打印输出流PrintStream和PrintWriter

1. **打印流**：**PrintStream和PrintWriter：存在于java.io包中。**

Print是打印的意思，固然就是输出，即打印流属于**输出流**。（当然就没有PrintReader了）。**打印流**为其他的输出流**添加新的功能**，使被装饰的流能够方便的打印**各种数据值的表示形式**。

1. **PrintStream存在于java.io包中，是OutputStream的子类，FilterOutputStream的直接子类。属于一种字节输出流，也是一种处理流，可以对字节输出流进行装饰。**
2. **PrintStream永远不会抛出IOException。**
3. **打印流作为处理流的功能所在：PrintStream提供了很多的打印方法，可以对多种数据类型值进行打印，并保持数据的表示形式。**
4. **打印流的特点**： 提供了特别多的**ptint方法**，可以对**多种数据类型**打印，并保持数据的表现形式。**原理**：是把数据先直接转成了字符串，然后打印出来。

write在写一个int值时，只保留了最后一个字节，忽略了高三个字节。如：

PrintStream out = new PrintStream("e:\\text.txt");//打印到文件中

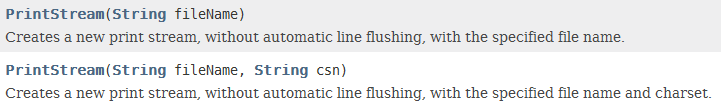
out.println(97);//打印出9和7两个字符，即97

out.write(97);//打印出97对应的字符， a

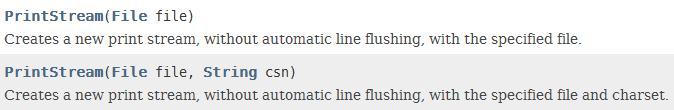
out.write(97+256); //同上，自动忽略256

out.close();

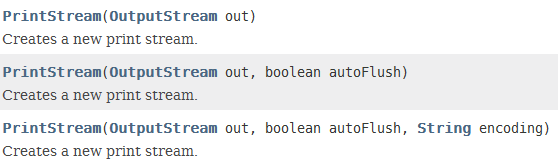
1. PrintStream打印的所有字符都使用**平台默认的编码**转换成字节，如果需要写入的是字符而不是字节，应该使用**PrintWriter类**。
2. PrintStream打印流的**构造方法**：（分为三类：**File对象、输出流、字符串路径**）
3. **字符串路径**；



1. **File对象**；说明PrintStream流可以直接**操作文件**。



1. **字节输出流**：对字节输出流进行装饰，处理一种处理流。



1. FileStream和FileOutputStream的区别：

都可以操作**文件**；但是**FileOutputStream只可以操作文件**，而FileStream还可以对**输出流**进行**封装**。

1. **System类中out属性、err属性**返回的就是**PrintWriter流** ：
   1. **标准输出流和错误标准输出流，两者区别只是err打印出的字符颜色是红色，以示区分，其他的并无差别。**





通常的**System.out.println()、System.err.println()。**

* 1. System.out和System.err，这两个打印输出流默认的都是打印到控制台上，但是可以通过**System的静态方法setOut或setErr**修改打印到的位置，位置需要通过打印流进行包装。





示例：

修改打印的位置到文件中：

File file = **new** File("e:/aaa.txt");

PrintStream ps1 = **new** PrintStream(**new** BufferedOutputStream(

**new** FileOutputStream(file)), **true**);

**System.*setOut*(ps1);**

System.***out***.println("我一定要坚强");//打印到了文件中，而非控制台

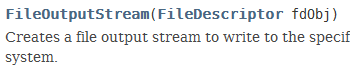
再修改为控制台的方法：

PrintStream ps2 = **new** PrintStream(**new** BufferedOutputStream(**new** **FileOutputStream(FileDescriptor.*out*)**),**true**);

System.*setOut*(ps2);

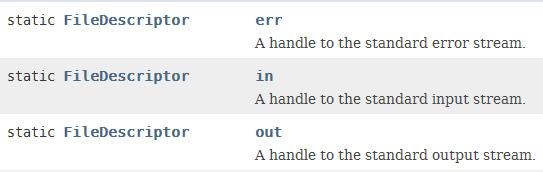
System.***out***.println("我一定要坚强");//**打印到了控制台**

利用了FileOutputStream的构造方法：



1. **System的out、err、in默认情况下的打印位置**：控制台和键盘输入。

**FileDescriptor类**有三个属性值：



**System的初始化**：

FileInputStream fdIn = new FileInputStream(**FileDescriptor.in**);

FileOutputStream fdOut = new FileOutputStream(**FileDescriptor.out**);

FileOutputStream fdErr = new FileOutputStream(**FileDescriptor.err**);

*setIn0*(**new** BufferedInputStream(fdIn));

*setOut0*(*newPrintStream*(fdOut, *props*.getProperty("sun.stdout.encoding")));

*setErr0*(*newPrintStream*(fdErr, *props*.getProperty("sun.stderr.encoding")));

1. System.in 是一个**标准的输入流**，默认是**通过键盘输入**。



也可以通过**System的静态方法setIn**更改它的源头。

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.println("请输入:");

System.***out***.println(scanner.nextLine());

//利用转换流

BufferedReader reader = **new** BufferedReader(**new** InputStreamReader(System.***in***));

System.***out***.println("请输入:");

System.***out***.println(reader.readLine());

//修改标准输入流的源头为文件输入

File file = **new** File("e:/aaa.txt");

BufferedInputStream is = **new** BufferedInputStream(**new** FileInputStream(file));

System.*setIn*(is);

**byte**[] buff = **new** **byte**[1024];

System.***out***.println(System.***in***.read(buff));

System.***out***.println(Arrays.*toString*(buff));

//重新把标准输入流的源头设置为键盘输入

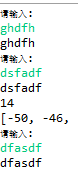
InputStream is2 = **new** FileInputStream(FileDescriptor.***in***);

System.*setIn*(is2);

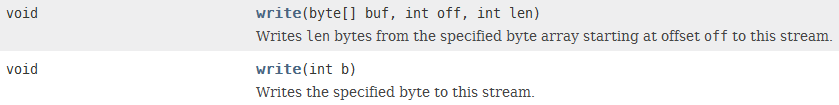
BufferedReader reader2 = **new** BufferedReader(**new** InputStreamReader(System.***in***));

System.***out***.println("请输入:");

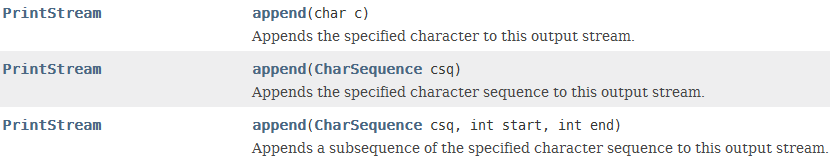
System.***out***.println(reader2.readLine());

结果：

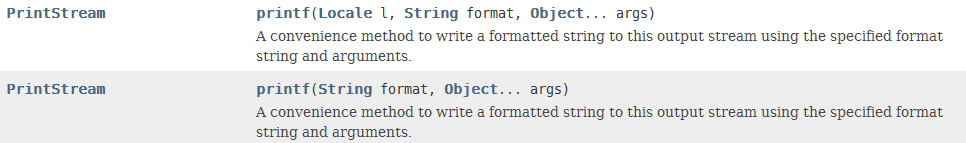
1. **FileStream的成员方法**：**主要有print、println、append、write等。**
2. **print方法**：参数包括八种基本数据类型、字符数组、字符串、Object类型。
3. **write方法**：自动获取int的**低8位，打印输出的是字符，而print方法打印的数据的表示形式。**



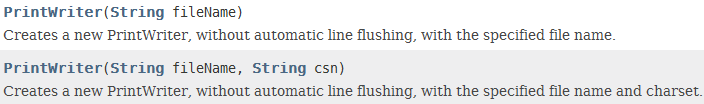
1. **println方法**和print方法接收的参数完全相同，只不过是print在最后会**自动换行**。
2. **append方法**：返回一个PrintWriter对象。

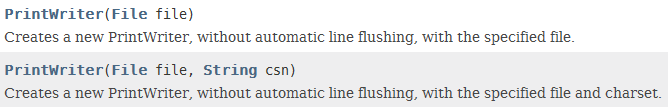


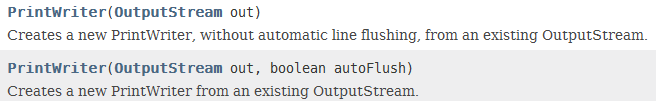
1. **printf方法：**返回一个PrintWriter对象。



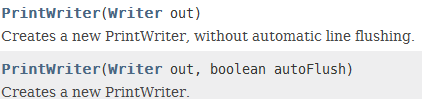
1. **PrintWriter打印流**：存在于java.io包中，直接父类是**java.io.Writer**。其作用和PrintStream相似，只不过PrintWriter直接操作字符。
2. **PrintWriter打印流的构造方法：**分为四类：**字符串路径、File对象、字节输出流、字符输出流。**
3. **字符串路径：**



1. **File对象：**
2. **字节输出流：（可以使其具有自动刷新功能）**



1. **字符输出流：（可以使其具有自动刷新功能）**



1. PrintWriter类的构造方法比PrintStream多1种，可以接受Writer对象，可以对**OutputStream和Writer**添加新的功能。而**PrintStream**只可以对**OutputStream**添加新的功能。
2. 在实际Web开发中，用的最多的就是**PrintWriter**，服务端直接把数据打印出来。**服务器端向客户端发来就用PrintWriter。**
3. 练习题：
4. 从**控制台入**，然后从**控制台**输出其**大写字母**。

打印：

PrintStream ps = new PrintStream(System.out,true);

BufferedReader bufin = new BufferedReader(new InputStreamReader(**System.in**));

String line = null;

while((line = bufin.readLine())!= null){

**if(line.equals("over"))**

**break;**

ps.println(line.toUpperCase());

}

bufin.close(); ps.close();

1. **从控制台输入，写到指定文件中去**。

代码：

PrintStream ps = new PrintStream(new FileOutputStream(new File("e:\\print.txt"),true));

BufferedReader bufin = new BufferedReader(new InputStreamReader(System.in));//利用了转换流

String line = null;

while((line = bufin.readLine())!= null){

if(line.equals("over"))

break;

ps.println(line.toUpperCase());

}

bufin.close();ps.close();