· 覆写

场景:子类可以继承父类的所有方法，但在有些场景下该方法不合适，因此需要方法的覆写

1. 方法覆写（案例JavaDemo1）
   * 1. 当子类定义了与父类方法名称相同，参数类型及个数不同的时候，我们称为覆写
     2. 子类覆写是为了优化父类方法。
     3. 由于我们现在实例化的是子类对象,所以此时我们调用的是子类覆写的方法，如果子类没有覆写则调用父类方法。
     4. 如果子类覆写了方法则，在子类中想要调用父类中的方法必须使用super关键字 eg: super.connect().
     5. 覆写的意义是保留已有方法的名称，根据不同的情况优化方法。

结论:在子类中调用父类方法，一定要加上super()；

1. 方法覆写限制
   * 1. 虽然利用方法的覆写可以更好的扩充父类方法，但是对于覆写也是有限制要求的----覆写的方法不能有比父类更为严格的访问控制权限
        1. 对于访问控制权限现在有三种

Public > default(不写) > private， private权限是最小的，

对于1）的解释:如果父类的方法使用了 default 那么子类覆写的方法只能是 public 和 default 的访问权限；

补充权限: private 除了可以定义属性 也可以定义方法;

范例:（案例JavaDemo2）

class Channel {

private void connect(){

System.out.println("【Channel】");

}

void fun(){

this.connect();

}

}

class DatabaseChannel extends Channel {

//此时并不是一个覆写，因为父类connect方法对于子类来说是不可见，因此子类的connect方法是一个新的方法

public void connect(){

System.out,println("【子类方法】");

}

}

public class JavaDemo{

public static void main (String args []){

DatabaseChannel channel = new DatabaseChannel();

channel.fun(); //输出一个【channel】

}

}

结论:在以后开发中只要定义方法 95% 都是使用public ，所以覆写的时候也考虑使用public

面试题: 请解释override 与 overloading 的区别。Overload时返回参数是否相同？

|  |  |  |  |
| --- | --- | --- | --- |
| NO | 区别 | override | overloading |
| 1 | 中文含义 | 覆写 | 重载 |
| 2 | 概念 | 方法名称，参数类型，返回值都相同 | 方法名称相同，参数的类型和个数不同 |
| 3 | 权限 | 没有权限限制 | 不能有比父类更加严格的权限控制 |
| 4 | 范围 | 发生在一个类中 | 发生在继承关系类中 |

在返回类型上没有做要求，但我们应该保持方法类型的一致性，方便处理。

1. 属性覆盖
   1. 当子类定义了一个和父类一样的属性，称为属性覆盖。

范例:

|  |
| --- |
| class Channel {  //String info = "ww.baid.com" ;  private String info = "ww.baid.com" ;  public getInfo(){  return this.info ;  }  }  class DatabaseChannel extends Channel {  String info = "www.xinlang.com" ;  public void fun (){  System.out,println(this.info);  System.out.println(super.getInfo);  }  }  public class JavaDemo2{  public static void main (String args []){  DatabaseChannel channel = new DatabaseChannel();  channel.fun();  }  } |

当我们的属性没有进行封装的时候，我们会采用就近原则（调用本类属性），但是一旦进行封装使用private 后， 我们父类的属性实际上和子类就已经没有关系了，即便是名字一样也只是定义了一个新的属性。

面试题: 请解释 super 与 this 的区别

·this 首先查找本类属性或者方法， 如果本类不存在则查找父类定义的，如果使用super 则不查找子类直接查找父类

·this 与 super 都可以进行构造方法的调用，但是this 只能是调用本类构造，而super()则是子类调用父类构造，并且两个都必须放在构造方法的首行，因此不能同时出现。

·this 可以表示当前对象

1. 关键字 final
   1. 在程序之中描述的是终结器的概念，在java中使用final可以实现如下功能:
      1. 定义不能被继承的类，方法，常量；
         1. final class Demo{} //无法被继承
         2. public final void connect(){}; //无法被覆写
         3. public final int ON = 1; // 无法被修改
            1. 一般当我们定义全局常量时会使用

public static final int ON = 1; //在定义全局常量时常量首字母必须大写