## [Java 面试题基础部分（一）](http://www.cnblogs.com/wlx520/p/4467487.html)

<http://www.cnblogs.com/wlx520/p/4524901.html>

**1** 、一个**".java"**源文件中是否可以包括多个类（不是内部类）？有什么限制？  
可以有多个类，但只能有一个 public 的类，并且 public 的类名必须与文件名相一致。  
**2**、 **Java** 有没有 **goto?**  
java 中的保留字，现在没有在 java 中使用。  
**3**、说说**&**和**&&**的区别。  
&和&&都可以用作逻辑与的运算符，表示逻辑与（and），当运算符两边的表达式的结果都  
为 true 时，整个运算结果才为 true，否则，只要有一方为 false，则结果为 false。  
&&还具有短路的功能，即如果第一个表达式为 false，则不再计算第二个表达式，例如，对  
于 if(str != null&& !str.equals(“”))表达式，当 str 为 null 时，后面的表达式不会执行，所以不  
会出现 NullPointerException 如果将&&改为&，则会抛出 NullPointerException 异常。  
If(x==33 &++y>0) y 会增长， If(x==33 && ++y>0)不会增长  
&还可以用作位运算符，当&操作符两边的表达式不是 boolean 类型时， &表示按位与操作，  
我们通常使用0x0f 来与一个整数进行&运算，来获取该整数的最低4个 bit 位，例如， 0x31 &  
0x0f 的结果为0x01 。  
备注：这道题先说两者的共同点，再说出&&和&的特殊之处，并列举一些经典的例子来表明  
自己理解透彻深入、实际经验丰富。   
**4**、在 **JAVA** 中如何跳出当前的多重嵌套循环？  
在 Java 中，要想跳出多重循环，可以在外面的循环语句前定义一个标号，然后在里层循环  
体的代码中使用带有标号的 break 语句，即可跳出外层循环。例如，  
ok:  
for(int i=0;i<10;i++) {  
for(int j=0;j<10;j++) {  
System.out.println(“i=” + i + “,j=” + j);  
if(j == 5) break ok;  
}  
}  
另外，我个人通常并不使用标号这种方式，而是让外层的循环条件表达式的结果可以受到里  
层循环体代码的控制，例如，要在二维数组中查找到某个数字。  
int arr[][] ={{1,2,3},{4,5,6,7},{9}};  
boolean found = false;  
for(int i=0;i<arr.length&& !found;i++) {  
for(int j=0;j<arr[i].length;j++){  
System.out.println(“i=” + i + “,j=” + j);  
if(arr[i][j] ==5) {  
found = true;  
break;  
}  
}  
}  
**5**、 **switch** 语句能否作用在 **byte** 上，能否作用在 **long** 上，能否作用在 **String**  
上**?**  
在 switch（ expr1 ）中， expr1 只能是一个整数表达式或者枚举常量（更大字体），整数表达  
式可以是 int 基本类型或 Integer 包装类型，由于， byte,short,char 都可以隐含转换为 int，  
所以，这些类型以及这些类型的包装类型也是可以的。显然， long 和 String 类型都不符合  
switch 的语法规定，并且不能被隐式转换成 int 类型，所以，它们不能作用于 swtich 语句中。  
**6**、 **short s1 = 1; s1 = s1 + 1;**有什么错**? short s1 = 1; s1 += 1;**有什么错**?**  
对于 short s1 = 1; s1 = s1 + 1;由于 s1+1 运算时会自动提升表达式的类型，所以结果是 int型，再赋值给 short 类型 s1 时， 编译器将报告需要强制转换类型的错误。对于 short s1 = 1; s1 += 1;由于 +=是 java 语言规定的运算符， java 编译器会对它进行特殊处理，因此可以正确编译。

1. **char 型变量中能不能存贮一个中文汉字?为什么?**

char 型变量是用来存储 Unicode 编码的字符的， unicode 编码字符集中包含了汉字，所以，char 型变量中当然可以存储汉字啦。不过，如果某个特殊的汉字没有被包含在 unicode 编码字符集中，那么，这个 char 型变量中就不能存储这个特殊汉字。 补充说明： unicode 编码占用两个字节，所以， char 类型的变量也是占用两个字节。备注：后面一部分回答虽然不是在正面回答题目，但是，为了展现自己的学识和表现自己对问题理解的透彻深入，可以回答一些相关的知识，做到知无不言，言无不尽。

1. **用最有效率的方法算出 2 乘以 8 等于几?**

2 << 3，因为将一个数左移 n 位，就相当于乘以了2的 n 次方，那么，一个数乘以8只要将其左移3

[祥龙技术博客园](http://www.cnblogs.com/wlx520/)

## [JAVA面试题基础部分（二）](http://www.cnblogs.com/wlx520/p/4467522.html)

**10**、使用 **final** 关键字修饰一个变量时，是引用不能变，还是引用的对象不能变？  
使用 final 关键字修饰一个变量时，是指引用变量不能变，引用变量所指向的对象中的内容  
还是可以改变的。例如，对于如下语句：  
final StringBuffer a=new StringBuffer("immutable");  
执行如下语句将报告编译期错误：  
a=new StringBuffer("");  
但是，执行如下语句则可以通过编译：  
a.append(" broken!");  
有人在定义方法的参数时，可能想采用如下形式来阻止方法内部修改传进来的参数对象：  
public void method(final StringBuffer param){  
}  
实际上，这是办不到的，在该方法内部仍然可以增加如下代码来修改参数对象：  
param.append("a");  
**11** 、 **"=="**和 **equals** 方法究竟有什么区别？  
（单独把一个东西说清楚，然后再说清楚另一个，这样，它们的区别自然就出来了，混在一  
起说，则很难说清楚）  
==操作符专门用来比较两个变量的值是否相等，也就是用于比较变量所对应的内存中所存  
储的数值是否相同， 要比较两个基本类型的数据或两个引用变量是否相等，只能用==操作  
符。  
如果一个变量指向的数据是对象类型的，那么，这时候涉及了两块内存， 对象本身占用一块  
内存（ 堆内存），变量也占用一块内存，例如 Objet obj = new Object();变量 obj 是一个内存，  
new Object()是另一个内存，此时，变量 obj 所对应的内存中存储的数值就是对象占用的那  
块内存的首地址。对于指向对象类型的变量，如果要比较两个变量是否指向同一个对象，即  
要看这两个变量所对应的内存中的数值是否相等，这时候就需要用==操作符进行比较。  
equals 方法是用于比较两个独立对象的内容是否相同，就好比去比较两个人的长相是否相  
同，它比较的两个对象是独立的。例如，对于下面的代码：  
String a=new String("foo");  
String b=new String("foo");  
两条 new 语句创建了两个对象，然后用 a/b 这两个变量分别指向了其中一个对象，这是两  
个不同的对象，它们的首地址是不同的，即 a 和 b 中存储的数值是不相同的，所以，表达  
式 a==b 将返回 false，而这两个对象中的内容是相同的，所以，表达式 a.equals(b)将返回  
true。  
在实际开发中，我们经常要比较传递进行来的字符串内容是否等，例如， String input  
= …;input.equals(“quit”)，许多人稍不注意就使用==进行比较了，这是错误的，随便从网上  
找几个项目实战的教学视频看看，里面就有大量这样的错误。记住，字符串的比较基本上都  
是使用 equals 方法。  
如果一个类没有自己定义 equals 方法，那么它将继承 Object 类的 equals 方法， Object 类  
的 equals 方法的实现代码如下：  
boolean equals(Object o){  
return this==o;  
}  
这说明，如果一个类没有自己定义 equals 方法，它默认的 equals 方法（从 Object 类继承  
的）就是使用==操作符，也是在比较两个变量指向的对象是否是相同的。

**12**、静态变量和实例变量的区别？  
在语法定义上的区别： 静态变量前要加 static 关键字，而实例变量前则不加。  
在程序运行时的区别：实例变量属于某个对象的属性，必须创建了实例对象，其中的实例变  
量才会被分配空间，才能使用这个实例变量。 静态变量不属于某个实例对象，而是属于类，  
所以也称为类变量，只要程序加载了类的字节码，不用创建任何实例对象，静态变量就会被  
分配空间，静态变量就可以被使用了。总之，实例变量必须创建对象后才可以通过这个对象  
来使用，静态变量则可以直接使用类名来引用。  
例如，对于下面的程序，无论创建多少个实例对象，永远都只分配了一个 staticVar 变量，  
并且每创建一个实例对象，这个 staticVar 就会加1 ；但是，每创建一个实例对象，就会分配  
一个 instanceVar，即可能分配多个 instanceVar，并且每个 instanceVar 的值都只自加了1  
次。  
public class VariantTest{  
public static int staticVar = 0;  
public int instanceVar = 0;  
public VariantTest(){  
staticVar++;  
instanceVar++;  
System.out.println(“staticVar=” + staticVar + ”,instanceVar=”+  
instanceVar);  
}  
}  
备注：这个解答除了说清楚两者的区别

**13**、是否可以从一个 **static** 方法内部发出对非 **static** 方法的调用？  
不可以。因为非 static 方法是要与对象关联在一起的，必须创建一个对象后，才可以在该对  
象上进行方法调用，而 static 方法调用时不需要创建对象，可以直接调用。也就是说， 当一  
个 static 方法被调用时，可能还没有创建任何实例对象，如果从一个 static 方法中发出对非  
static 方法的调用，那个非 static 方法是关联到哪个对象上的呢？这个逻辑无法成立，所以，  
一个 static 方法内部发出对非 static 方法的调用。

**14**、 **Integer** 与 **int** 的区别  
int 是 java 提供的8种原始数据类型之一。 Java 为每个原始类型提供了封装类， Integer 是 java  
为 int 提供的封装类。 int 的默认值为0，而 Integer 的默认值为 null，即 Integer 可以区分出  
未赋值和值为0的区别， int 则无法表达出未赋值的情况，例如，要想表达出没有参加考试和  
考试成绩为0的区别，则只能使用 Integer。在 JSP 开发中， Integer 的默认为 null，所以用  
el 表达式在文本框中显示时，值为空白字符串，而 int 默认的默认值为0，所以用 el 表达式  
在文本框中显示时，结果为0，所以， int 不适合作为 web 层的表单数据的类型。  
在 Hibernate 中，如果将 OID 定义为 Integer 类型，那么 Hibernate 就可以根据其值是否为  
null 而判断一个对象是否是临时的，如果将 OID 定义为了 int 类型，还需要在 hbm 映射文  
件中设置其 unsaved-value 属性为0。  
另外， Integer 提供了多个与整数相关的操作方法，例如，将一个字符串转换成整数， Integer  
中还定义了表示整数的最大值和最小值的常量。

**15**、 **Math.round(11.5)**等於多少**? Math.round(-11.5)**等於多少**?**  
Math 类中提供了三个与取整有关的方法： ceil、 floor、 round，这些方法的作用与它们的英  
文名称的含义相对应，例如， ceil 的英文意义是天花板，该方法就表示向上取整，  
Math.ceil(11.3)的结果为12,Math.ceil(-11.3)的结果是-11 ； floor 的英文意义是地板，该方法  
就表示向下取整， Math.ceil(11.6)的结果为11,Math.ceil(-11.6)的结果是-12；最难掌握的是  
round 方法，它表示“四舍五入”，算法为 Math.floor(x+0.5)，即将原来的数字加上0.5后再向  
下取整，所以， Math.round(11.5)的结果为12， Math.round(-11.5)的结果为-11 。

**16**、下面的代码有什么不妥之处**?**  
1. if(username.equals(“zxx”){}  
username 可能为 NULL,会报空指针错误；改为"zxx".equals(username)  
2. **int** x = 1;  
return x==1?true:false; 这个改成 return x==1;就可以!  
**17**、请说出作用域 **public**， **private**， **protected**，以及不写时的区别  
这四个作用域的可见范围如下表所示。  
说明：如果在修饰的元素上面没有写任何访问修饰符，则表示 friendly。  
作用域 当前类 同一包（package） 子孙类 其他包（package）  
public        √           √                                   √         √  
protected   √           √                                   √         ×  
friendly       √           √                                    ×       ×  
private        √           ×                                    ×       ×  
备注：只要记住了有4种访问权限， 4个访问范围，然后将全选和范围在水平和垂直方向上  
分别按排从小到大或从大到小的顺序排列，就很容易画出上面的图了。

[祥龙技术博客园](http://www.cnblogs.com/wlx520/)

## [JAVA面试题基础部分（三）](http://www.cnblogs.com/wlx520/p/4524901.html)

20、接口是否可继承接口?抽象类是否可实现(implements)接口?抽象类是否可继承具体类(concrete class)?抽象类中是否可以有静态的main方法？

接口可以继承接口。抽象类可以实现(implements)接口，抽象类可以继承具体类。抽象类中可以有静态的main方法。  
备注：只要明白了接口和抽象类的本质和作用，这些问题都很好回答，你想想，如果你是java语言的设计者，你是否会提供这样的支持，如果不提供的话，有什么理由吗？如果你没有道理不提供，那答案就是肯定的了。  
只有记住抽象类与普通类的唯一区别：就是不能创建实例对象和允许有abstract方法。

21、写clone()方法时，通常都有一行代码，是什么？  
clone 有缺省行为，  
super.clone();

22、面向对象的特征有哪些方面

封装：在面向对象的编程语言中，对象是封装的最基本单位，面向对象的封装就是把描述一个对象的属性和行为的代码封装在一个“模块”中

抽象：抽象就是找出一些事物的相似和共性之处，然后将这些事物归为一个类

继承：在定义和实现一个类的时候，可以在一个已经存在的类的基础之上来进行，把这个已经存在的类所定义的内容作为自己的内容，继承是子类自动共享父类数据和方法的机制，这是类之间的一种关系，提高了软件的可重用性和可扩展性。

多态：多态是指程序中定义的引用变量所指向的具体类型和通过该引用变量发出的方法调用在编程时并不确定，而是在程序运行期间才确定，即一个引用变量倒底会指向哪个类的实例对象，该引用变量发出的方法调用到底是哪个类中实现的方法，必须在由程序运行期间才能决定。

23、java中实现多态的机制是什么？

靠的是父类或接口定义的引用变量可以指向子类或具体实现类的实例对象

24、abstract class（抽象类）和interface（接口）有什么区别?

含有abstract修饰符的class即为抽象类，abstract类不能创建的实例对象。含有abstract方法的类必须定义为abstract class，abstract class类中的方法不必是抽象的。abstract class类中定义抽象方法必须在具体(Concrete)子类中实现，所以，不能有抽象构造方法或抽象静态方法。如果的子类没有实现抽象父类中的所有抽象方法，那么子类也必须定义为abstract类型。  
接口（interface）可以说成是抽象类的一种特例，接口中的所有方法都必须是抽象的。接口中的方法定义默认为public abstract类型，接口中的成员变量类型默认为public static final。

下面比较一下两者的语法区别：  
1.抽象类可以有构造方法，接口中不能有构造方法。  
2.抽象类中可以有普通成员变量，接口中没有普通成员变量  
3.抽象类中可以包含非抽象的普通方法，接口中的所有方法必须都是抽象的，不能有非抽象的普通方法。  
4. 抽象类中的抽象方法的访问类型可以是public，protected和（默认类型,虽然  
eclipse下不报错，但应该也不行），但接口中的抽象方法只能是public类型的，并且默认即为public abstract类型。  
5. 抽象类中可以包含静态方法，接口中不能包含静态方法  
6. 抽象类和接口中都可以包含静态成员变量，抽象类中的静态成员变量的访问类型可以任意，但接口中定义的变量只能是public static final类型，并且默认即为public static final类型。  
7. 一个类可以实现多个接口，但只能继承一个抽象类。

25、什么是内部类？内部类可以引用它的包含类的成员吗？有没有什么限制？  
内部类就是在一个类的内部定义的类，内部类中不能定义静态成员。完全可以。如果不是静态内部类，那没有什么限制！

26、String是最基本的数据类型吗?  
基本数据类型包括byte、int、char、long、float、double、boolean和short。  
java.lang.String类是final类型的，因此不可以继承这个类、不能修改这个类。为了提高效率节省空间，我们应该用StringBuffer类

27、是否可以继承String类?  
String类是final类故不可以继承。

28、String和StringBuffer的区别  
JAVA平台提供了两个类：String和StringBuffer，

它们可以储存和操作字符串，即包含多个字符的字符数据。这个String类提供了数值不可改变的字符串。而这个StringBuffer类提供的字符串进行修改。

当你知道字符数据要改变的时候你就可以使用StringBuffer。典型地，你可以使用StringBuffers来动态构造字符数据。另外，String实现了equals方法，new String(“abc”).equals(newString(“abc”)的结果为true,而StringBuffer没有实现equals方法，所以，new StringBuffer(“abc”).equals(newStringBuffer(“abc”)的结果为false。

29、数组有没有length()这个方法? String有没有length()这个方法？  
数组没有length()这个方法，有length的属性。String有有length()这个方法。

30、final, finally, finalize的区别。  
final 用于声明属性，方法和类，分别表示属性不可变，方法不可覆盖，类不可继承。  
内部类要访问局部变量，局部变量必须定义成final类型，例如，一段代码……  
finally是异常处理语句结构的一部分，表示总是执行。  
finalize是Object类的一个方法，在垃圾收集器执行的时候会调用被回收对象的此方法，可以覆盖此方法提供垃圾收集时的其他资源回收，例如关闭文件等。JVM不保证此方法总被调用