II.4.4 Exceptions

Freitag, 16. Dezember 2016 17:00

- Exceptions können automatisch vom Laufzeitsystem geworfen werden.
- Exceptions müssen nicht zum Programmabbruch führen, sondern sie können in geeignetem Exception Handler gefangen werden
- try-Block wird abgebrochen, sobald Exception geworfen wird.
  Dann wird der erste catch-Block ausgeführt, der auf den geworfenen Exception-Typ passt.
  Der finally-Block wird auf jeden Fall ausgeführt, auch dann, wenn die Exception nicht gefangen wird.
- Exception Handler kann auf alle Variablen vor dem try-Block zugreifen und auf die gefangene Exception.
- Exceptions sind Objekte
- Error: sind Ausnahmen, von denen sich das Programm normalerweise nicht erholen kann u. die normalerweise zum Prog.-Abbruch führen sollten.
  RuntimeExceptions: sehr oft auftretende Exceptions

müssen nicht gefangen werden (unchecked exceptions)

Alle anderen Exceptions (checked exceptions) müssen gefangen werden.

- Exception-Objekte können mit Konstruktoren und new erzeugt werden, aber viele werden auch automatisch bei Fehlersituationen erzeugt.
- Bsp: A und B sind keine Unter-/Oberklassen.
  Da die Exc. vom Typ A in Methode M4 nicht gefangen wird, wird M4 abgebrochen u. man springt an die aufrufende Stelle (in Methode M3) u. wirft hier die Exc. vom Typ A.
- Programmierer kann eigene neue Exception-Klassen schreiben (U-Klassen von Throwable).
- Bsp: NegativeNumberException ist keine Unterklasse von RuntimeException ⇒ checked Exception, die gefangen werden muss
- Exception-Objekte können explizit geworfen werden (mit "throw").
  Falls eine Methode eine checked-Exception werfen könnte, die in dieser Methode nicht gefangen wird, dann muss dies im Methodenkopf angegeben werden (mit "throws").
- Exceptions haben Klassenhierarchie.
  Wenn im try-Block eine Exception geworfen wird, werden die catch-Blöcke von oben nach unten durchlaufen und der erste davon ausgeführt, dessen Typ passt.
  (Java Compiler erlaubt es nicht, wenn ein catch-Block

mit Oberklasse vor catch-Block mit Unterklasse kommt.)

- Bsp zur Demonstration von "finally":
  test ruft fak auf und fängt NegativeNumberExc., aber nicht die TooBigNumberException
  test hat einen finally-Block. Dieser wird immer ausgeführt, auch dann wenn test aufgrund einer TooBigNumberException abgebrochen wird.
  - Eingabe von 3: Fak von 3 ist 6
    Ende des try-catch Blocks
    Ende der Methode test.

  Eingabe von -3: Fehler! -3 < 0
  Ende des try-catch Blocks
  Ende der Methode test.

  Eingabe von 17: Ende des try-catch Blocks
  Fehler! Es trat die folgende Ausnahme auf ...

- Exceptions sollten für Ausnahmebehandlung verwendet, nicht für Sprünge, Fallunterscheidung etc.
- "throws" in Ober- und Unterklassen bei überschriebenen Methoden:
  - throws ... in Oberklasse
    Überschreibende Methode in U-klasse muss

kein throws haben (d.h. sie wirft ggf keine Exception)

- throws ... in Unterklasse in überschreibender Methode
  ⇒ man braucht auch throws ... in der Oberklasse