**Java常用类&IO流&反射**

**String相关**

1. 字符型常量和字符串常量的区别

形式上: 字符常量是单引号引起的一个字符; 字符串常量是双引号引起的若干个字符

含义上: 字符常量相当于一个整型值( ASCII 值),可以参加表达式运算; 字符串常量代表一个地址值(该字符串在内存中存放位置)

占内存大小: 字符常量只占**2个**字节; 字符串常量占若干个字节(至少一个字符结束标志) (注意： **char在Java中占两个字节**)

1. 什么是字符串常量池

为了提高性能和减少开销，JVM提供了一个特殊的内存——常量池  
常量池，顾名思义：就是存储常量的内存空间，有String字符串常量池、包装类常量池等

字符串常量池有助于为Java运行时节省大量空间，虽然创建字符串时需要更多的时间。

当我们使用双引号创建一个字符串时，首先在字符串常量池中查找是否有相同值的字符串，如果发现则返回其引用，否则它会在池中创建一个新的字符串，然后返回新字符串的引用。

如果使用new运算符创建字符串，则会强制String类在堆空间中创建一个新的String对象。我们可以使用intern()方法将其放入字符串常量池或从字符串常量池中查找具有相同的值字符串对象并返回其引用

1. String是最基本的数据类型吗

String不是基本数据类型，是一个引用数据类型。基本数据类型有8种。。。。

1. String有哪些特性

不变性：对它进行任何操作，其实都是创建一个新的对象，再把引用指向该对象。不变模式的主要作用在于当一个对象需要被多线程共享并频繁访问时，可以保证数据的一致性。

常量池优化：String 对象创建之后，会在字符串常量池中进行缓存，如果下次创建同样的对象时，会直接返回缓存的引用。

final：使用 final 来定义 String 类，表示 String 类不能被继承，提高了系统的安全性。

1. String为什么是不可变的

**看到String类被final修饰**。这里你就要说出被final修饰的类不能被继承，方法不能被重写，变量不能被修改。

1. String真的是不可变的吗

利用反射机制也可以变

1. 是否可以继承Stirng类

不可以，因为String类被final修饰

1. String str=”i”与String str=new String（”i”）一样吗

不一样，因为内存的分配方式不一样。String str="i"的方式，Java 虚拟机会将其分配到常量池中；而 String str=new String(“i”)方式，则会被分到堆内存中。

1. String s=new String（”xyz”）创建了几个字符串对象

两个对象，一个是静态区的”xyz”，一个是用 new 创建在堆上的对象。

1. 如何将字符串反转

reverse（）

1. 数组有没有 length（）方法？String 有没有length（）方法

数组：length属性

String：length（）方法

1. String类的常用方法有哪些

indexOf():返回指定字符的索引。  
charAt():返回指定索引处的字符。  
replace():字符串替换。  
trim():去除字符串两端空白。  
split():分割字符串，返回一个分割后的字符串数组。  
getBytes():返回字符串的byte类型数组。  
length():返回字符串长度。  
toLowerCase():将字符串转成小写字母。

1. 在使用HashMap的时候，用String做key有什么好处

HashMap 内部实现是通过 key 的 hashcode 来确定 value 的存储位置，因为字符串是不可变的，所以当创建字符串时，它的 hashcode 被缓存下来，不需要再次计算，所以相比于其他对象更快。

1. String和StringBuffer、StringBuilder的区别是什么？ String为什么是不可变的

String是不可变字符串，StringBuffer和StringBuilder是可变字符串

看到String类被final修饰。这里你就要说出被final修饰的类不能被继承，方法不能被重写，变量不能被修改。

**Date相关**

**包装类相关**

1. 自动装箱与拆箱

装箱： 将基本类型用它们对应的引用类型包装起来

拆箱： 将包装类型转换为基本数据类型

1. int和Integer的区别

值的存储

int 存储在栈中

Integer 对象的引用存储在栈空间中，对象的数据存储在堆空间中。

初始化

int 初始化值为0。

Integer 初始化值为null。

传参

int 是值传递，栈中的数据不可变。

Integer 对象是引用传递，引用不可变，但是引用指向的堆空间地址中的值是可以改变的。

泛型支持

泛型不支持int，但是支持Integer。

运算

int 可以直接做运算，是类的特性。

Integer 的对象可以调用该类的方法，但是在拆箱之前不能进行运算，需要转化为基本类型int。

**单元测试**

**日志**

**IO流**

1. java中的io流分几种

IO流可以理解为连接****程序****与****文件/数组/网络连接/数据库****。

1. ****按流向分类：****

输入流  
 输出流

1. ****按处理数据不同分类：****  
    字节流：二进制，可以处理一切文件，包括：纯文本、doc、音频、视频等。

FileInputStream

FileOutputStream

字符流：文本文件，只能处理纯文本。

FileReader

FileWriter

BufferedReader

BufferedWriter

1. BIO，NIO，AIO有什么区别

同步阻塞的BIO、同步非阻塞的NIO、异步非阻塞的AIO。

1. Files的常用方法都有哪些

Files. exists()：检测文件路径是否存在。  
Files. createFile()：创建文件。  
Files. createDirectory()：创建文件夹。  
Files. delete()：删除一个文件或目录。  
Files. copy()：复制文件。  
Files. move()：移动文件。  
Files. size()：查看文件个数。  
Files. read()：读取文件。  
Files. write()：写入文件。

反射

1. 什么是反射机制

Java反射机制是在运行状态中，对于任意一个类（Class文件），都能够知道这个类的属性和方法；

对于任意一个对象都能调用它的方法和属性；

这种动态获取的信息以及调用对象的方法的功能称为java语言的反射机制。

1. 反射机制优缺点

首先，反射机制极大的提高了程序的灵活性和扩展性，降低模块的耦合性，提高自身的适应能力。

其次，通过反射机制可以让程序创建和控制任何类的对象，无需提前硬编码目标类。

再次，使用反射机制能够在运行时构造一个类的对象、判断一个类所具有的成员变量和方法、调用一个对象的方法。

最后，反射机制是构建框架技术的基础所在，使用反射可以避免将代码写死在框架中。

java的反射机制就是增加程序的灵活性，避免将程序写死到代码里。

1. 反射机制的应用场景有哪些

1.我们在使用JDBC连接数据库时使用class.forName()通过反射加载数据库的驱动程序  
2.Spring框架的IOC（动态加载管理Bean）创建对象以及AOP功能都和反射有联系  
3.动态配置实例的属性

1. Java获取反射的三种方法

1. 通过new对象实现反射机制（ 对象.getClass() ）

2. 通过路径实现反射机制（ Class.forName("包名.类名") ）

3. 通过类名实现反射机制 （ 类名.Class ）

网络编程