**为什么要在Object类定义一个无参构造方法?**

Object是所有类的父类,在实例化子类的时候,会默认调用父类

的无参构造

简单来讲,一个简单java类应该覆写Object类中的三个方法:

1. 取得对象信息:public String toString();
2. 对象内容比较是否相等:public boolean equals(Object obj)
3. 取得对象HASH码:public int hashCode();

**范例:内容比较 equals 覆写**

|  |
| --- |
| public class test{  public static void main(String args[]) throws Exception {  Book b1 = new Book("java开发",34.5);  Book b2 = new Book("java开发",34.5);  System.out.print(b1.equals(b2));  System.out.print(b1.equals("hello,world"));  }  }  class Book{  private String title;  private double price;  public Book(String title,double price){  this.title = title;  this.price = price;  }  public boolean equals(Object obj){  if(this == obj){ //自己和自己比较  return true;  }  if(obj == null){  return false;  }  if(!(obj instanceof Book)){ //不是本类实例  return false;  }  Book book = (Book) obj;  if(this.title.equals(book.title) &&  this.price == book.price){  return true;  }  return false;  }  } |

Object是所有类的父类,所以可以接受所有类的对象,连数组对象也可以接受

**方法:**

|  |  |
| --- | --- |
| 析构函数 | protected void finalize()  throws Throwable |
| 对象克隆(浅拷贝) | protected Object clone()  throws **CloneNotSupportedException** |

范例:clone

|  |
| --- |
| public class test{  public static void main(String args[]) throws Exception{  Book b1 = new Book("java开发",34.5);  Book b2 = (Book)b1.clone();  System.out.println(b1);  b1.setTitle("C++开发");  System.out.println(b1);  System.out.println(b2);  }  }  class Book implements Cloneable{ //表示此类对象可以被克隆  private String title;  private double price;  public Book(String title, double price) {  this.title = title;  this.price = price;  }  public void setTitle(String title) {  this.title = title;  }  @Override  public String toString() {  return "书名:" + this.title + ",价格:" + this.price;  }  @Override //由于clone()是protected的,在外面无法调用,所以重写此方法,改为public  public Object clone() throws CloneNotSupportedException {  return super.clone();  }  } |