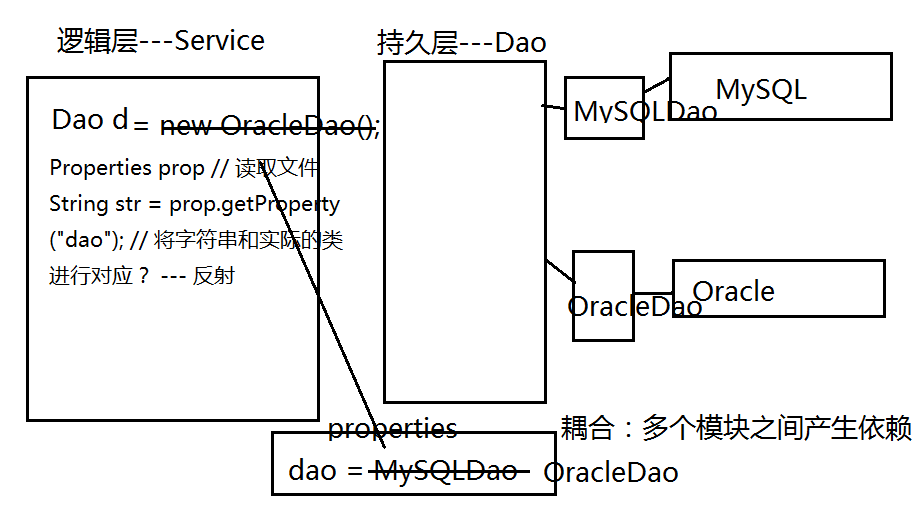
复习：

1. UDP：基于流的。不建立连接，不可靠，传输速率相对较快。在发送数据的时候需要对象数据进行封包，每个包不超过64K --- 发送端和接收端 --- DatagramSocket。

2. TCP：基于流的。建立连接，经过三次握手，可靠，传输速率相对较慢。不限制数据的大小 --- 客户端(Socket)和服务器端(ServerSocket)

## 反射



能够实现解耦操作

Class --- 代表字节码的类 --- 代表类的类

Field --- 代表属性的类

Constructor --- 代表构造方法的类

Method --- 代表方法的类

Package --- 代表包的类

解剖一个类，然后分析这个类的字节码以获取字节码对象以及对应的实例对象

### 如何去获取一个Class对象？

1. 通过类名.class方式来获取一个字节码对象

2. 通过对象.getClass()方式来获取这个对象的实际类型对应的字节码

3. 通过Class.forName(类的全路径名)方式来获取指定类的字节码

### 获取实例对象

1. 如果这个类中提供了无参构造，那么可以直接利用字节码对象.newInstance()来获取对应的实例对象

2. 如果想使用这个类中含参构造来创建对应的实例对象，那么需要先获取这个类中对应形式的构造函数，然后再调用newInstance(Object...)来创建实例对象

如果在获取这个类中的属性/方法/构造方法的时候，这些属性/方法是非公有的，那么在执行之前需要进行暴力破解---setAccessiable

## JVM的一些运行参数

栈内存(Stack)：执行代码块---计算---每一个线程独有

堆内存(Heap)：存储对象 --- 被所有线程共享的

新生代(Young Generation)：伊甸园区(eden)和幸存区(Survivor)

老生代(Old Generation)

持久代(Permanent)/方法区(Method Area)：存储---被所有线程所共享的

静态常量池：存储类信息 --- 字节码

运行时常量池：字面量(数值、字符、字符串、逻辑值和null)和自定义常量

本地方法栈(Native Stack)：执行本地方法（native）---每一个线程独有的

寄存器/PC计数器(programming counter)：用于存储和调度线程中的指令---每一个线程都都有一个PC计数器

所有的栈所占用的内存不能超过物理内存的1/3或者是2G

堆内存所占用的大小不能超过物理内存的1/4或者是2G

-... 标准参数

-X... 非标准参数

-Xss 用于限制栈内存大小的

-Xss128K 表示栈内存是128K

-Xms 表示堆内存分配的初始大小

-Xms5M 表示堆内存初始大小是5M

-Xmx 限制堆内存的最大占用大小

-Xmx10M 表示堆内存最大是10M

-Xmn 限制新生代的大小

扩展参数 -XX:

-XX:+PrintGC

-XX:+PrintGCDetails

### 回收机制

Mark-Cleaning：先找到无用的对象，然后将对象标记，最后清理标记的对象---清理相对较快，但是会产生内存的碎片化

Copying：将内存分为两块，使用其中的一块，如果要回收，就将不需要回收的对象挪到另一块去，然后重建要回收的这一块区域 --- 不会产生内存的碎片化，但是耗费内存并且耗费时间

Mark-Sweeping：既能保证清理速度又能保证内存不会产生碎片化

jdk1.7采取的回收器是G1回收，G1回收算法针对新生代和老生代采取不同的回收机制。