Java Lab2: String, Array and IO

2019.9.9

内容

- 字符串
- 数组
- 输入输出流

字符串

- 声明

  String str

- 创建

  str = new String(“Hello World”)

  str = “Hello World”

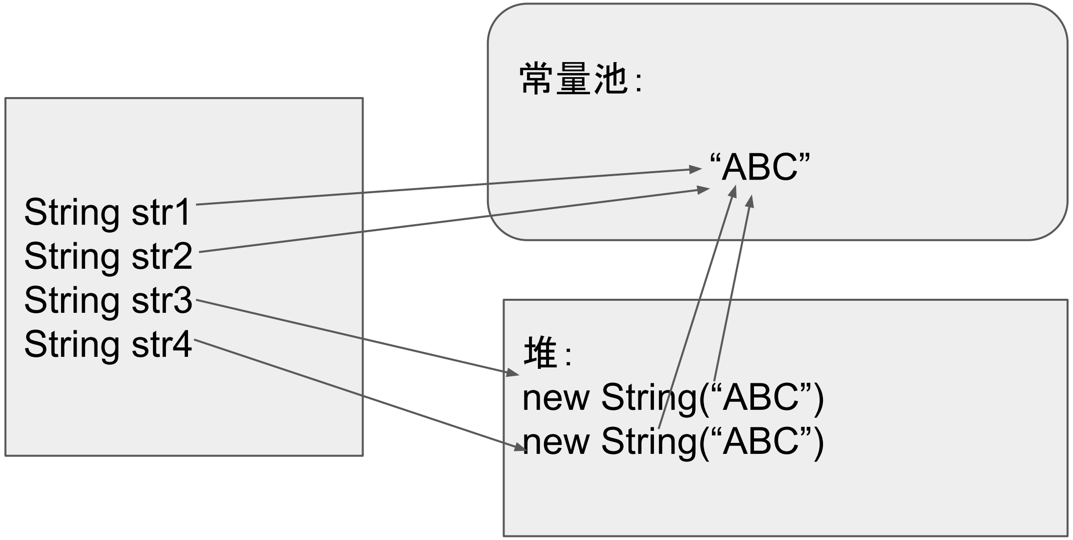
  String = new String(“###”)

字符串

- str = new String(“Hello World”)和str = “Hello World”有什么区别？

- 后者存储在常量池，前者存储在堆中，并保存一个指向常量池的引用

字符串


常量池:
"ABC"
String str1
String str2
String str3
String str4
堆:
new String("ABC")
new String("ABC")

字符串

String str1 = new String(“ABC”);

String str2 = new String(“ABC”);

str1 == str2 返回False

String str3 = “ABC”;

String str4 = “ABC”;

str3 == str4 放回True

- 对于字符串内容的比较，使用equals()

字符串

- 不可变类型（immutable）

  String str = “abc”;

  str = str + “d”;

- 创建了两个字符串：”abcd”和”abc”

字符串

- int length()	返回字符串的长度
- String toUpperCase()	将串中字符变成大写
- String toLowerCase()	将串中字符变成小写
- char charAt(int i)	返回位置i处的字符
- String substring(int s,int e)	返回从位置s到e的字符子串
- String substring(int s)	返回从位置s到末尾的字符子串
- int indexOf(String s)	返回首次出现字符串s的位置
- int indexOf(String s,int i)	返回在位置i之后首次出现s的位置
- String trim()	返回一个新串，去除前后空白字符
- String replace(String a,String b)	返回一个新串，将a替换为b

数组

- 语法

  type[] name 或者 type name[]

- 创建数组

  int[] i = new int[10]; *//int默认为0*

  int[] i = {1, 2, 3, 4, 5}; *//静态初始化*

- 数组一旦被创建，其大小便不可改变

多维数组

- 声明

  type[][] array，type array[][]

- 创建

  int a[][] = new int[2][3]

  Int a[][] = new int[2][];

  a[0] = new int[3];

  a[1] = new int[4];//长度可变

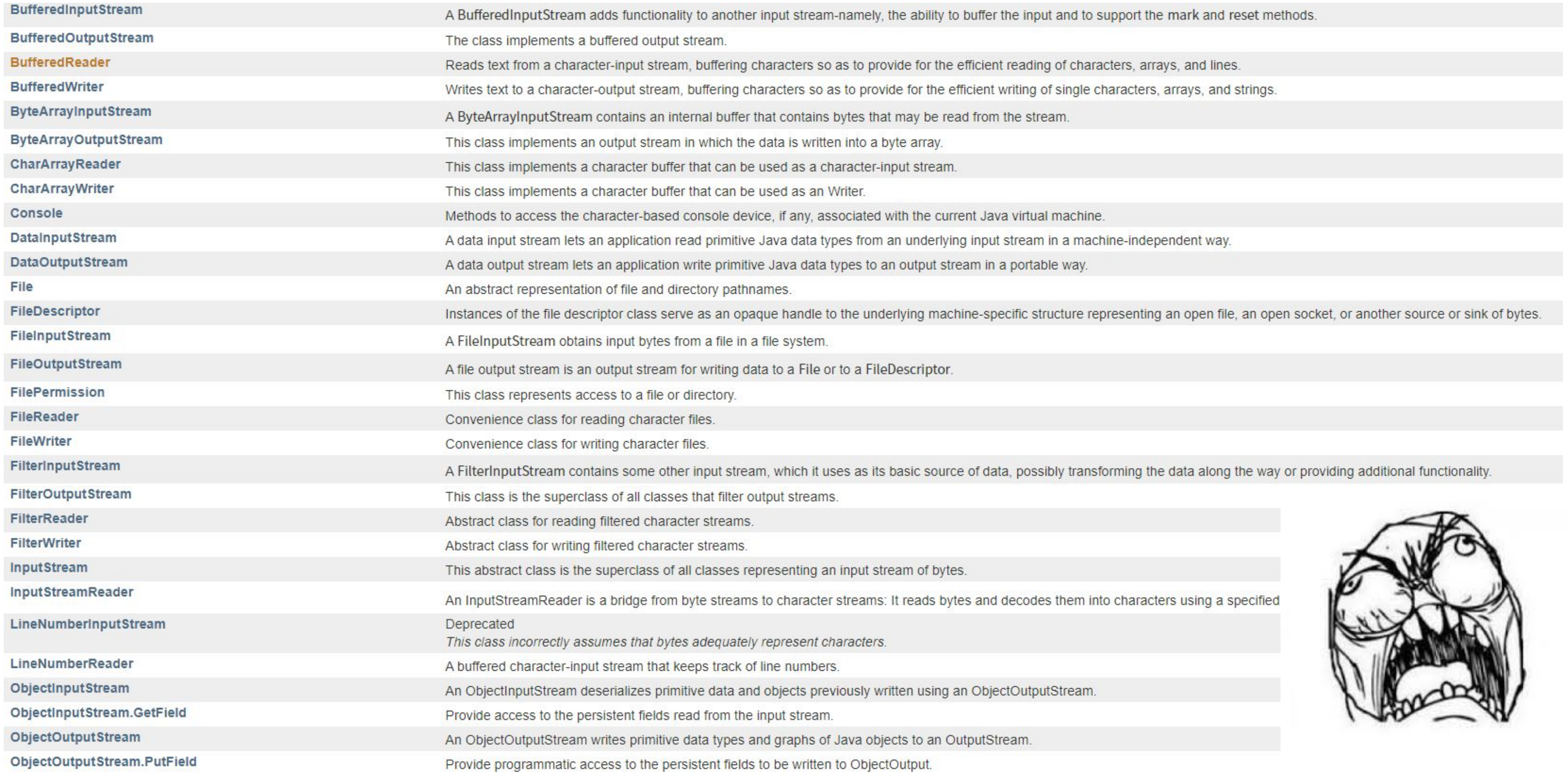
文件

- File类型（别忘了import java.io.*;）

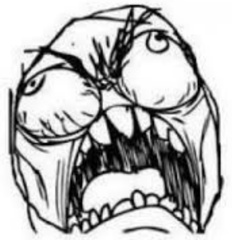
- 获取文件基本信息，如所在目录、长度、读写权限等

- public File(String pathname) 通过文件名创建一个file实例
- public boolean canRead() 判断文件是否可读
- public boolean canWrite() 判断文件是否可写
- public boolean exits() 判断文件或目录是否存在
- public long length() 获取文件长度
- public String getName() 获取文件名字，不包含路径
- public String getAbsolutePath() 获取文件的绝对路径
- public boolean isFile() 判断是否是一个文件
- public boolean isDirectory() 判断是否是一个目录
- public Boolean mkdir() 创建一个目录
- public boolean createNewFile() 创建新文件
- public boolean delete() 删除文件或空目录
- public boolean setReadOnly() 设置文件属性为只读

输入输出流



BufferedInputStream	A BufferedInputStream adds functionality to another input stream-namely, the ability to buffer the input and to support the mark and reset methods.
BufferedOutputStream	The class implements a buffered output stream.
BufferedReader	Reads text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines.
BufferedWriter	Writes text to a character-output stream, buffering characters so as to provide for the efficient writing of single characters, arrays, and strings.
ByteArrayInputStream	A ByteArrayInputStream contains an internal buffer that contains bytes that may be read from the stream.
ByteArrayOutputStream	This class implements an output stream in which the data is written into a byte array.
CharArrayReader	This class implements a character buffer that can be used as a character-input stream.
CharArrayWriter	This class implements a character buffer that can be used as an Writer.
Console	Methods to access the character-based console device, if any, associated with the current Java virtual machine.
DataInputStream	A data input stream lets an application read primitive Java data types from an underlying input stream in a machine-independent way.
DataOutputStream	A data output stream lets an application write primitive Java data types to an output stream in a portable way.
File	An abstract representation of file and directory pathnames.
FileDescriptor	Instances of the file descriptor class serve as an opaque handle to the underlying machine-specific structure representing an open file, an open socket, or another source or sink of bytes.
FileInputStream	A FileInputStream obtains input bytes from a file in a file system.
FileOutputStream	A file output stream is an output stream for writing data to a File or to a FileDescriptor.
FilePermission	This class represents access to a file or directory.
FileReader	Convenience class for reading character files.
FileWriter	Convenience class for writing character files.
FilterInputStream	A FilterInputStream contains some other input stream, which it uses as its basic source of data, possibly transforming the data along the way or providing additional functionality.
FilterOutputStream	This class is the superclass of all classes that filter output streams.
FilterReader	Abstract class for reading filtered character streams.
FilterWriter	Abstract class for writing filtered character streams.
InputStream	This abstract class is the superclass of all classes representing an input stream of bytes.
InputStreamReader	An InputStreamReader is a bridge from byte streams to character streams: It reads bytes and decodes them into characters using a specified
LineNumberInputStream	Deprecated *This class incorrectly assumes that bytes adequately represent characters.*
LineNumberReader	A buffered character-input stream that keeps track of line numbers.
ObjectInputStream	An ObjectInputStream deserializes primitive data and objects previously written using an ObjectOutputStream.
ObjectInputStream.GetField	Provide access to the persistent fields read from the input stream.
ObjectOutputStream	An ObjectOutputStream writes primitive data types and graphs of Java objects to an OutputStream.
ObjectOutputStream.PutField	Provide programmatic access to the persistent fields to be written to ObjectOutput.



输入输出流

- 流（stream）：对一串数据的抽象

- 一组有顺序、有起点和终点的字节集合

输入输出流

- 分为两大类：**字节流**，**字符流**

- 字节流：读入单位为**字节**，用于读取二进制数据

  例：exe文件

  名称：**InputStream/OutputStream**
- 字符流：如数单位为**字符**，用于读取字符数据

  例：txt文件

  名称：**Reader/Writer**

输入输出流

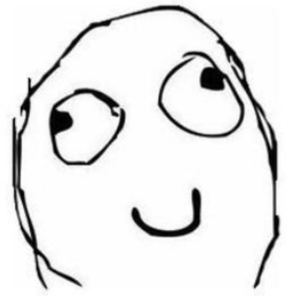
- 使用**装饰者模式**（decorator pattern）

  File file = new File(“example.txt”); *// 需要读取的文件*

  FileInputStream fInput = new FileInputStream(file);*//包装file构建输入流*

  BufferedInputStream bInput = new BufferedInputStream(fInput);*//如果要使用缓冲，包装fInput构建缓冲读入对象*

  //使用缓冲区用于提高读写效率，减少磁盘IO次数



用字节流写文件

```
        public static void fileOutput() throws IOException{
                String str = "hello world!";
                File file = new File("d:\\test.txt");   //创建file对象
                if(!file.exists()){
                                file.createNewFile();   //如果文件不存在，则进行
创建
                }
                FileOutputStream fOutput = new FileOutputStream(file);
                BufferedOutputStream bOutput = new
BufferedOutputStream(fOutput);
                byte[] buffer = str.getBytes(); //将字符串文本转换成字节数组
                bOutput.write(buffer);
                bOutput.close();
                fOutput.close();
        }
```

用字符流写文件

```
        public static void fileWriter() throws IOException{
                String str = "hello world!";
                File file = new File("d:\\test.txt");
               if(!file.exists()){
                        file.createNewFile();    //如果文件不存
在，则进行创建
                }
                FileWriter fwWriter = new FileWriter(file);
                BufferedWriter bWriter = new
BufferedWriter(fwWriter);
                bWriter.write(str);
                bWriter.close();
                fwWriter.close();
        }
```

用字符流读文件

```
public static void fileReader() throws IOException{
        File file = new File("d:\\test.txt");
        FileReader fReader = new FileReader(file);
        BufferedReader buReader = new BufferedReader(fReader);
        String temp = null;
        while((temp = buReader.readLine()) != null){
            System.out.println(temp);
        }
        buReader.close();
        fReader.close();
}
```

输入输出流

一些注意事项

- 需要java io包 import java.io.*;
- 不要忘了异常处理 throws IOException
- 使用完文件需要关闭文件操作 . close()
  - Java GC只关心堆中的对象，对于系统资源需要程序员显示操作
  - 保证信息写入文件，释放系统资源

Q？