一、Java 基础知识

1、Object 类相关方法

getClass 获取当前运行时对象的 Class 对象。

hashCode 返回对象的 hash 码。

clone 拷贝当前对象， 必须实现 Cloneable 接口。浅拷贝对基本类型进行值拷贝，对引用类型拷贝引用；深拷贝对基本类型进行值拷贝，对引用类型对象不但拷贝对象的引用还拷贝对象的相关属性和方法。两者不同在于深拷贝创建了一个新的对象。

equals 通过内存地址比较两个对象是否相等，String 类重写了这个方法使用值来比较是否相等。

toString 返回类名@哈希码的 16 进制。

notify 唤醒当前对象监视器的任一个线程。

notifyAll 唤醒当前对象监视器上的所有线程。

wait 1、暂停线程的执行；2、三个不同参数方法（等待多少毫秒；额外等待多少毫秒；一直等待）3、与 Thread.sleep(long time) 相比，sleep 使当前线程休眠一段时间，并没有释放该对象的锁，wait 释放了锁。

finalize 对象被垃圾回收器回收时执行的方法。

2、基本数据类型

整型：byte(8)、short(16)、int(32)、long(64)

浮点型：float(32)、double(64)

布尔型：boolean(8)

字符型：char(16)

3、序列化

Java 对象实现序列化要实现 Serializable 接口。

反序列化并不会调用构造方法。反序列的对象是由 JVM 自己生成的对象，不通过构造方法生成。

序列化对象的引用类型成员变量，也必须是可序列化的，否则，会报错。

如果想让某个变量不被序列化，使用 transient 修饰。

单例类序列化，需要重写 readResolve() 方法。

4、String、StringBuffer、StringBuilder

String 由 char[] 数组构成，使用了 final 修饰，是不可变对象，可以理解为常量，线程安全；对 String 进行改变时每次都会新生成一个 String 对象，然后把指针指向新的引用对象。

StringBuffer 线程安全；StringBuiler 线程不安全。

操作少量字符数据用 String；单线程操作大量数据用 StringBuilder；多线程操作大量数据用 StringBuffer。

5、重载与重写

重载 发生在同一个类中，方法名相同，参数的类型、个数、顺序不同，方法的返回值和修饰符可以不同。

重写 发生在父子类中，方法名和参数相同，返回值范围小于等于父类，抛出的异常范围小于等于父类，访问修饰符范围大于等于父类；如果父类方法访问修饰符为 private 或者 final 则子类就不能重写该方法。

6、final

修饰基本类型变量，一经出初始化后就不能够对其进行修改。

修饰引用类型变量，不能够指向另一个引用。

修饰类或方法，不能被继承或重写。

7、反射

在运行时动态的获取类的完整信息

增加程序的灵活性

JDK 动态代理使用了反射

8、JDK 动态代理

使用步骤

创建接口及实现类

实现代理处理器：实现 InvokationHandler ，实现 invoke（Proxy proxy，Method method，Object[] args） 方法

通过 Proxy.newProxyInstance(ClassLoaderloader, Class[] interfaces, InvocationHandler h) 获得代理类

通过代理类调用方法。

9、Java IO

普通 IO ，面向流，同步阻塞线程。

NIO，面向缓冲区，同步非阻塞。