

Dateien - Eingabe / Ausgabe

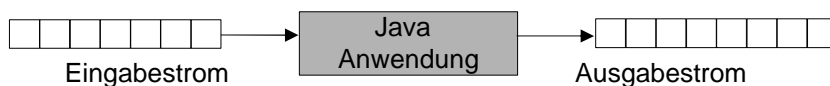
Silke Trißl

Wissensmanagement in der Bioinformatik

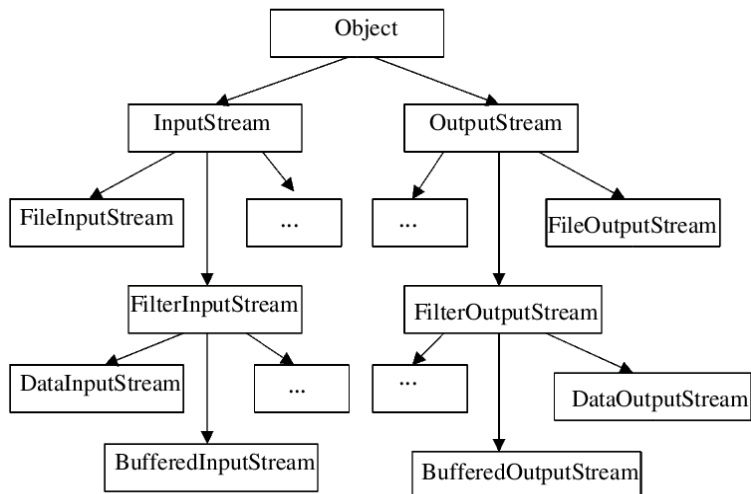


Ein- und Ausgabe

- Daten in Dateien schreiben und von Dateien auslesen
- Package: `java.io.*`
- Stromkonzepte: Standard Ein- / Ausgabe
-



Klassen von java.io



Silke

3

In Dateien schreiben

```
PrintWriter out =  
    new PrintWriter(new FileWriter('out.txt'));
```

- `PrintWriter`
 - Strings und Zahlen in Textformat umwandeln
 - hat Methoden `print()` und `println()`
 - siehe auch `System.out`
- `FileWriter`
 - Zeigt auf das File, wo der `OutputStream` hingehen soll
 - ohne Pfadangabe: File im aktuellen Pfad (*.java)

Exceptions

- Mögliche Probleme beim Schreiben / Lesen einer Datei
 - keine Schreibberechtigung im Verzeichnis
 - Datei existiert nicht
 - keine Leseberechtigung der Datei
 - über das Ende der Datei hinaus gelesen
- Programm terminiert, ohne sich um den Rest des Codes zu kümmern
 - Exceptions fangen dies ab
 - IOException, EOFException (bei Dateien)

Exception auffangen

- Fehler im try-Block:
 - überspringt restlichen Code
 - führt Anweisungen im catch-Block aus

```
try {  
    // Code  
    // Weiterer Code  
}  
catch (ExceptionType e) {  
    // Behandlungsroutine für diesen Typ  
}
```

Beispiel - FileOutputStream.java

```
try {
    // PrintWriter (DataOutputStream)
    // kombiniert mit dem Ziel-Stream 'FileWriter'
    PrintWriter out =
        new PrintWriter( new FileWriter("test.txt") );
    out.print("Der");
    out.println(" Test");
    out.print("Nummer " + 5);

    // Schließt das OutputFile wieder
    out.close()
}
catch(IOException exception) {
    exception.printStackTrace();
}
```

Daten aus Dateien lesen

- Benötigt Exception-handling (try ... catch)

```
BufferedReader in = new
    BufferedReader (new FileReader("in.txt"));
```

Zeilenweise Auslesen

- BufferedReader ermöglicht eine Zeile aus der Datei auszulesen mit der Methode `readLine`

```
String s;  
while ( (s = in.readLine()) != null ) {  
    // etwas mit s unternehmen  
}
```

- Sobald eine Zeile ausgelesen wurde, wird beim nächsten Aufruf die folgende Zeile ausgelesen

Daten aus Dateien lesen - cont.

- Jede Zeile ergibt nur einen String
- Sind mehrere Werte in einer Zeile gespeichert, so müssen sie durch einen eindeutigen Begrenzer getrennt sein
- Beispiel:
 - `Mustermann|Thomas|123456|5|Biophysik|thmuster`
 - `Musterfrau|Petra|654321|5|Biophysik|pemuster`

Tokenizer

- Zeile in Teilstrings (Tokens) zerlegen

```
StringTokenizer t =
    new StringTokenizer(s, "|");

// jeden Token ausgeben
while( t.hasMoreTokens() ) {
    String substr = t.nextToken();
}
```

Substrings umwandeln

- Tokenizer erzeugt Strings
- Umwandlung notwendig für mathematische Operationen

```
String s;
int i = Integer.parseInt(s);
double d = Double.parseDouble(s);
```

```
try {
    int i = Integer.parseInt(s); }
catch (NumberFormatException exception) {
    ... }
```



Aufgabe: Auslesen von Daten aus einer Datei

- In der Datei 'Musterdaten.txt' sind Informationen über Studenten folgendermaßen gespeichert:
 - Nachname|Vorname|Matrikelnummer|Semester|Studiengang|(Account)
 - Account kann vorhanden sein, muss aber nicht

- Lest diese Daten ein und gebt sie in folgender Weise wieder aus:
 - Vornamen Namen ist
 - Wenn das Semester < 5 ist: nicht im Praktikum
 - sonst: im Praktikum