Java中的Object类

1. **Object类**是Java中所有类的**根类**，**Java中的任何一个类**都继承于Object，要么**直接**继承于**Object类**，要么**间接**的继承于Object类。
2. **Object类**存在于**java.lang包**中。即**全名java.lang.Object**。
3. Java中的任何一个类都**继承**了Object类中的任何一个方法，不过会对Object类中的方法进行了**覆盖（Override）**操作。
4. **Object类只有1个构造方法，11个成员方法，没有成员变量**。
5. **Object中的成员方法体现了所有类的共性，只要Object类中有此方法，那么Java中的任何一个类都会具有此方法。但是可能会被覆写（重写）Override。**
6. **Object类的构造方法**：只有**1个**。

**如：创建一个Object对象：Object o = new Object();**

1. **Object类中的成员方法**：
2. **equals方法：Object类中的equals方法可以实现任意两个对象比较，因为利用Object进行调用且接收的也是Object类型，（由于多态性，Object引用可以接收任意类的对象），比较的就是是否是同一个对象，即地址是否相同。**



由于任何两个对象都可以进行比较，那么就会失去了意义，所以一般情况下，都需要**覆写equals方法**。覆盖equals方法，利用类的**特有属性**作为判断对象是否相同的依据。注意：覆写equals方法的时候一般需要覆写**hashCode（）方法，以维护hashCode的常规协定：相等对象必须有相等的哈希码（哈希码）。**

1. **toString（）方法**：返回一个指定的**字符串**。



Object类的toString方法返回的字符串为 **类名@哈希值的十六进制形式**



建议所有子类都覆写此方法。

1. **hashCode()方法**：返回该对象的**哈希值（int类型）**。



1. **getClass()方法**：返回当前类的**运行时类，即此类的字节码文件对象（Class类的对象）**。用于**反射机制**。



**注意**：**一个类的Class对象**只有一个，且不需要我们创建，Java虚拟机会自动创建。**调用getClass方法**只是把这个对象取出来，而非创建。一个类有很多对象，而这些对象通过**getClass方法**获取的是**同一个Class对象**。一个类只有唯一的一个字节码文件对象，通过**三种方式获取的字节码文件对象**都是一样的。

**如：**Class clazz1 ="nihao".getClass();

Class clazz2 = "iloveyou".getClass();

System.out.println(clazz1 == clazz2);//**返回结果是true**

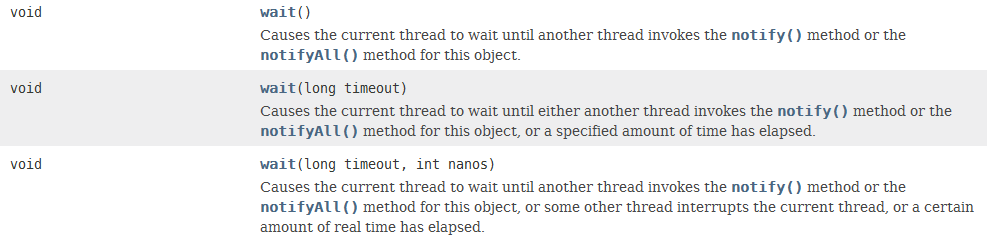
**获取后的Class对象如何使用请见Class类介绍。**

1. **notify() 和 notifyAll() 方法**：在**多线程**中会使用到。





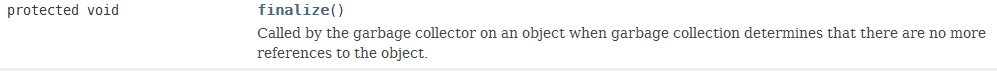
1. **wait()方法**：**3个重载方法**。在**多线程**中会使用到。



1. **clone（）方法：克隆方法。protected修饰的。**



1. **finalize（）方法：protected修饰的。**



1. **覆写Object类的equals方法要抓住两点**：
2. **类型是否相同**，即传进来的对象是否是当前类的一个实例，利用instanceof 运算符来判断，如果不是，最好是**抛出异常**，因为返回false造成不清楚，到底是同类型不相等呢还是不同类呢。
3. **成员变量的值是否相同：根据此类的特有属性值对两个对象进行比较是否相同。**
4. Object类中的**equals方法**和**双等号==**功能一样，Object 中的equals方法本质上就是**双等号==**。双等号==对于引用来说，判断的始终都是**引用的地址是否相同**。
5. **Equals 方法的性质**：
6. **reflexive（反身性）**：任何非空类引用x.equals(x)返回值是true；
7. **symmetric（对称性）**：任何非空类引用x.equals(y) 与x.equals(x)返回值一样；
8. **transitive（传递性）**：任何非空类引用x,y,z 如果x与y、y与z返回结果为true，则x与z的结果也为true。
9. **consistent（一致性）**：多次调用x.equals(y) 返回结果都相同；
10. 任何一个**非空类引用**与null判断，返回结果都为false；
11. **双等号==的判断：**
12. **对于原生数据类型来说，比较的是等号左右两端的值是否相等；**
13. **对于引用类型来说，比较的是左右两端的引用是否指向同一对象或者说左右两端的引用地址是否相同。**
14. 任何一个类都有**toString方法**，因为Object类就有此方法，子类的toString方法要么是自己覆写的，要么是继承父类的或者直接从Object继承的。**直接输出一个对象引用名时，其实是默认调用此对象的toString（）方法**。System.out.println(p); 相当于System.out.println(p.toString());只不过是省略了调用toString（）而已。
15. **instanceof** 运算符 ：使用方法： **boolean result = 对象名instanceof 类名**

返回类型为布尔类型，object为**任意对象表达式**，class是**任意类**；

如果object是class的一个实例，则instanceof 运算符返回true，否则（不是指定类的一个实例或者是null），返回false。如 str instanceof String