Teil IX

Datenspeicherung und -transfer – Java Streams

UNIVERSITAT LEIPZIG

Streams: Interfaces

Try with resources

- java.io.AutoCloseable: void close() throws Exception
 - ∟ java.io.Closeable
- java.io.Flushable: void flush() throws IOException
- java.io.Appendable
- java.io.Readable

Streams: Lesen

Zeichen-basiertes Lesen

- java.io.Reader
 - ∟ java.io.BufferedReader
 - \bot java.io.LineNumberReader
 - $\ \ \, \bot \ \, \texttt{java.io.InputStreamReader}$
 - \bot java.io.FileReader

implements:

- ▶ Closeable
- ► AutoCloseable
- ► Readable

Streams: Lesen

Zeichen-basiertes Lesen

- java.io.Reader
 - ∟ java.io.BufferedReader
 - ∟ java.io.LineNumberReader
 - - oxdot java.io.FileReader

implements:

- ► Closeable
- ► AutoCloseable
- ► Readable

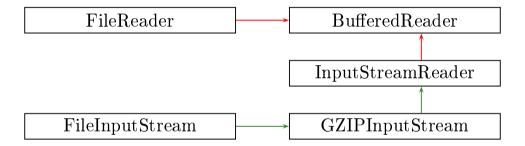
Byte-basiertes Lesen

- java.io.InputStream
 - ∟ java.io.FilterInputStream
 - ∟ java.io.BufferedInputStream
 - ∟ java.io.LineNumberInputStream
 - ∟ java.util.zip.InflaterInputStream
 - ∟ java.util.zip.GZIPInputStream
 - ∟ java.util.zip.ZipInputStream
 - ∟ java.io.FileInputStream

implements:

- ► Closeable
- ► AutoCloseable

Streams: Lesen



Streams: Daten lesen

```
public static List < String > readLines (
       String fileName
     ) {
       List < String > result = new ArrayList <>():
       File file = new File(fileName);
       trv (
         FileReader fileReader
           = new FileReader(file):
10
11
12
13
14
15
         BufferedReader bufferedReader
16
           = new BufferedReader(fileReader):
17
18
         readLine(bufferedReader, result):
19
       } catch (FileNotFoundException fnfEx) {
20
         fnfEx.printStackTrace(System.err);
21
       } catch (IOException ioEx) {
22
         ioEx.printStackTrace(System.err):
23
       } finally {
24
         return result:
25
26
```

Streams: Daten lesen

```
public static List < String > readLines(
                                                              public static List < String >
       String fileName
                                                                   readLinesCompressed(
     ) {
                                                                String fileName
                                                          3
       List < String > result = new ArrayList <>():
                                                              ) {
 56789
       File file = new File(fileName):
                                                                List < String > result = new ArrayList <>();
       trv (
                                                                File file = new File(fileName):
         FileReader fileReader
                                                                try (
           = new FileReader(file):
                                                                  FileInputStream fis
                                                                     = new FileInputStream(file);
10
                                                                  BufferedInputStream bis
11
                                                         10
                                                                     = new BufferedInputStream(fis);
12
                                                        11
                                                                  GZIPInputStream gzis
13
                                                                     = new GZIPInputStream(bis):
14
                                                        13
                                                                  InputStreamReader isr
15
                                                         14
         BufferedReader bufferedReader
                                                                     = new InputStreamReader(gzis);
16
                                                                  RufferedReader bufferedReader
           = new BufferedReader(fileReader):
17
                                                         16
                                                                     = new BufferedReader(isr)
18
                                                         17
         readLine(bufferedReader, result):
                                                                ) {
19
                                                         18
       } catch (FileNotFoundException fnfEx) {
                                                                  readLine(bufferedReader, result):
20
                                                         19
                                                                } catch (FileNotFoundException fnfEx) {
         fnfEx.printStackTrace(System.err):
21
       } catch (IOException ioEx) {
                                                         20
                                                                  fnfEx.printStackTrace(System.err);
22
                                                         21
         ioEx.printStackTrace(System.err):
                                                                } catch (IOException ioEx) {
23
                                                         22
       } finally {
                                                                  ioEx.printStackTrace(System.err):
                                                         23
24
                                                                } finally {
         return result:
25
                                                        24
                                                                  return result:
26
                                                        25
                                                        26
```

Streams: Daten lesen

```
private static void readLine(
   BufferedReader bufferedReader,
   List<String> result
   throws IOException {
   String line = bufferedReader.readLine();
   while (line! = null) {
      result.add(line);
   line = bufferedReader.readLine();
}
```

Streams: Zeichen-basiertes Schreiben

java.io.Writer

- ▶ implements: Closeable, Flushable, Appendable, AutoCloseable
- abstract void close() throws IOException
- abstract void flush() throws IOException
- ▶ void write(<Typ> arg) throws IOException

java.io.PrintWriter extends Writer

- void close()
- void flush()
- void write(<Typ> arg)
- void print(<Typ> arg)
- void println(<Typ> arg)
- → keine Exceptions

Streams: Zeichen-basiertes Schreiben

java.io.OutputStreamWriter extends Writer
java.io.FileWriter extends OutputStreamWriter

java.io.BufferedWriter extends Writer

UNIVERSITAT LEIPZIG

Streams: Byte-basiertes Schreiben

java.io.OutputStream

- ▶ implements: Closeable, Flushable, AutoCloseable
- void close() throws IOException
- void flush() throws IOException
- abstract void write()

java.io.FilterOutputStream extends OutputStream

java.io.BufferedOutputStream extends FilterOutputStream

java.util.zip.DeflaterOutputStream extends FilterOutputStream java.util.zip.GZIPOutputStream extends DeflaterOutputStream java.util.zip.ZipOutputStream extends DeflaterOutputStream

Streams: Schreiben



Streams: Daten schreiben

```
public static void writeLines(
         String fileName,
         List < String > lines
     ) {
       File file = new File(fileName);
       trv (
         FileWriter fw = new FileWriter(file);
10
11
         BufferedWriter by = new BufferedWriter
          (fw);
13
         PrintWriter pw = new PrintWriter(bw);
14
15
         writeLine(pw, lines);
16
       } catch (FileNotFoundException fnfEx) {
17
         fnfEx.printStackTrace(System.err);
18
       } catch (IOException ioEx) {
19
         ioEx.printStackTrace(System.err);
20
```

Streams: Daten schreiben

```
public static void writeLines(
                                                              public static void writeLinesCompressed(
         String fileName,
                                                                String fileName,
 3
         List < String > lines
                                                               List < String > lines
     ) {
                                                             ) {
 56789
                                                               File file = new File(fileName):
       File file = new File(fileName):
       trv (
                                                               trv (
         FileWriter fw = new FileWriter(file);
                                                                  FileOutputStream fos
                                                                    = new FileOutputStream(file);
                                                                  GZIPOutputStream gzos
10
                                                        10
                                                                    = new GZIPOutputStream(fos);
11
                                                        11
         BufferedWriter by = new BufferedWriter
                                                                  BufferedOutputStream bos
                                                        12
          (fw):
                                                                    = new BufferedOutputStream(gzos);
                                                        13
                                                                  PrintWriter pw = new PrintWriter(bos):
13
                                                        14
         PrintWriter pw = new PrintWriter(bw):
14
                                                        15
                                                                  writeLine(pw, lines);
15
                                                        16
         writeLine(pw. lines):
                                                               } catch (FileNotFoundException fnfEx) {
16
                                                        17
       } catch (FileNotFoundException fnfEx) {
                                                                  fnfEx.printStackTrace(System.err);
17
                                                        18
         fnfEx.printStackTrace(System.err);
                                                               } catch (IOException ioEx) {
18
                                                        19
       } catch (IOException ioEx) {
                                                                  ioEx.printStackTrace(System.err);
19
         ioEx.printStackTrace(System.err):
                                                        20
20
       7
21
```

Streams: Daten schreiben

```
private static void writeLine(
    PrintWriter pw,
    List<String> lines

    ) {
      for (String line : lines) {
        pw.println(line);
      }
}
```