# Übungen zum Programmieren in Java

## Aufgabe 1 (P) Exceptions

Betrachten wir die verschiedenen Klassen zur Darstellung von Mengen von `Integer`–Objekten von Übungsblatt 3 und dort speziell die verschiedenen Realisierungen der Methode `insert()`.

a) Welche verschiedenen Fehlersituationen wurden dort „stillschweigend“ dadurch bereinigt, daß der Aufruf von `insert()` wirkungslos blieb?

b) Schreiben Sie eine Klasse `IntegerSetException` und für jede der in Teilaufgabe a) gefundenen Fehlersituationen eine entsprechende Unterklasse dazu.

   Definieren Sie dabei eventuell auch zusätzliche Konstruktoren, die bei geeignet gewählten Parametern passende Fehlermeldungen erzeugen.

c) Ändern Sie die Programme von Übungsblatt 3 so ab, daß

   (i) bei der Definition der `insert()`–Methoden geeignete Exceptions erzeugt werden und

   (ii) beim Aufruf der `insert()`–Methoden diese Exceptions geeignet behandelt werden.

## Aufgabe 2 (P) Applets und Threads

Eine bisher noch nicht behandelte Variante von Java-Programmen sind die sog. `Applets`. Applets unterscheiden sich von Java-Applikationen dadurch, dass sie nicht als eigenständige Prozesse, sondern in einem Browser laufen und deshalb auf WWW-Seiten erscheinen können.

Typischerweise sind einem Applet Rechte verwehrt, die für eine Applikation selbstverständlich sind, etwa der Zugriff auf das lokale Datei-System (Zur genauen Abstufung der Rechte von Java-Appliktionen und -Applets siehe [Bishop98, S.348f]).

Applets sind deswegen von so großem Nutzen, weil sie auf WWW-Seiten auch nicht-statische Information darstellen und (anders als cgi-Skripten) Berechnungen nicht nur auf dem Web-Server, sondern auch auf dem Client durchführen können.

In dieser Aufgabe soll eine Uhr programmiert werden, die die aktuelle Zeit anzeigt und nur solange Rechenzeit verbraucht, wie sie auf dem Bildschirm auch sichtbar ist. Kopieren Sie dazu das Programm-Gerüst

`Tag5/Clock.skelett`

in Ihr Arbeitsverzeichnis! Außerdem ist es hilfreich, wenn Sie zur Lösung der Aufgaben die API-Dokumentation der `Applet`-Klasse[1] in Ihren WWW-Browser laden.

[1] `http://medoc.informatik.tu-muenchen.de/Java/java11/api/java.applet.Applet.html`

a) Bevor Sie das Applet selbst vervollständigen, erstellen Sie bitte eine einfache WWW-Seite `clock.html`, die es beinhaltet und in der das Zeitintervall zwischen der Darstellung zweier Einzelbilder festgelegt wird!

   **Hinweis:** HTML-Tags für das Applet:

```
<APPLET CODE="klassendatei" WIDTH=n HEIGHT=m>
  <PARAM NAME="parametername" VALUE="wert">
</APPLET>
```

Ein Applet wird typischerweise programmiert, indem es von `java.applet.Applet` erbt. Für notwendige Initialisierungen schreibt man aber nicht eine neue Konstruktor-Methode, sondern überschreibt die Methode `init()`.

b) Ergänzen Sie `init()`, so dass es den auf der WWW-Seite festgelegten Parameter holt und den Wert der Instanzvariable `repaintInterval` zuweist!

Die Klasse `java.applet.Applet` ist eine Spezialisierung der der Klasse `java.awt.Panel`, die Sie von Java-Applikationen her kennen. Was auf die Bildfläche des Applets gezeichnet werden soll, schreibt man in die Methode `paint(Graphics g)`. `Graphics`[2] ist ein Graphik-Kontext, also ein Bildschirmbereich mit Methoden zum Zeichnen von Linien, Flächen und Text und zum Einstellen von Farbe und Font

c) Vervollständigen Sie die Methode `paint(Graphics g)`, indem Sie nach dem Aufruf von `setHandCoordinates()` die Uhrzeiger mit Hilfe der berechneten Koordinaten (`centerX`...`secondY`) als Linien zeichnen! Der Sekundenzeiger soll rot, die beiden anderen Zeiger blau gezeichnet werden. Zeichnen Sie außerdem den Uhr-Umriss als Kreis mit Durchmesser `clockSize`!

Im Gegensatz zu den gestrigen Aufgaben werden keine weiteren Graphikobjekte (Bottons etc.) zur Bildfläche des Applets hinzugefügt (obwohl dies analog zur `Panel`-Klasse auch möglich wäre). Stattdessen erzeugt und startet die Methode `start()` einen Thread, der das Zeichnen im Applet-Fenster erledigt. `start()` wird immer dann automatisch aufgerufen, wenn das Applet-Fenster auf dem Bildschirm sichtbar wird.

d) Was bewirkt die Standard-Methode `start()` eines *Threads*? Wie muss also die Methode heißen, die in einer Endlosschleife das `repaintInterval` abwartet und dann durch Aufruf von `repaint()` eine Neuzeichnung veranlasst? Welches Interface muss weiterhin die Klasse `Clock` implementieren, damit das Clock-Applet selbst als akuteller Parameter des `Thread`-Konstruktors verwendet werden kann?

   **Anmerkung:** Die Klasse `Applet` stellt die Methode `repaint()` zum Anstoßen einer Neuzeichnung zur Verfügung. Diese Methode teilt dem automatisch erzeugten und im Hintergrund laufenden Graphik-Thread den Wunsch nach einem Aufruf von `paint()` mit. Der Graphik-Thread kann die Neuzeichnungswünsche zusammenfassen, wenn diese zu zahlreich auftreten, und damit die Graphik-Ausgabe etwas optimieren.

[2] `http://medoc.informatik.tu-muenchen.de/Java/java11/api/java.awt.Graphics.html`

e) Starten Sie das Applet durch

```
appletviewer Clock.html
```

Sie können auch

```
appletviewer -debug Clock.html
```

aufrufen, wodurch das Applet in den JDK-Debugger[3] geladen wird. Im Debugger müssen Sie dann `run` eingeben, um das Applet zu starten. Mit dem Debugger-Kommando `threads` können Sie den momentanen Zustand der erzeugten Threads erfahren.

Wenn die Zeichenfläche des Applets nicht mehr sichtbar ist, wird automatisch die Methode `stop()` des Applets aufgerufen. In diesem Fall ist es dann auch nicht mehr nötig, laufend neu zu zeichnen, d.h. der `repaintThread` des `Clock`-Applets sollte gestoppt werden.

f) Implementieren Sie die Methode `stop()` entsprechend!

## Aufgabe 3 (P) Fleißaufgabe: Dokumentation von Java-Programmen mit `javadoc`

Fügen Sie Kommentare zur Klasse `EOrderedListIntegerSet.java` und deren Methoden hinzu, die von dem Tool `javadoc`[4] erkannt werden (Solche Kommentare müssen mit `/**` beginnen)! Machen Sie u.a. Gebrauch von den Tags `@param`, `@return`, `@exception` und `@see`!

Bedenken Sie bei Umlauten und Sonderzeichen, dass `javadoc` die Kommentare im HTML-Format erwartet!

Bei Zeitmangel können Sie sich auch einfach anschauen, welche HTML-Dateien

```
javadoc Clock.java
```

(ggf. mit Optionen `-package` oder `-private`) in Ihrem Verzeichnis produziert.

[3] http://medoc.informatik.tu-muenchen.de/Java/java11/tooldocs/solaris/index.html
[4] http://medoc.informatik.tu-muenchen.de/Java/java11/tooldocs/solaris/javadoc.html