静态内部类的作用只是为了降低包的深度，方便类的使用，静态内部类适用于包含类当中，但又**不依赖与外在的类**，不用使用外在类的非静态属性和方法，只是为了方便管理类结构而定义。在创建静态内部类的时候，不需要外部类对象的引用。

内部类一个很大的特点就是可以自由的使用外部类的所有变量的方法。

# 静态嵌套类

由于是static的，意味着：

1. 要创建静态嵌套类的对象，并不需要其外围类的对象；
2. 不能够从静态内部类的对象中访问外部类的非静态成员。

而且静态内部类可以声明静态成员。

public class **Invoice** {

private static class **Item**{

String description;

int quantity;

}

public static void **main**(String[] args){

Invoice.Item item = new Invoice.Item(); *//实例化一个静态内部类，不需要有外部类的实例*

}

}

Item类定义为private，所以只有Invoice才能够访问到它，所以不需要操心Item内部的实例变量改为private。

# 内部类

内部类不能声明静态成员，如果内部类有静态成员，那么static的含义将变得模糊，将意味着虚拟机中只有一个实例，还是每个外部对象只有一个实例？

public class **Network** {

public class **Member**{

private String name;

private ArrayList<Member> friends;

public **Member**(String name){

this.name = name;

friends = new ArrayList<>();

}

public void **leave**(){

members.**remove**(this); *//访问了外部的实例变量*

}

}

private ArrayList<Member> members;

public Member **enroll**(String name){

Member newMember = new **Member**(name); //这里可以是因为enroll不是static的

members.**add**(newMember);

return newMember;

}

public static void **main**(String[] args){

Network network = new **Network**();

Network.Member fred = network.**enroll**("fred");

Network.Member tom = network.new **Member**("tom"); *//实例化一个内部类需要先生成一个外部类的实例对象，通过外部类的实例对象生成内部类对象*

}

}