

Übungsblatt 11

Abgabe bis Montag, 9.7.07, 11 Uhr

Hinweis:

Programmieraufgaben immer per E-mail (eine E-mail pro Blatt und Gruppe) an den zuständigen Tutor schicken (Java Quellcode und eventuell benötigte Datendateien). Bitte werfen Sie Ihre schriftlichen Lösungen in die Briefkästen in Geb. 051, Erdgeschoss ein. Für den Erhalt von Bonuspunkten müssen Sie in wenigstens 8 Übungen anwesend sein und müssen wenigstens 8 Übungszettel bearbeitet haben.

Die Java API Dokumentation finden Sie unter:

<http://java.sun.com/j2se/1.4.2/docs/api/>

Aufgabe 1

Bei der Arbeit mit Robotern oder anderen Sensorplattformen ist es hilfreich, empfangene Daten in einem sogenannten Logfile zu speichern. Häufig kommen dabei menschenlesbare Textdateien zum Einsatz.

Gehen Sie davon aus, dass Sie einen Roboter haben, der seine aktuelle Position einerseits anhand der Umdrehungen seiner Räder messen kann (sog. Odometriemessungen) und zusätzlich einen GPS Empfänger besitzt. Des Weiteren ist der Roboter mit einem Berührungssensor (sog. Bumper) ausgestattet, mit dessen Hilfe Kollisionen mit Hindernissen erkannt werden können. Jede Sensormessung, auch Nachricht genannt, besitzt eine eindeutige Zeichenkette (Bezeichner) zur Identifikation der Nachricht (z.B. ODOM, GPS, etc.) sowie einen Zeitstempel. Der Zeitstempel speichert dabei die vergangenen Sekunden (seit 1.1.1970, 00:00:00).

Jede Nachricht hat das folgende Format:

```
Bezeichner Zeit Rest
```

wobei der Rest erst durch die Nachricht selbst spezifiziert wird. Jede Nachricht steht in einer neuen Zeile in der Textdatei.

Es existieren die folgenden Nachrichten im angegebenen Format:

- ODOM Zeitstempel X Y Orientierung
- GPS Zeitstempel Longitude Latitude
- BUMPER Zeitstempel BumperID an/aus
- SPEED Zeitstempel Translationsgeschw. Rotationsgeschw.

Beispiel eines Logfiles:

```
GPS 0.0 1321321.0 42342.1
ODOM 0.0 0.0 0.0 0.0
SPEED 0.0 1.0 0.0
BUMPER 0.5 0 1
ODOM 1.0 1.0 0.0 0.0
SPEED 0.0 0.5 0.0
ODOM 2.0 1.5 0.0 0.0
GPS 2.0 1321321.1 42343.0
BUMPER 2.2 0 0
```

Ihre Aufgabe ist es nun ein System zum Einlesen, Verarbeiten und Schreiben solcher Logfiles zu erstellen. Nutzen Sie dabei Techniken wie Klassenhierarchie, Vererbung, Polymorphismus, etc.

1. Geben Sie eine Klassenhierarchie an, in dem Sie diese als Graphen visualisieren. Zeichnen Sie ein Rechteck für jede Klasse und einen Pfeil für jede Vererbung (jeweils von der Subklasse zur Superklasse).
2. Setzen Sie Ihre Klassenhierarchie in Java um und implementieren Sie alle benötigten Funktionen.
3. Schreiben Sie eine Methode, die alle Nachrichten eines Logfiles einliest und in einem `Vector` speichert.
4. Schreiben Sie eine Methode, die alle Nachrichten, die in einem `Vector` gespeichert sind, in eine Datei schreibt.
5. Implementieren Sie das Interface `Comparable` für Ihre Nachrichten-Klassen. Die Vergleichsmethode soll dazu verwendet werden, ein Logfile nach dem Zeitstempel zu sortieren.
6. Implementieren Sie eine Sortiermethode Ihrer Wahl, um ein Logfile zu sortieren.
7. Schreiben Sie eine Methode, die zwei Logfiles zusammenfügt und zeitlich korrekt in eine neue Datei schreibt.