

继 承

生活中的继承

- 孩子像爸妈
- 富二代
- 师徒关系

程序中的继承

- 表示私有

Cat

- name:String
- month:int
- species:String
- weight:double

+ 表示公有

+ eat():void
+ getName():String
+ getMonth():int
+ getSpecies():String
+ getWeight():double
+ run():void
+ Cat()

Dog

- name:String
- month:int
- species:String
- sex:String

+ eat():void
+ getName():String
+ getMonth():int
+ getSpecies():String
+ getSex():String
+ sleep():void
+ Dog()

如果后续有大量类似动物产生，会产生大量重复代码，怎么破？

程序中的继承

Animal

- name:String
- month:int
- species:String

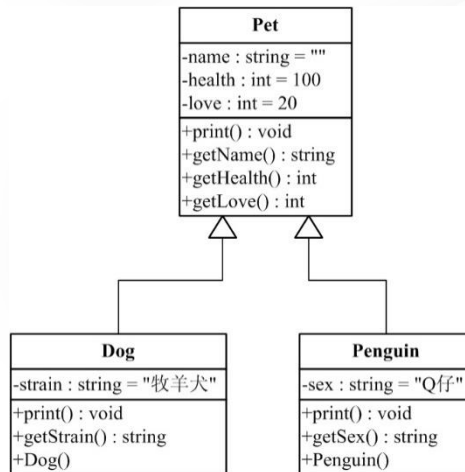
- + eat():void
- + getName():String
- + getMonth():int
- + getSpecies():String
- + getWeight():double

Cat

- weight:double
- + run():void
- + Cat()

Dog

- sex:String
- + sleep():void
- + Dog()



继承

特点：

- 1、利于代码复
- 2、缩短开发周期

继承

- 一种类与类之间的关系
- 使用已存在的类的定义作为基础建立新类
- 新类的定义可以增加新的数据或新的功能，也可以用父类的功能，但不能选择性地继承父类
- 父类 基类
- 子类 派生类

继承的关系

- 满足 “A is a B ” 的关系就可以形成继承关系
如：

猫、狗是动物 ==》 猫，狗继承自动物

学生、老师是人 ==》 学生，老师继承自人

小轿车、大卡车是汽车 ==》 小轿车，大卡车继承自汽车

实现继承

使用extends实现封装

– 编写父类

```
class Animal{  
    //公共的属性和方法  
}
```

– 编写子类，继承父类

```
class Dog extends Animal{  
    //子类特有的属性和方法  
}
```

```
class Cat extends Animal{  
}
```

只能继承一个父类

方法重写

语法规则：

- 返回值类型
 - 方法名
 - 参数类型、顺序、个数
- 都要与父类继承的方法相同

方法重写 PK 方法重载

- **方法重写：**

- 在满足继承关系的子类中
- 方法名、参数个数、顺序、类型与父类、返回值相同
- 访问修饰符的限定范围大于等于父类方法

- **方法重载：**

- 在同一个类中
- 方法名相同
- 参数个数、顺序、类型不同
- 返回值类型、访问修饰符任意

方法重写存在
属性重写不存在

访问修饰符

- 公有的：public
- 私有的：private
- 受保护的：protected
- 默认

访问修饰符

访问修饰符	本类	同包	子类	其他
private	√			
默认	√	√		
protected	√	√	√	
public	√	√	√	√

**如何区分调用的是继承父类的方法
还是子类自己重写的方法？**

super

- 子类访问父类成员

- 访问父类成员方法

```
super.print();
```

- 访问父类属性

```
super.name;
```

实例化顺序

- 继承后的初始化顺序

父类静态成员

子类静态成员

父类对象构造

（属性（赋值）、构造代码块、构造方法）

子类对象构造

（属性（赋值）、构造代码块、构造方法）

子类对象实例化时，能否选择父类的构造形式（调用指定的构造方法）？

super

- 子类访问父类成员

- 访问父类成员方法

```
super.print();
```

- 访问父类属性

```
super.name;
```

- 访问父类构造方法

```
super();
```

super

- 子类的构造的过程中必须调用其父类的构造方法
- 如果子类的构造方法中没有显示调用父类的构造方法，则系统默认调用父类无参的构造方法
- 如果子类构造方法中既没有显式调用父类的构造方法，而父类又没有无参的构造方法，则编译出错
- 使用super调用父类指定构造方法，必须在子类的构造方法的第一行

super pk this

- **this: 当前类对象的引用**

- 访问当前类的成员方法
- 访问当前类的成员属性
- 访问当前类的构造方法
- 不能在静态方法中使用

- **super: 父类对象的引用**

- 访问父类的成员方法
- 访问父类的成员属性
- 访问父类的构造方法
- 不能在静态方法中使用

25
26
27
28
29
30
31
32
33
34
~

```
public Cat(String name,int month){  
    /* 子类构造默认调用父类无参构造方法  
     * 可以通过super()调用父类允许被访问的其他构造方法  
     * super()必须放在子类构造方法有效代码第一行  
     */  
    this();  
    super(name,month); //this  
  
    System.out.println("我是子类的带参构造方法")  
}
```

- 构造方法调用时，super和this不能同时出现