[Java内部类详解 - Matrix海子 - 博客园 (cnblogs.com)](https://www.cnblogs.com/dolphin0520/p/3811445.html)

[**Java中内部类到底有什么用？ (juejin.cn)**](https://juejin.cn/post/6844903859186319367)

[(1条消息) Java中静态内部类和非静态内部类有什么区别？\_vcliy的博客-CSDN博客\_静态内部类](https://blog.csdn.net/vcliy/article/details/85235363)

[(1条消息) 浅谈java内部类以及注意点\_weixin\_43243916的博客-CSDN博客\_java内部类的注意](https://blog.csdn.net/weixin_43243916/article/details/96706825?utm_term=java%E5%86%85%E9%83%A8%E7%B1%BB%E7%94%9F%E5%91%BD%E5%91%A8%E6%9C%9F&utm_medium=distribute.pc_aggpage_search_result.none-task-blog-2~all~sobaiduweb~default-4-96706825&spm=3001.4430)

[(1条消息) 内部类的对象生命周期和static的作用\_Thinkingcao的专栏-CSDN博客](https://blog.csdn.net/Thinkingcao/article/details/69950761)

[(1条消息) 内部类详解————局部内部类\_Morty的技术乐园-CSDN博客\_局部内部类](https://blog.csdn.net/u014745069/article/details/80210288)

**普通内部类持有一个指向外部类的引用。要创建普通内部类，一定要先创建外部类。**

**static内部类是内部类中一个比较特殊的情况，Java文档中是这样描述static内部类的：一旦内部类使用static修饰，那么此时这个内部类就升级为顶级类。**

成员内部类的实例几种创建方式，以及原因解释：

一、将创建方法交给外部类

1.非static修饰的内部类。

1、通过外部类非静态方法进行获取内部类实例，直接返回new XXX().

2、通过外部类静态方法进行获取内部类实例，返回new 外部类().new 内部类()。

3、原因：内部类需要持有一个外部类的引用，所以需要先获取外部类引用，（也就是需要知道具体创建内部类的外部类对象）。第一种方式，非静态方法，外部类必须要创建实例才能使用该方法，所以一是为了保证外部类对象的一致性，二是创建时已经确定了没有必要。

第二种方式，静态方法，Java建议通过静态方式访问静态资源，所以默认不会去创建外部类，那么就没有外部类引用，所以，我们需要创建一个外部类实例再创建内部类实例。（那么如果我们通过非静态方式进行访问，这个内部类实例的引用是哪个呢？new 外部类().new 内部类()，是这个）

2.static修饰的内部类。

1、无论是静态方法还是非静态方法都只需要new 内部类（）.

2、原因：此时该类已经升级为顶级类了，不算纯粹的内部类，所以当普通类进行创建即可。

3、static修饰内部类，对于内外两者的更多作用是用来标明静态属性、方法之间的访问逻辑。

二、将创建方法交给内部类

1、非静态内部类需要外部类引用且不可定义静态方法，故创建内部类对象需要先创建内部类对象。矛盾，费事。故无意义。

2、非静态方法同上原因无意义。静态内部类已经相当于顶级类，所以定义静态方法同普通类一致。

三、局部内部类：与普通内部类定义形式一致，而局部内部类相当于定义局部变量。局部内部类类似方法的局部变量，所以在类外或者类的其他方法中不能访问这个内部类，但这并不代表局部内部类的实例和定义了它的方法中的局部变量具有相同的生命周期。我们将局部内部类In定义在了一个if条件作用域中，因此，在if之外的部分，尽管没有离开function()方法，就会报编译错误，无法访问到In这个内部类，因此，还是需要广义的理解局部内部类的含义和访问条件，不可片面的理解为仅仅是方法中定义的内部类。

四、局部内部类访问局部变量必须是final,因为访问的方式是通过将局部变量拷贝为内部类属性，所以必须要final，不然一旦被修改就会造成数据不一致。‘

局部内部类访问外部类的属性不用是final，因为它访问的方式是通过外部类的引用，所有的对象都是持有相同的引用，所以不用final。

五、[静态内部类实现的单例模式是线程安全的 - little\_mao - 博客园 (cnblogs.com)](https://www.cnblogs.com/little-mao2020/p/14673899.html)

**成员内部类：**

**1、非静态：定义逻辑-相当于定义了一个普通的类作为属性，故只能通过外部类访问内部类。实际逻辑是：内部类的定义模块嵌入了外部类中，内部类实例化需要外部实例的引用，故需要先实例化外部类，再通过类指针到嵌入代码处执行实例化内部类代码并返回。一个外部类实例可以创建多个内部类实例，一个内部类实例只有一个外部实例。非静态内部类不能定义静态属性和方法。就像一个寄生母虫一样，外部类不能对它做啥，但是内部想做啥做啥。**

**2、静态：定义了一个普通的类，这个类是独立，不会随着外部类的加载而加载，只有当需要有人new它的时候才会被加载，这个类可以定义任何类型的方法和属性。它的实例化不依赖外部类。它只能访问外部类的静态方法和属性，不能肆意妄为。**

**总结下：静态与非静态主要两个区别，一是加载时是否随着外部类的加载；二是对外部类的访问权限。**

**局部内部类：**

**1、非匿名内部类：内嵌到外部类的方法代码块中，需要实例化且只能在其作用域内进行实例化，相当于一个局部变量，可以访问外部。它的访问权限受限于定义于静态方法中还是非静态。因为它持有一个外部实例的引用，当调用这个方法的时候，就会赋给它这个引用，静态方法中是不会赋给它的。**

**2、匿名内部类：与上面差别不大，主要就是这个类是只能实例化一次，因为没有显式化的定义，这意味着该类的对象的回收并不明显。**