Static subclass vs non-static subclass

### 区别

1. **访问权限的区别1**

Non-static subclass里不能有static variable，比如说对于node class，是不可以把data和next设成static的，不然编译器会报错的。但是static subclass里能申明static variables

1. **访问权限的区别2**

non-static subclass可以访问其外面的class的variable和methods，即使外面的那个variable或者method被设置成private！！！这个我们很少见，你可以自己尝试一下哦。但是，static subclass只能访问外部class里面的static variable和method（注意这里是从内部的class访问外面，而不是从外面访问里面）

1. **initialize的区别**

non-static subclass和static-subclass如果要在你的包含main方法的class里面initialize的话，它们的initialize的方式完全不同哦，我用上次写的linkedlist来举个例子：

上次我们的linkedlist里面我们写的static class Node，假设我们想要initialize的话（不是在linkedlist那个class里写main method，而是在外面新建class写main method，我们需要只要写：

Node n1 = new LinkedList.Node();

注意看这里我们的写法，Node是static class，所以左边边标签直接写出来，然而右边我们是通过LinkedList.Node()来找到这个class的具体位置的，上面这个是static

**Generally, 建立一个static subclass的对象的方法：**

StaticInnerClass sic= new OuterClass.StaticInnerClass();

如果我们把Node class改成non-static，那么我们想要在这个main method里新建一个Node需要如下步骤：

LinkedList list = new LinkedList();

LinkedList.Node n1 = list.new Node();

我们需要先建立LinkedList的对象，然后再通过LinkedList的对象去新建一个Node

**Generally, 建立一个non-static subclass的对象的方法**：

OuterClass oc = new OuterClass();

OuterClass.InnerClass ic = oc.new InnerClass();

### 总结：

1. static class只能是subclass，外部class不能作为static class
2. non-static subclass可以在这个subclass里面直接访问所有关于外部class的所有variable和method，包括private，可以看成外部的class和内部的non-static subclass的关系如同父亲和亲儿子，父亲可以给儿子共享所有的父亲拥有的内容，包括private，所以如果在main里想要建立一个亲儿子的对象，得先问父亲同不同意，先建立父亲的对象，才能通过父亲的对象建立儿子的对象，而且儿子的type并非儿子，而是父亲.儿子（父亲.儿子 儿子1 = 父亲.new 儿子(); 这样建立对象，左边是从属关系，右边体现的是儿子出不出生得经过父亲同意）
3. static subclass只能访问外部class里面的static内容，外部的如果没写static或者设置成了private，那么内部的static class就无法访问那些内容了。可以看成static-subclass和外部的class关系是公司里的上下级，内部的下级只能看上级的公开资料，但是关系并不亲密，如果要从main method新建一个static-subclass的对象时，就是（下级 下级1 = new 上级.下级();这样建立对象，左边是下级本身的标签，右边是想要新建下级，得新分配一个上下级的关系）

### 测试代码

public class OuterClass {

public class InnerClass{}

public static class StaticInnerClass{}

public static void main(String[] args) {

OuterClass oc = new OuterClass();

InnerClass ic=oc.new InnerClass();

StaticInnerClass sic=new OuterClass.StaticInnerClass();

}

}

### 补充：关于为什么要有static-subclass：

如果内部类与外部类关系不紧密，不需要访问外部类的所有属性或方法，那么设计成静态内部类。而且，由于静态内部类与外部类并不会保存相互之间的引用，因此在一定程度上，还会节省那么一点内存资源。因此，静态内部类，和外部类没有什么“强依赖”上的关系，也就是说，没有外部类提供一些属性，内部静态类也可以创建并初始化。

既然上面已经说了什么时候应该用静态内部类，那么如果需求不符合静态内部类所提供的一切好处，那就应该考虑使用内部类了。最大的特点就是：你在内部类中需要访问有关外部类的所有属性及方法。