compact1, compact2, compact3
java.io

Class BufferedReader

java.lang.Object
  java.io.Reader
    java.io.BufferedReader

All Implemented Interfaces:

Closeable, AutoCloseable, Readable

已知直接子类:

LineNumberReader

```
public class BufferedReader
extends Reader
```

从一个字符输入流中读取文本，缓冲字符，以便提供字符、数组和行的有效读取。

可以指定缓冲区大小，也可以使用默认大小。默认是足够大的用于大多数目的。

在一般情况下，每一个读的读者提出的要求导致相应的读请求是由底层字符或字节流。这是包体周围的read()操作可能是昂贵的任何读者因此为宜，如filereaders和inputstreamreaders。例如，

```
BufferedReader=新BufferedReader（FileReader（“foo”））；
```

将缓冲输入从指定的文件。无缓冲，每次调用read()或readline()可能导致数据被从文件读取，转换成文字，然后返回，这可以是非常低效的。

程序使用datainputstreams文本输入可以通过一个合适的是局部更换每个输入流。

从以下版本开始:

JDK1.1

另请参见:

FileReader, InputStreamReader, Files.newBufferedReader(java.nio.file.Path, java.nio.charset.Charset)

Field Summary

Fields inherited from class java.io.Reader

lock

构造方法摘要

构造方法

Constructor and Description

BufferedReader(Reader in)
创建一个使用默认大小输入缓冲区的缓冲字符输入流。

BufferedReader(**Reader** in, int sz)
创建一个使用指定大小的输入缓冲区的缓冲字符输入流。

方法摘要

所有方法 **接口方法** **具体的方法**

Modifier and Type	Method and Description
void	**close**() 关闭流并释放与它相关联的任何系统资源。
Stream<**String**>	**lines**() 返回一个 Stream，其中的元素是线从这 BufferedReader读。
void	**mark**(int readAheadLimit) 标记流中的当前位置。
boolean	**markSupported**() 告诉这是否流支持的mark()操作，它。
int	**read**() 读取单个字符。
int	**read**(char[] cbuf, int off, int len) 将字符读入一个数组的一部分。
String	**readLine**() 读一行文本。
boolean	**ready**() 告诉是否该流已准备好阅读。
void	**reset**() 将流到最近的标记。
long	**skip**(long n) 跳过的字符。

Methods inherited from class java.io.Reader

read, read

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

BufferedReader

```
public BufferedReader(Reader in,
                      int sz)
```

创建一个使用指定大小的输入缓冲区的缓冲字符输入流。

参数

in -读者

sz输入缓冲区的大小

异常

如果 sz <= 0 IllegalArgumentException

BufferedReader

```
public BufferedReader(Reader in)
```

创建一个使用默认大小输入缓冲区的缓冲字符输入流。

参数

in -读者

方法详细信息

read

```
public int read()
         throws IOException
```

读取单个字符。

重写:

read 方法重写，继承类 Reader

结果

个性阅读，如在范围0到65535的整数（ 0x00-0xffff），或1如果已到达流的末尾

异常

IOException如果I/O错误发生

read

```
public int read(char[] cbuf,
                int off,
                int len)
         throws IOException
```

将字符读入一个数组的一部分。

该方法实现了对Reader类相应的read方法一般合同。作为一个额外的便利，它试图读尽可能多的字符，通过反复调用的底层流read方法。这read迭代直至下列条件之一为真：

- 指定的字符数已被读取，
- 对基础流返回-1的read方法，表示文件结束，或
- 对基础流返回false的ready方法，表明进一步的输入请求块。

如果基础流的第一 read返回 -1表示文件结束之后这个方法返回 -1。否则，此方法返回实际读取的字符数。

这个类的子类是鼓励的，但不是必需的，试图以相同的方式阅读尽可能多的字符。

通常此方法需要从这个流的字符缓冲区中的字符，从底层流填充它作为必要的。然而，如果缓冲区是空的，标记无效，并且请求的长度至少和缓冲区一样大，那么这个方法将从底层流直接读取到给定的数组中的字符。因此BufferedReaders不会复制不必要的数据冗余。

Specified by:

read 方法重写，继承类 Reader

参数

cbuf -目的缓冲区

off偏移，开始存储字符

len -最大字符数读

结果

读取的字符数，或-如果流的结束已达到- 1

异常

IOException如果I/O错误发生

readLine

```
public String readLine()
                throws IOException
```

读一行文本。一条线是由一个换行的任何一个终止（'\n'），回车（“R”），或一个回车紧接着换行。

结果

一个字符串，包含该行的内容，不包括任何行终止字符，或空，如果流结束已达到

异常

IOException如果I/O错误发生

另请参见:

Files.readAllLines(java.nio.file.Path, java.nio.charset.Charset)

skip

```
public long skip(long n)
          throws IOException
```

跳过的字符。

重写:

skip 方法重写，继承类 Reader

参数

n -字符数跳过

结果

实际上跳过的字符数

异常

IllegalArgumentException -如果 n是负的。

IOException如果I/O错误发生

ready

```
public boolean ready()
              throws IOException
```

告诉是否该流已准备好阅读。如果缓冲区不是空的，或者基本的字符流已准备就绪，则缓冲字符流已准备好了。

重写:

ready 方法重写，继承类 Reader

结果

如果下一read()保证不阻塞输入真，否则为假。请注意，返回错误不保证下一次读将块。

异常

IOException如果I/O错误发生

markSupported

```
public boolean markSupported()
```

告诉这是否流支持的mark()操作，它。

重写:

`markSupported` 方法重写，继承类 `Reader`

结果

如果和只有这条流支持标记操作，则是真的。

mark

```
public void mark(int readAheadLimit)
          throws IOException
```

标记流中的当前位置。随后调用reset()将试图重新定位流这一点。

重写:

`mark` 方法重写，继承类 `Reader`

参数

`readAheadLimit`的字符数限制，可同时仍保留了马克读。在读到这个极限或超越的字符后，重新设置流的尝试可能会失败。一个大于输入缓冲区大小的限制值将导致一个新的缓冲区被分配，其大小不小于极限。因此，大的价值观应该谨慎使用。

异常

如果 `readAheadLimit < 0` `IllegalArgumentException`

`IOException`如果I/O错误发生

reset

```
public void reset()
           throws IOException
```

将流到最近的标记。

重写:

`reset` 方法重写，继承类 `Reader`

异常

`IOException`如果流没有被标记，或者如果该商标已失效

close

```
public void close()
           throws IOException
```

从阶级复制的描述： **`Reader`**

关闭流并释放与它相关联的任何系统资源。一旦流已经关闭，进一步read()，ready()，mark()，reset()，或skip()调用会抛出IOException。关闭先前关闭的流没有影响。

Specified by:

`close` 接口 `Closeable`

Specified by:

`close` 接口 `AutoCloseable`

Specified by:

`close` 方法重写，继承类 `Reader`

异常

`IOException`如果I/O错误发生

lines

```
public Stream<String> lines()
```

返回一个 `Stream`，其中的元素是线从这 `BufferedReader`读。是的 `Stream`懒洋洋地填充，即读 terminal stream operation 只发生在。

在终端流操作的执行过程中，不得操作读写器。否则，终端流操作的结果是不确定的。

执行后的终端流操作有没有保证，读者将在一个特定的位置，从中读取下一个字符或行。

如果一个`IOException`被访问底层的`BufferedReader`时，它被包裹在一个`UncheckedIOException`将从引起阅读发生的`Stream`方法引发。这个方法如果在BufferedReader，关闭调用返回一个流。在流需要阅读从体后关闭任何操作，将导致uncheckedioexception被。

结果

提供文字描述的这 `BufferedReader`线 `Stream<String>`

从以下版本开始:

一点八