一、概念

1、编译时多态 ----》设计时多态，方法重载

2、运行时多态 ----》 程序运行时动态觉得调用那个方法

3、实现关键：满足继承关系、父类引用指向子类对象，只支持单继承

4、向上转型：隐式转型，自动转型

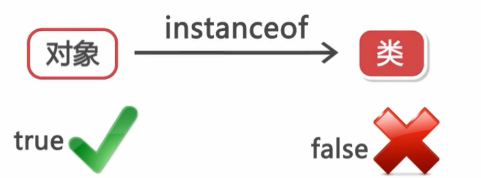
父类引用指向子类1实例，可以调用子类重写父类的方法及父类派生的方法，无法调用子类独有方法

小类转型为大类

6、向下转型：强制类型转换

子类引用指向父类对象，此处必须进行强转，可以调用子类特有的方法

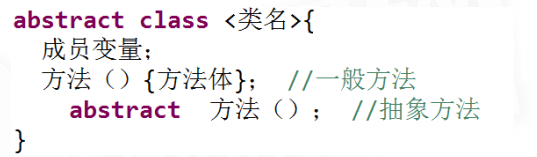
7、instanceof：判断对象是否是继承的



8、static不重写，为本身特有的方法

9、抽象类：abstract，不允许实例化，可以向上转型，指向子类实例，提供抽象方法被不同子类实现

1）抽象类和抽象方法声明格式

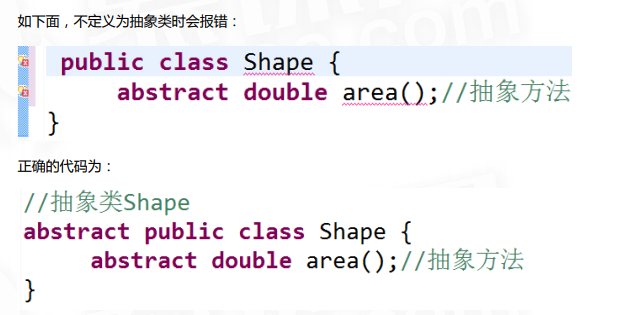


2）特点：不允许直接实例化，只有声明不能有实现

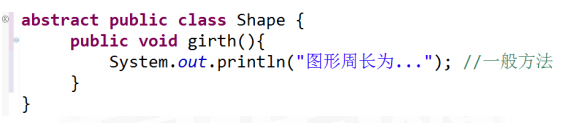
3）意义：为子类提供公共类型、分装子类中重复内容（成员变量和方法）、避免无意义父类实例化

4）注意点

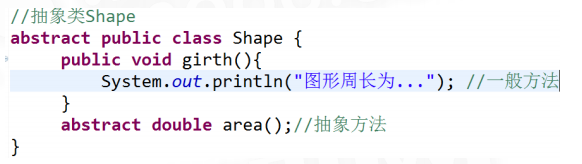
a）含抽象发放的类只能被定义成抽象类



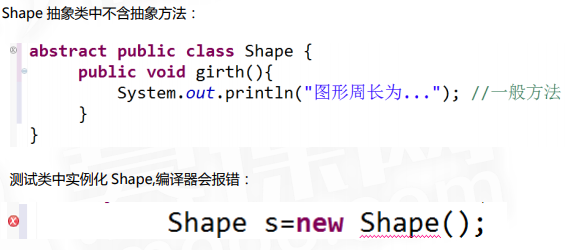
b）抽象类不一定包含抽象方法



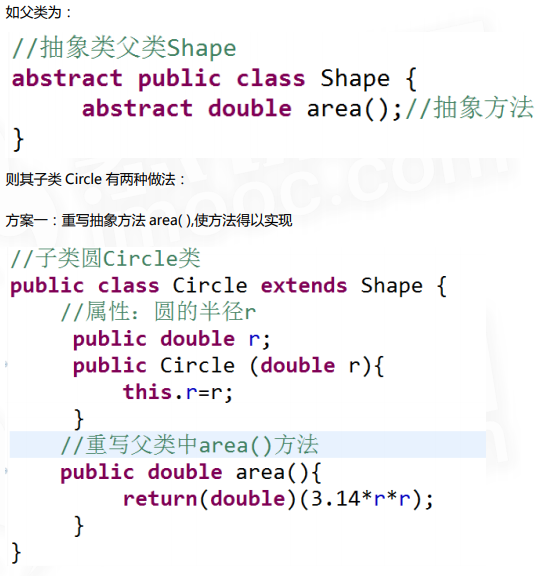
c）在抽象类中的成员方法可以包括一般方法和抽象方法



d）抽象类不能被实例化

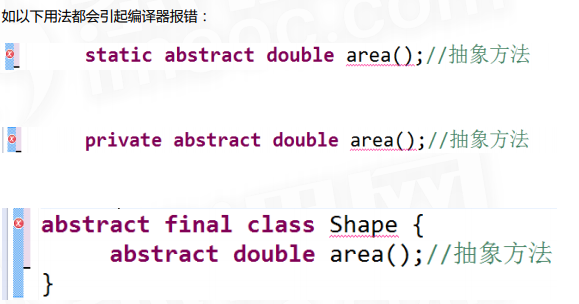


e）继承抽象类必须实现该方法



f）父类具体的，子类也可以是抽象的

g）abstract方法不能用static和private修饰；对于类不能同时使用final和abstract修饰



10、接口：interface ----》implements

1）定义某一批类所需要遵守的规范

2）不关心这些类的内部数据，也不关心这些类中方法实现细节，只规定这些类里必须提供某些方法

3）接口访问修饰符：public 默认

4）接口中抽象方法可以不写abstract关键词

5）访问修饰符为public

6）当实现接口时，需要实现接口中所有的抽象方法，否则设置为抽象类

7）可以设置常量，默认为public static final

8）jdk1.8之后可添加的方法

default：可以在类中重写，并可以通过接口引用调用

static：不可在实现类中重写，可以通过接口名调用

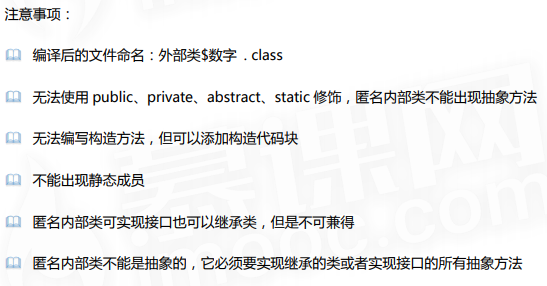
调用默认方法 ----》 接口名.super.方法名



9）可以继承多个，多个中同名方法，实现时必须重写这个同名方法，有同名的必须自己重新实现

10）可以实现多继承，单都要实现

11、匿名内部类：没有名字的内部类，用来简化代码编写



1、成员类：

普通内部类

a）获取内部类实例：new 外部类.new 内部类 外部类对象.new 内部类 外部类对象.获取方法

b）内部类在外部使用时无法直接实例化，需要借由外部类信息才能完成实例化

c）内部类方法修饰符，可以任意，但访问范围会影响

d）内部类可以直接访问外部类成员，如果出现同名属性，优先访问内部类中定义的

e）可以使用外部类.this**.**成员方法，访问外部类中同名的信息

f）外部类访问内部类信息，需要通过内部类实例，无法直接访问

g）内部类编译后.class文件命名：外部类$内部类.class

静态内部类

a）静态内部类，之内访问外部类的静态方法，如果调用非静态成员，可以通过对象实例

b）静态内部类对象实例时，可以不依赖外部对象

c）可以通过外部类.内部类。静态成员的方式，访问内部类中的静态成员

d）当内部类属性与外部类属性同名时，默认调用内部类中的成员，

如果需要访问外部类中的静态属性，可以通过外部类.属性的方式

如果需要访问外部类中的非静态属性，可以通过 new 外部类().属性 的方式

方法内部类 ----》局部内部类

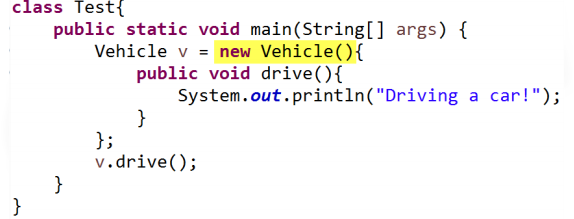
a）定义在方法内部，作用范围在方法内

b）和方法内部成员使用规则一样，class前面不可以加public、private、protected、static

c）类中不能包含静态成员

d）类中可以包含final、abstract修饰的成员

匿名内部类



a）内部没有类型名称、实例对象名称

b）编译后的文件命名：外部类.$数字.class

c）无法使用private、public、protected、abstract、static修饰

d）无法编写构造方法，可以添加构造代码块

e）不能出现静态成员

f）可以实现接口也可以继承父类，但不能同时用

适用场景

a）只用到类的一个实例

b）类定义后马上用到

c）给类命名并不会导致代码更容易被理解