对象序列化:保存在内存中的对象数据转化为二进制数据

进行传输的操作,不是所有类的对象都可以进行序列化,要想实现序列化

必须实现java.io.Serializable接口,这个接口没有任何方法,它只是一个标识接口,表示一种能力

序列化:将内存对象序列化到文件里

反序列化:将序列化的文件反序列化到程序中

class Book implements Serializable{  
 private String title;  
 private double price;  
  
 public Book(String title, double price) {  
 this.title = title;  
 this.price = price;  
 }  
 @Override  
 public String toString() {  
 return "Book{" +  
 "title='" + title + '\'' +  
 ", price=" + price +  
 '}';  
 }  
}

序列化类:java.io.ObjectOutputStream,将对象变为了指定格式的二进制数据

反序列化类:java.io.ObjectInputStream,将序列化的对象转化为对象内容

## ObjectOutputStream

java.lang.Object

java.io.OutputStream

java.io.ObjectOutputStream

|  |  |
| --- | --- |
| 构造函数 | public ObjectOutputStream(OutputStream out) throws IOException |
| 将对象序列化输出到指定的文件 | public final void writeObject(Object obj)  throws IOException |

范例:将对象序列化到指定文件中

class Book implements Serializable{  
 private String title;  
 private double price;  
  
 public Book(String title, double price) {  
 this.title = title;  
 this.price = price;  
 }  
 @Override  
 public String toString() {  
 return "Book{" +  
 "title='" + title + '\'' +  
 ", price=" + price +  
 '}';  
 }  
}  
  
public class test{  
 public static void main(String args[]) throws Exception {  
 *ser*();  
 }  
 public static void ser() throws Exception{  
 ObjectOutputStream oss = new ObjectOutputStream(new FileOutputStream(new File("F:\\demo\\01.txt")));  
 oss.writeObject(new Book("java开发",34.5));  
 oss.close();  
 }  
}

## ObjectInputStream

java.lang.Object

java.io.InputStream

java.io.ObjectInputStream

|  |  |
| --- | --- |
| 构造函数 | public ObjectInputStream(InputStream in)  throws IOException |
| 读取方法 | public final Object readObject()  throws IOException,ClassNotFoundException |

class Book implements Serializable{  
 private String title;  
 private double price;  
  
 public Book(String title, double price) {  
 this.title = title;  
 this.price = price;  
 }  
 @Override  
 public String toString() {  
 return "Book{" +  
 "title='" + title + '\'' +  
 ", price=" + price +  
 '}';  
 }  
}  
  
public class test{  
 public static void main(String args[]) throws Exception {  
// ser();  
 *dser*();  
 }  
 public static void ser() throws Exception{ //序列化函数  
 ObjectOutputStream oss = new ObjectOutputStream(new FileOutputStream(new File("F:\\demo\\01.txt")));  
 oss.writeObject(new Book("java开发",34.5));  
 oss.close();  
 }  
 public static void dser() throws Exception{  
 ObjectInputStream ois = new ObjectInputStream(new FileInputStream(new File("F:\\demo\\01.txt")));  
 Object obj = ois.readObject();  
 Book book = (Book)obj;  
 System.*out*.print(book);  
 }  
  
}

## transient关键字

序列化默认会将对象的所有的属性序列化保存,如果

某个属性不需要被保存，就在在属性前加上transient关键字

Private transient String title;