目录

[Day13.    Java](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500650)

[练习1   学生对应成绩](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500651)

[1     io](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500652)

[2     File](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500653)

[2.1       文件、文件夹属性方法](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500654)

[2.2       创建、删除](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500655)

[2.3       目录列表](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500656)

[练习2  File](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500657)

[练习3   求文件夹的总大小](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500658)

[3     流 Stream](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500659)

[4     InputStream,OutputStream](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500660)

[5     FileInputStream,FileOutputStream](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500661)

[练习4   文件流](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500662)

[练习5   文件复制](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500663)

[6     ObjectInputStream，ObjectOutputStream](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500664)

[6.1   Serializable 接口](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500665)

[6.2       不序列化的变量](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500666)

[6.3       序列化版本](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500667)

[练习6   序列化](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500668)

[7     作业](http://code.tarena.com.cn/CGBCode/cgb1903/01-java/%b1%ca%bc%c7/day13.htm#_Toc6500669)

**Day13.  Java**

**练习1  学生对应成绩**

学生对象 -- 98

项目：dy1301\_学生对应成绩

类：day1301.Test1

             Student

Student

**package** day1301;

**public** **class** Student {

**private** **int** id;

**private** String name;

**private** String gender;

**private** **int** age;

**public** Student() {

**super**();

}

**public** Student(**int** id, String name, String gender, **int** age) {

**super**();

**this**.id = id;

**this**.name = name;

**this**.gender = gender;

**this**.age = age;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getGender() {

**return** gender;

}

**public** **void** setGender(String gender) {

**this**.gender = gender;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

@Override

**public** String toString() {

**return** "Student [id=" + id + ", name=" + name + ", gender=" + gender + ", age=" + age + "]";

}

@Override

**public** **int** hashCode() {

**final** **int** prime = 31;

**int** result = 1;

    result = prime \* result + age;

    result = prime \* result + ((gender == **null**) ? 0 : gender.hashCode());

    result = prime \* result + id;

    result = prime \* result + ((name == **null**) ? 0 : name.hashCode());

**return** result;

}

@Override

**public** **boolean** equals(Object obj) {

**if** (**this** == obj)

**return** **true**;

**if** (obj == **null**)

**return** **false**;

**if** (getClass() != obj.getClass())

**return** **false**;

    Student other = (Student) obj;

**if** (age != other.age)

**return** **false**;

**if** (gender == **null**) {

**if** (other.gender != **null**)

**return** **false**;

    } **else** **if** (!gender.equals(other.gender))

**return** **false**;

**if** (id != other.id)

**return** **false**;

**if** (name == **null**) {

**if** (other.name != **null**)

**return** **false**;

    } **else** **if** (!name.equals(other.name))

**return** **false**;

**return** **true**;

}

}

Test1

**package** day1301;

**import** java.util.HashMap;

**public** **class** Test1 {

**public** **static** **void** main(String[] args) {

    Student s1 = **new** Student(9527,"唐伯虎","男",19);

    Student s2 = **new** Student(9527,"唐伯虎","男",19);

    //哈希值相同，才能保证计算出相同下标位置

    System.***out***.println(s1.hashCode());

    System.***out***.println(s2.hashCode());

    //即使计算出相同下标，equals()也必须相等，才能覆盖

    System.***out***.println(s1.equals(s2));

    HashMap<Student, Integer> map = **new** HashMap<>();

**for** (**int** i = 0; i < 20; i++) {

        map.put(**new** Student(), 99);

    }

    System.***out***.println(map);

}

}

**1   io**

Input / Ouput

输入 / 输出

java.io 包

File

InputStream/OutputStream

FileInputStream/FileOutputStream

ObjectinputStream/ObjectOutputStream

Reader/Writer

InputStreamReader/OutputStreamWriter

BufferedReader

PrintWriter、PrintStream

**2   File**

封装一个磁盘路径字符串，提供了一组对文件、文件夹操作的方法

可以封装文件路径、文件夹路径、不存在的路径

{path="d:/abc"}

{path="d:/a.txt"}

{path="f:/xx/xxx/xx"}

**2.1     文件、文件夹属性方法**

getName()

文件名

getParent()

父目录

getAbsolutePath()

完整路径，绝对路径

length()

文件的字节量，对文件夹无效

isFile()

是否是文件

isDirectory()

是否是文件夹

**2.2     创建、删除**

createNewFile()

新建文件

文件已经存在，不会再新建，返回 false

所在的文件夹不存在，会出现异常

mkdirs()

逐层创建多层文件夹

delete()

删除文件，或空目录

**2.3     目录列表**

list()

得到 String[]，包含所有的文件名

["a.txt", "b.mp3", "c.jpg"]

listFiles()

得到 File[]，包含所有文件的封装的File对象

[{...}, {...}, {...}]

**练习2  File**

项目：day1302\_File

类：day1302.Test1

**package** day1302;

**import** java.io.File;

**public** **class** Test1 {

**public** **static** **void** main(String[] args) {

    String path;

    //path = "D:\\home\\java\\eclipse\\eclipse.exe";//存在的文件

    //path = "D:/home/java/eclipse";//存在的文件夹

    path = "f:/xxx/xx/xx";//不存在的路径

    File f = **new** File(path);//封装路径

    System.***out***.println(f.getName());//取文件名

    System.***out***.println(f.getParent());//取父目录

    System.***out***.println(f.getAbsolutePath());//取完整路径

    System.***out***.println(f.length());//文件字节量，目录无效

    System.***out***.println(f.isFile());//是否是文件

    System.***out***.println(f.isDirectory());//是否是文件夹

}

}

Test2

**package** day1302;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Scanner;

**public** **class** Test2 {

**public** **static** **void** main(String[] args) **throws** Exception {

    /\*

     \* d:/abc/aa/

     \* d:/abc/aa/f1

     \*

     \* 删除f1

     \* 删除aa文件夹

     \*/

    System.***out***.println("按回车执行");

**new** Scanner(System.***in***).nextLine();

    File dir = **new** File("d:/abc/aa");

    dir.mkdirs();

**new** Scanner(System.***in***).nextLine();

    File file = **new** File(dir, "f1");

    file.createNewFile();

**new** Scanner(System.***in***).nextLine();

    file.delete();

**new** Scanner(System.***in***).nextLine();

    dir.delete();

}

}

Test3

**package** day1302;

**import** java.io.File;

**import** java.util.Scanner;

**public** **class** Test3 {

**public** **static** **void** main(String[] args) {

    System.***out***.println("文件夹：");

    String s = **new** Scanner(System.***in***).nextLine();

    File dir = **new** File(s);

**if** (! dir.isDirectory()) {

        System.***out***.println("请输入正确的文件夹");

**return**;

    }

    /\*

     \* 如果对文件进行列表、

     \* 对不存在的路径列表、

     \* 对无权访问的文件夹列表

     \* 都返回 null 值

     \*/

    String[] names = dir.list();

    File[] files = dir.listFiles();

**if**(names ==**null**) {

        System.***out***.println("无法获得目录列表");

**return**;

    }

**for** (**int** i = 0; i < names.length; i++) {

        System.***out***.println(names[i]);

    }

    System.***out***.println("\n\n-------------------------");

**for** (**int** i = 0; i < files.length; i++) {

        File f = files[i];

        System.***out***.println(

         f.getName()+" - "+f.length());

    }

}

}

**练习3  求文件夹的总大小**

[a]

  |- b

  |- c

  |- [d]

       |- e

       |- [f]

            |- h

            |- i

       |- [g]

            |- j

            |- k

Test4

**package** day1302;

**import** java.io.File;

**import** java.util.Scanner;

**public** **class** Test4 {

**public** **static** **void** main(String[] args) {

    System.***out***.println("文件夹：");

    String s = **new** Scanner(System.***in***).nextLine();

    File dir = **new** File(s);

**if** (! dir.isDirectory()) {

        System.***out***.println("请输入正确的文件夹");

**return**;

    }

**long** size = *dirLength*(dir);

    System.***out***.println(size);

}

**private** **static** **long** dirLength(File dir) {

    //对dir列表

    File[] list = dir.listFiles();

**if** (list == **null**) {//不能获得列表

**return** 0;

    }

**long** sum = 0;

**for** (**int** i = 0; i < list.length; i++) {

        File f = list[i];

**if** (f.isFile()) {//f是文件

           sum += f.length();

        } **else** {//f是文件夹

           sum += *dirLength*(f);//递归求f文件夹大小

        }

    }

**return** sum;

}

}

**3   流 Stream**

数据的读写操作（io操作），抽象成数据在管道中流动

单方向流动

  输入流，只能用来读取数据（读入内存）

  输出流，只能用来输出数据（内存数据向外输出）

只能从头到尾，顺序流动一次，不能反复流动，如果要重复流动，可以重新创建新的流

**4   InputStream,OutputStream**

字节流的抽象父类

方法：

**write(int b)**

只输出int四个字节中，末尾的一个字节值

[1][2][3][4] ---> [4]

**write(byte[], start, length)**

输出byte[] 数组中，从start开始的length个字节值

**read()**

读取一个字节值，补三个0字节，变成int

[4] ---> [1][2][3][4]

读取结束后，再读取，返回 -1

**read(byte[] buff)**

按数组的长度，读取一批字节值，存放到指定的数组中，并返回这一批的字节数量

读取结束后，再读取，返回 -1

**5   FileInputStream,FileOutputStream**

**练习4  文件流**

项目：day1303\_文件流

类：day1303.Test1

**package** day1303;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**public** **class** Test1 {

**public** **static** **void** main(String[] args) **throws** Exception {

    /\*

     \* FOS--d:/abc/f2

     \*

     \* new FileOutputStream(文件)

     \* 不管文件是否存在，都新建空文件

     \*/

    FileOutputStream out =

**new** FileOutputStream("d:/abc/f2");

    out.write(97);//00 00 00 61 --> 61

    out.write(98);//00 00 00 62 --> 62

    out.write(99);//00 00 00 63 --> 63

    out.write(356);//00 00 01 64 --> 64

**byte**[] a = {

           101,102,103,104,105,

           106,107,108,109,110

    };

    out.write(a, 0, 10);

    out.write(a, 3, 4);

    out.close();//释放系统资源

}

}

Test2

**package** day1303;

**import** java.io.FileInputStream;

**public** **class** Test2 {

**public** **static** **void** main(String[] args) **throws** Exception {

    /\*

     \* FIS--f2

     \*/

    FileInputStream in =

**new** FileInputStream("d:/abc/f2");

    //标准格式

**int** b;

**while**((b = in.read()) != -1) {

        System.***out***.println(b);

    }

    in.close();

    /\*int b;

    while(true) {

        b = in.read();

        if(b == -1) {

           break;

        }

        System.out.println(b);

    }\*/

}

}

Test3

**package** day1303;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.util.Arrays;

**public** **class** Test3 {

**public** **static** **void** main(String[] args) **throws** Exception {

    /\*

     \* FIS--f2

     \*/

    FileInputStream in =

**new** FileInputStream("d:/abc/f2");

    //批量读取标准格式，写了删，删了写

**byte**[] buff = **new** **byte**[5];

**int** n;//用来保存每次读取的数量

**while**((n = in.read(buff)) != -1) {

        System.***out***.println(

         n+"个："+Arrays.*toString*(buff));

    }

    in.close();

    /\*while(true) {

        n = in.read(buff);

        if (n == -1) {

           break;

        }

        System.out.println(

         n+"个："+Arrays.toString(buff));

    }\*/

}

}

**练习5  文件复制**

Test4

**package** day1303;

**import** java.io.File;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.util.Scanner;

**public** **class** Test4 {

/\*

 \* 原文件：d:/abc/f2

 \* 目标文件： d:/def/f3

 \*/

**public** **static** **void** main(String[] args) {

    System.***out***.println("原文件：");

    String s1 = **new** Scanner(System.***in***).nextLine();

    File from = **new** File(s1);

**if**(! from.isFile()) {

        System.***out***.println("请输入正确的文件");

**return**;

    }

    System.***out***.println("目标文件：");

    String s2 = **new** Scanner(System.***in***).nextLine();

    File to = **new** File(s2);

**if** (to.isDirectory()) {

        System.***out***.println(

         "请具体到文件，不能是文件夹");

**return**;

    }

**try** {

*copy*(from, to);

        System.***out***.println("完成");

    } **catch** (Exception e) {

        System.***out***.println("失败");

        e.printStackTrace();

    }

}

**private** **static** **void** copy(

        File from, File to) **throws** Exception {

    /\*

     \* 文件输入流插在 from

     \* 文件输出流插在 to

     \* 从头到尾读取 from 原文件数据，

     \* 读取的数据，直接向目标文件 to 输出

     \*/

    FileInputStream in =

**new** FileInputStream(from);

    FileOutputStream out =

**new** FileOutputStream(to);

    /\*int b;

    while((b = in.read()) != -1) {

        out.write(b);

    }\*/

    // 8k - 8192

**byte**[] buff = **new** **byte**[8192];

**int** n;//保存每一批的数量

**while**((n = in.read(buff)) != -1) {

        System.***out***.println(n);

        out.write(buff, 0, n);

    }

    in.close();

    out.close();

}

}

**6   ObjectInputStream，ObjectOutputStream**

对象序列化、反序列化

序列化 把一个对象的信息，按固定的字节格式，变成一串字节序列输出

方法：

**writeObject(Object obj)**

把对象变成一串字节序列输出

**readObject()**

读取序列化数据，反序列化恢复对象

**6.1     Serializable 接口**

被序列化的对象，必须实现 Serializable 接口

**6.2     不序列化的变量**

static - 属于类，不随对象被序列化输出

transient - 临时，只在程序运行期间，在内存中存在，不会被序列化持久保存

**6.3     序列化版本**

控制旧版本的数据，不允许恢复成新版本的类型

如果自己不定义版本，编译器编译时，会根据类的定义信息，计算生成一个版本值

static final long serialVersionUID

**练习6  序列化**

项目：day1304\_序列化

类：day1304.Test1

             Student

Student

**package** day1304;

**import** java.io.Serializable;

/\*

 \* 空接口，

 \* 标识接口，用来标识学生类型实例，允许被序列化

 \*/

**public** **class** Student **implements** Serializable{

**private** **static** **final** **long** ***serialVersionUID*** = 2019L;

**private** **int** id;

**private** String name;

**private** **static** String *gender*;

**private** **transient** **int** age;

**public** Student() {

}

**public** Student(**int** id, String name, String gender, **int** age) {

**this**.id = id;

**this**.name = name;

**this**.*gender* = gender;

**this**.age = age;

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getGender() {

**return** *gender*;

}

**public** **void** setGender(String gender) {

**this**.*gender* = gender;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

@Override

**public** String toString() {

**return** "Student [id=" + id + ", name=" + name + ", gender=" + *gender* + ", age=" + age + "]";

}

}

Test1

**package** day1304;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** java.io.ObjectOutputStream;

**public** **class** Test1 {

**public** **static** **void** main(String[] args) **throws** Exception {

    Student s =

**new** Student(9527,"唐伯虎","男",19);

    /\*

     \* OOS--FOS--f3

     \*/

    ObjectOutputStream out =

**new** ObjectOutputStream(

**new** FileOutputStream("d:/abc/f3"));

    out.writeObject(s);

    out.close();

}

}

Test2

**package** day1304;

**import** java.io.FileInputStream;

**import** java.io.FileNotFoundException;

**import** java.io.IOException;

**import** java.io.ObjectInputStream;

**public** **class** Test2 {

**public** **static** **void** main(String[] args) **throws** Exception {

    /\*

     \* OIS--FIS--f3

     \*/

    ObjectInputStream in =

**new** ObjectInputStream(

**new** FileInputStream("d:/abc/f3"));

    Student s = (Student) in.readObject();

    System.***out***.println(s);

    in.close();

}

}

**7   作业**

  重写

  day1302\_File - Test4

文件夹总大小

  day1303\_文件流 - Test4

文件复制

  在文件夹中查找所有的图片文件，包括深层目录文件

jpg,png,gif,bmp