Tutorium Praktische Informatik 2

Jonathan Köhn

Technische Hochschule Köln

Fakultät für Informations-, Medien- und Elektrotechnik

Institut für Nachrichtentechnik

17. Mai 2016

Wiederholung

1. Was bedeutet `Serialisierung`?

Wiederholung

1. Was bedeutet `Serialisierung`?

A: Strukturierte Daten (hier Java-Objekte) in Bytestrom umwandeln.

Wiederholung

1. Was bedeutet `Serialisierung`?

A: Strukturierte Daten (hier Java-Objekte) in Bytestrom umwandeln.

2. Zu welchem Zweck können wir Serialisierung nutzen?

Wiederholung

1. Was bedeutet `Serialisierung`?

A: Strukturierte Daten (hier Java-Objekte) in Bytestrom umwandeln.

2. Zu welchem Zweck können wir Serialisierung nutzen?

A Persistente (dauerhafte) Speicherung von Objekten.

Wiederholung

1. Was bedeutet `Serialisierung`?

A: Strukturierte Daten (hier Java-Objekte) in Bytestrom umwandeln.

2. Zu welchem Zweck können wir Serialisierung nutzen?

A Persistente (dauerhafte) Speicherung von Objekten.

3. Mit welchem Schlüsselwort werden serialisierbare Klassen in Java gekennzeichnet?

Wiederholung

1. Was bedeutet `Serialisierung`?

A: Strukturierte Daten (hier Java-Objekte) in Bytestrom umwandeln.

2. Zu welchem Zweck können wir Serialisierung nutzen?

A Persistente (dauerhafte) Speicherung von Objekten.

3. Mit welchem Schlüsselwort werden serialisierbare Klassen in Java gekennzeichnet?

A `serializable`

serialisieren/deserialisieren

1. Schreiben Sie in einer Datei `Student.java` eine Entitätsklasse `Student` mit den Attributen `vorname`, `nachname` und `matrikelnummer`.
2. Schreiben Sie in einer Datei `Serialisieren.java` eine Klasse `Serialisieren` die eine Hauptmethode enthält. In der Hauptmethode soll ein Objekt von Student erzeugt werden.
3. Schreiben Sie eine Methode `serialisieren()`, die als Parameter ein Objekt von `Student` erhält und dieses mit `ObjectInputStream.writeObject()` serialisiert.
4. Schreiben Sie eine Methode `deserialisieren()`, die `ObjectInputStream.readObject()` verwendet, um die persistent gespeicherten Daten wieder einzulesen und in einem Objekt von `Student` zu speichert. Geben Sie die Attribute des `Student`-Objekts aus.

serialisieren/deserialisieren

5. Tauschen Sie die persistent gespeicherte Datei mit einem ihren Kommilitonen und deserialisieren Sie seine Datei.
6. Ändern Sie im Programm den Attributnamen `matrikelnummer` in `matrikelnr`.

? *Lässt sich die Datei mit dem serialisierten Objekt nun noch deserialisieren?*

Datei- und Verzeichnisoperateionen

1. Schließen Sie alle zuvor erzeugten Dateien und erzeugen Sie eine Klasse `Dateioperationen` in einer Datei `Dateioperationen.java` mit einer Hauptmethode.

! *Öffnen Sie im Browser die Webseite `docs.oracle.com/javase/8/docs/api` und suchen Sie im Paket `java.io` nach der Klasse `File`. Suchen Sie dort in den nächsten Schritten nach den nötigen Methoden um die folgenden Aufgaben zu lösen.*

2. Erstellen Sie zwei Objekte vom Typ `File`. Eines mit dem Namen `dir` und eines mit dem Namen `file`.
3. Erzeugen Sie mit `dir` einen Ordner mit dem Namen `Unterverzeichnis`. Er soll automatisch gelöscht werden, wenn das Programm beendet wird.

Datei- und Verzeichnisoperateionen

! *Bei Bedarf können Sie mit* `System.in.read()` *den Programmablauf bis zum Drücken einer Taste anhalten.*

4. Erstellen Sie mit `file` eine Datei `Neue Datei.txt` in dem Ordner aus Aufgabe 3.
5. Listen Sie den Inhalt des Ordners auf.

Tipp: `for(String s : dir.list())`

6. Geben Sie aus, ob Schreibrechte für die zuvor erstellte Datei vorhanden sind.
7. Ändern Sie die Zugriffsrechte der Datei so, dass sie nur gelesen werden kann.
8. Geben Sie noch einmal aus, ob Schreibrechte vorhanden sind.
9. Löschen Sie die Datei.