修饰符主要分为以下两类：

1. 访问修饰符
2. 非访问修饰符

**访问控制修饰符**

Java中，可以使用访问控制符来保护对类、变量、方法和构造方法的访问。  
 Java 支持 4 种不同的访问权限。

1. default (即默认，什么也不写）: 在同一包内可见，不使用任何修饰符。

使用对象： 类、接口、变量、方法。

不同包： 可以访问同包内的基类 protected 方法，不同包的不能访问

1. private : 在同一类内可见。使用对象：变量、方法。 注意：不能修饰类（外部类） 同一包内也访问不到
2. public : 对所有类可见。使用对象：类、接口、变量、方法
3. protected : 对同一包内的类和所有子类可见。使用对象：变量、方法。 注意：不能修饰类（外部类）。

protected的可见性在于两点：

1. 基类的 protected 成员是包内可见的，并且对子类可见；
2. 若子类与基类不在同一包中，那么在子类中，子类实例可以访问其从基类继承而来的protected方法，而不能访问基类实例的protected方法。

例：



1. obj是实例化的父类，父的clone方法可见是自身跟子类，此操作属于在类外访问自身的protected方法
2. tobj是实例化的子类，父子不同包,子类实例可以访问从父类继承来的protect方法。

非访问修饰符

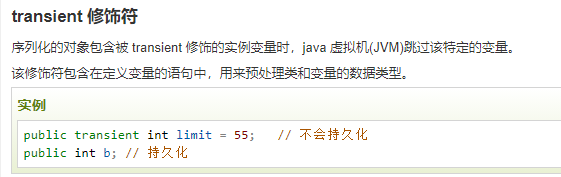
1. static 修饰符，用来修饰类方法和类变量。
2. final 修饰符，用来修饰类、方法和变量，final 修饰的类不能够被继承，修饰的方法不能被继承类重新定义，修饰的变量为常量，是不可修改的。
3. abstract 修饰符，用来创建抽象类和抽象方法。
4. synchronized 和 volatile 修饰符，主要用于线程的编程。

**synchronized**

synchronized 关键字声明的方法同一时间只能被一个线程访问

synchronized 修饰符可以应用于四个访问修饰符。

**transient**



**volatile**

volatile 修饰的成员变量在每次被线程访问时，都强制从共享内存中重新读取该成员变量的值。

当成员变量发生变化时，会强制线程将变化值回写到共享内存。

这样在任何时刻，两个不同的线程总是看到某个成员变量的同一个值。

一个 volatile 对象引用可能是 null。

