行信息交互的媒介--- 0~65535---其中0~1024已经被计算机内部以及一些常用应用占用

news.baidu[.com --- 域名](http://www.baidu.com---域名)

DNS服务器---将域名解析为对应的IP地址

网络模型

七层

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

### UDP

基于流的。不需要建立连接，不可靠。传输速度比较快，会对数据进行封包，每个包不超过64k。适用与一些要求速度而不要求可靠性的场景。

发送端：

1. 创建套接字对象

2. 准备数据包，绑定要发送的地址

3. 发送数据

4. 关流

接收端：

1. 创建套接字对象，绑定端口号

2. 准备数据包

3. 接收数据

4. 关流

5. 将数据从数据包解析出来

练习：单人聊天

255.255.255.255---广播地址---只要是同一个网段内的计算机都可以收到这个数据

### TCP

基于流的。建立连接，经过三次握手，可靠。传输速度相对较慢，理论上不限制数据大小。适用于要求可靠性而不要求速度的场景。

客户端：

1. 创建客户端套接字对象

2. 发起连接，绑定连接地址

3. 获取输出流，写出数据

4. 通知服务器端数据写出完毕

5. 关流

服务器端：

1. 创建服务器端套接字对象，绑定端口号

2. 接受连接，获取一个Socket对象

3. 获取输入流，读取数据

4. 通知客户端数据已经读取完毕

5. 关流

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

扩展：BIO---BlockingIO---同步式阻塞式IO

NIO---NewIO---NonBlockingIO---同步式非阻塞式IO---JDK1.4

AIO---AsynchronousIO---异步式非阻塞式IO---JDK1.8

练习：上传文件

反射