


Serialisierung

Serialisierung

Unter Serialisierung verstehen wir die Fähigkeit, ein Objekt, das im Hauptspeicher der Anwendung existiert, in ein Format zu konvertieren, das es erlaubt, das Objekt in eine Datei zu schreiben oder über eine Netzwerkverbindung zu transportieren. Der umgekehrte Weg wird ebenfalls mit eingeschlossen.

-
- 
-
- ❖ ObjectOutputStream erlaubt das Schreiben von serialisierten Objekten
 - ❖ ObjektInputStream erlaubt das Lesen von serialisierten Objekten
 - ❖ beide brauchen einen Anschlussstrom

Beispiel

```
FileOutputStream fos =
    new FileOutputStream(new File("/Users/schulung/workspace/serialisierung/Time.ser"));
ObjectOutputStream oos = new ObjectOutputStream(fos);
Time time = new Time(10, 20, 23);
time.setMilliSekunden(45);
System.out.println("Zeit vorm Seralisieren: "+time);
System.out.println("Und die Sekunden: "+ time.getMilliSekunden());
oos.writeObject(time);
oos.close();

FileInputStream fis =
    new FileInputStream(new File("/Users/schulung/workspace/serialisierung/Time.ser"));
ObjectInputStream ois = new ObjectInputStream(fis);
Time time2 = (Time)ois.readObject();
ois.close();
//Beachte die Millisekunden haben wieder den Defaultwert von int
System.out.println("Zeit nach dem Deserialisieren: "+time2);
//Vaterklasse implementiert nicht Serializable, also Aufruf des parameterlosen Konstruktors.
System.out.println("Und die Sekunden: "+ time2.getMilliSekunden());
```


writeObject

- ❖ schreibt folgendes in den OutputStream
 - ❖ Die Klasse des als Argument übergebenen Objekts
 - ❖ Die Signatur der Klasse
 - ❖ alle nichtsstatischen, nichttransienten
Membervariablen des übergebenen Objekts inkl. der
aus allen Vaterklassen geerbten Membervariablen

Deserialisieren

- ❖ stark vereinfacht wie folgt:
 - ❖ Objektanlage des zu deserialisierenden Typs und die Membervariablen werden mit Default-Werten vorbelegt. Zudem wird der Default-Konstruktor der ersten nichtserialisierbaren Superklasse aufgerufen.
 - ❖ serialisierten Daten werden gelesen und den entsprechenden Membervariablen zugewiesen.
 - ❖

readObject

- ❖ Kriterien für erfolgreiches Ausführen:
 - ❖ Element des Eingabestreams ist tatsächlich ein Object
 - ❖ das Objekt muss sich vollständig und fehlerfrei aus der Eingabedatei lesen lassen
 - ❖ Konvertierung auf den gewünschten Typ erlauben, also entweder zu derselben oder einer daraus abgeleiteten Klasse gehören.
 - ❖ Bytecode muss vorhanden sein.
 - ❖ Klasseninformationen des ser. Objekts und die Bytecode-Demi müssen zueinander kompatibel sein.

Versionierung

- ❖ serialVersionUID wird erzeugt und in die Ausgabedatei geschrieben
- ❖ beim Deserialisieren Vergleich der
UIDs(InvalidClassException bei Ungleichheit)
- ❖ kleinste Abänderung verändert die serialVersionUID

Versionierung

- ❖ Möglichkeit der Vergabe der serialVersionUID durch zu serialisierende Klasse
- ❖ Anwendung trägt damit selbst Sorge dafür, dass eventuelle Änderungen kompatibel bleiben.

Änderung von zu serialisierenden Objekten

- ❖ Regeln:
 - ❖ Hinzufügen und Entfernen von Methoden ist meist unkritisch
 - ❖ Entfernen von Membervariablen meist unkritisch
 - ❖ Hinzufügen neuer Membervariablen bleiben nach dem Deserialisieren uninitialisiert.
 - ❖ Kritisch ist meist:
 - ❖ Umbenennen von Membervariablen, sie transient oder static zu setzen

Objektreferenzen

- ❖ auch Referenzen werden gesichert
- ❖ gleiche Objekte werden nur einmal serialisiert
- ❖ Hashtable zur Erkennung ob Objekt doppelt oder nicht

Wahr oder Falsch

- ❖ Serialisierung eignet sich, um die Daten für die Verwendung durch Nicht Java-Programme zu speichern.
- ❖ Objektzustände können nur mittels Serialisierung gespeichert werden.
- ❖ Mit der Klasse ObjectOutputStream werden serialisierte Objekte gespeichert.
- ❖ Verkettungsströme können für sich allein oder zusammen mit Anschlussströmen benutzt werden.
- ❖ Ein einziger Aufruf von writeObject kann die Speicherung von vielen Objekten bewirken.

Wahr oder Falsch

- ❖ Alle Klassen sind standardmäßig serialisierbar.
- ❖ Mit einem transient-Modifizierer kann man Instanzvariablen serialisierbar machen.
- ❖ Wenn eine Superklasse nicht serialisierbar ist, kann die Unterklasse es auch nicht sein.
- ❖ Bei der Deserialisierung werden Objekte in umgekehrter Reihenfolge, d.h. nach dem Prinzip „last in first out“ eingelesen.
- ❖ Bei der Deserialisierung eines Objektes wird der Konstruktion nicht ausgeführt.

Wahr oder Falsch

- ❖ Sowohl bei der Serialisierung als auch bei der Speicherung in eine Textdatei können Exceptions ausgelöst werden.
- ❖ Ein BufferedWriter kann mit einem FileWriter verkettet werden.
- ❖ File-Objekte repräsentieren Dateien, aber keine Verzeichnisse
- ❖ Sie können einen Puffer nicht zum Senden seiner Daten zwingen, bevor er voll ist.
- ❖ Sowohl FileReader als auch FileWriter können wahlweise auch gepudert werden.

Wahr oder Falsch

- ❖ Jede Veränderung an einer Klasse macht alle zuvor serialisierten Objekte der Klasse unbrauchbar.