java.io

**Class BufferedReader**

* [java.lang.Object](https://docs.oracle.com/javase/8/docs/api/java/lang/Object.html)
  + [java.io.Reader](https://docs.oracle.com/javase/8/docs/api/java/io/Reader.html)
    - java.io.BufferedReader
* **All Implemented Interfaces:**

[Closeable](https://docs.oracle.com/javase/8/docs/api/java/io/Closeable.html), [AutoCloseable](https://docs.oracle.com/javase/8/docs/api/java/lang/AutoCloseable.html), [Readable](https://docs.oracle.com/javase/8/docs/api/java/lang/Readable.html)

**Direct Known Subclasses:**

[LineNumberReader](https://docs.oracle.com/javase/8/docs/api/java/io/LineNumberReader.html)

BufferedReader可以从字符输入流中读取字符，当通过此类读文件是，会先尽量将文件中的字符读取到缓存区，之后若使用read（）方法，会先从缓存区读取，如果缓存区不足，才会再从文件中读取（也就是再从内存中读。内存与缓存的区别，请参考硬件方面知识）。

缓存区的大小是可以指定的，也可以使用默认的大小。一般来说默认的缓存大小就足够使用。

BufferedReader是用来提高读取速度的，所以构造时，必须以Reader的其他子类作为参数，来进行构造。不能直接构造一个无参数的BufferedReader。

|  |
| --- |
| **Constructors** |
| **Constructor and Description** |
| [**BufferedReader**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#BufferedReader-java.io.Reader-)([**Reader**](https://docs.oracle.com/javase/8/docs/api/java/io/Reader.html) in)  使用默认的大小创建一个字符输入流。传入的参数可以是Reader的所有子类。 |
| [**BufferedReader**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#BufferedReader-java.io.Reader-int-)([**Reader**](https://docs.oracle.com/javase/8/docs/api/java/io/Reader.html) in, int sz)  通过sz指定的大小，创建一个BufferedReader |

方法

|  |  |
| --- | --- |
| int | [**read**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#read--)()  从字符流中读取一个字符 |
|  |  |

|  |
| --- |
| 实例 |
| **public** **static** **void** main(String[] args) **throws** IOException{  **char**[] chars = {'我','b','c'};  CharArrayReader charArrayReader = **new** CharArrayReader(chars);  //BufferedReader是用来提高读取速度的缓存输入流，所以必须配合Reader的子类来创建。  //所以这里使用Reader的一个子类CharArrayReader来作为参数创建BufferedReader  BufferedReader reader = **new** BufferedReader(charArrayReader);  System.***out***.println((**char**)reader.read());//我。如果不强转则输出一个0到65535之间的数字，因为一个char两个字节2的16次方就是65536  } |

|  |  |
| --- | --- |
| int | [**read**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#read-char:A-int-int-)(char[] cbuf, int off, int len)  从当前流的当前位置读取len个字符，将读取的字符放到cbuf这个数组中，从cbuf哪个位置开始往里放，由off来决定。Off+len的值不能大于cbuf的容量。 |

|  |
| --- |
| 实例 |
| **public** **static** **void** main(String[] args) **throws** IOException{  **char**[] chars = {'我','b','c'};  **char**[] chars1 = **new** **char**[10];  CharArrayReader charArrayReader = **new** CharArrayReader(chars);  //BufferedReader是用来提高读取速度的缓存输入流，所以必须配合Reader的子类来创建。  //所以这里使用Reader的一个子类CharArrayReader来作为参数创建BufferedReader  BufferedReader reader = **new** BufferedReader(charArrayReader);  reader.skip(1);//将流中当前读取位置设置到b这个位置  reader.read(chars1, 2, 1);//从b这个字符开始读取，读取一个，然后放到chars1数组index为2的位置。  System.***out***.println(chars1[2]);//b。前条语句将b这个字符保存到了chars1[2]中  } |

|  |  |
| --- | --- |
| void | [**close**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#close--)()  关闭当前流。 |

|  |
| --- |
| 实例 |
| **public** **static** **void** main(String[] args) **throws** IOException{  **char**[] chars = {'我','b','c'};  **char**[] chars1 = **new** **char**[10];  CharArrayReader charArrayReader = **new** CharArrayReader(chars);  charArrayReader.close();  charArrayReader.skip(1);//发生异常。因为流已经关闭了。如下：  }  Exception in thread "main" java.io.IOException: Stream closed  at java.io.CharArrayReader.ensureOpen(CharArrayReader.java:92)  at java.io.CharArrayReader.skip(CharArrayReader.java:159)  at jdkdemo.ReaderTest.main(ReaderTest.java:15) |

|  |  |
| --- | --- |
| [**Stream**](https://docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html)<[**String**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html)> | [**lines**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#lines--)()  从BufferedReader中读取所有内容，将其放到一个Stream对象中。. |

|  |
| --- |
| 实例： |
| **public** **static** **void** main(String[] args) **throws** IOException{  **char**[] chars = {'我','b','c'};  CharArrayReader charArrayReader = **new** CharArrayReader(chars);  BufferedReader reader = **new** BufferedReader(charArrayReader);  // FileReader fileReader = new FileReader("C:\\Users\\Administrator\\Desktop\\新建文本文档.txt");  // BufferedReader reader = new BufferedReader(fileReader);  Stream<String> stream = reader.lines();    Iterator<String> iter = stream.iterator();  **while**(iter.hasNext()){  System.***out***.println(**new** String(iter.next().getBytes("utf-8")));  }  } |

实例中试验了了文件和字符数组两种形式。只要是用BufferedReader构建的，此方法就可以吧流中的内容全部读到Stream对象中。然后就可以通过Stream（java8新特性）对象操作了。

|  |  |  |
| --- | --- | --- |
| void | | [**mark**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#mark-int-)(int readAheadLimit)  在流的当前位置做标记，reset时，将指针移动到标记位置，流会从标记位置重新开始读取。可参照ArrayInputStream |
| boolean | | [**markSupported**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#markSupported--)()  查看是否支持mark方法. |
| void | | [**reset**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#reset--)()  将指针移动到mark的位置。 |
| long | [**skip**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#skip-long-)(long n)  将指针从当前位置顺势向后移动n个字符（因为是字符流。所以是n个字符） | |

|  |
| --- |
| **实例** |
| **public** **static** **void** main(String[] args) **throws** IOException{  **char**[] chars = {'我','b','c'};  CharArrayReader charArrayReader = **new** CharArrayReader(chars);  BufferedReader reader = **new** BufferedReader(charArrayReader);    reader.skip(1);  reader.mark(1);//定位到b  System.***out***.println((**char**)reader.read());//b  reader.reset();  System.***out***.println((**char**)reader.read());//b    } |

|  |  |
| --- | --- |
| [**String**](https://docs.oracle.com/javase/8/docs/api/java/lang/String.html) | [**readLine**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#readLine--)()  读取一行。一行的定义如下：当碰到\n或者\r或者carriage return和linefeed（中文里叫做回车和换行符吧）就算是一行了。 |

|  |
| --- |
| **实例** |
| **public** **static** **void** main(String[] args) **throws** IOException{  **char**[] chars = {'我','b','\r','c'};  CharArrayReader charArrayReader = **new** CharArrayReader(chars);  BufferedReader reader = **new** BufferedReader(charArrayReader);    System.***out***.println(reader.readLine());//因为碰到\r了，所以就表示一行的结束。输出：我b  System.***out***.println(reader.readLine());//在读的时候，把剩下的都读出来来。输出：c  } |

这里用的是char数组。所有手动设置了\r来展示何为一行的概念。

当读文件的时候，文件中一行就是默认的一行。

|  |  |
| --- | --- |
| boolean | [**ready**](https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html#ready--)()  判断当前流是否已经可以使用来读取了。如果不懂得bufferedReader中的字符流不是空的。则返回true，表示可以读取了。 |

|  |
| --- |
| 实例 |
| **public** **static** **void** main(String[] args) **throws** IOException{  **char**[] chars = {};  CharArrayReader charArrayReader = **new** CharArrayReader(chars);  BufferedReader reader = **new** BufferedReader(charArrayReader);    System.***out***.println(reader.ready());//false    } |

因为chars没有内容，所以ready方法返回false。如果chars里面有值，则返回true