2013

Nebelläufige Roboterfabrik

Klepp,Vogt
4AHIT
10.10.2013

Aufgabenstellung

Es soll eine Spielzeugroboter-Fabrik simuliert werden. Die einzelnen Bestandteile des Spielzeugroboters (kurz Threadee) werden in einem Lager gesammelt. Dieses Lager wird als Verzeichnis und die einzelnen Elementtypen werden als Files im Betriebssystem abgebildet. Der Lagermitarbeiter verwaltet regelmäßig den Ein- und Ausgang des Lagers um Anfragen von Montagemitarbeiter und Kunden zu beantworten. Die Anlieferung der Teile erfolgt durch Ändern von Files im Verzeichnis, eine Lagerung fertiger Roboter ebenso.

Ein Spielzeugroboter besteht aus zwei Augen, einem Rumpf, einem Kettenantrieb und zwei Armen.
Die Lieferanten schreiben ihre Teile ins Lager-File mit zufällig (PRNG?) erstellten Zahlenfeldern. Die Art der gelieferten Teile soll nach einer bestimmten Zeit gewechselt werden.

Die Montagemitarbeiter müssen nun für einen "Threadee" alle entsprechenden Teile anfordern und diese zusammenbauen. Der Vorgang des Zusammenbauens wird durch das Sortieren der einzelnen Ganzzahlenfelder simuliert. Der fertige "Threadee" wird nun mit der Mitarbeiter-ID des Monteurs versehen.

Es ist zu bedenken, dass ein Roboter immer alle Teile benötigt um hergestellt werden zu können. Sollte ein Monteur nicht alle Teile bekommen, muss er die angeforderten Teile wieder zurückgeben um andere Monteure nicht zu blockieren. Fertige "Threadee"s werden zur Auslieferung in das Lager zurück gestellt.

Alle Aktivitäten der Mitarbeiter muss in einem Logfile protokolliert werden. Verwenden Sie dazu Log4J [1].

Die IDs der Mitarbeiter werden in der Fabrik durch das Sekretariat verwaltet. Es dürfen nur eindeutige IDs vergeben werden. Das Sekretariat vergibt auch die eindeutigen Kennungen für die erstellten "Threadee"s.

Beachten Sie beim Einlesen die Möglichkeit der Fehler von Files. Diese Fehler müssen im Log protokolliert werden und entsprechend mit Exceptions abgefangen werden.

Tipps und Tricks

Verwenden Sie für die einzelnen Arbeiter das ExecutorService mit ThreadPools. Achten Sie, dass die Monteure nicht "verhungern". Angeforderte Ressourcen müssen auch sauber wieder freigegeben werden.

Beispiel für Teile-Files

-- "auge.csv"
Auge,11,24,3,4,25,6,8,8,9,10,11,12,13,14,15,16,17,18,195,5
Auge,91,62,3,4,54,6,7,8,9,10,11,12,13,14,15,16,17,18,119,32
Auge,91,62,3,4,54,6,7,8,9,10,11,12,13,14,15,16,17,18,119,520

-- "rumpf.csv"
Rumpf,91,62,3,4,54,6,7,8,9,10,11,12,13,14,15,16,17,18,119,21

Beispiel für Threadee-File

-- "auslieferung.csv"
Threadee-ID123,Mitarbeiter-
ID231,Auge,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,Auge,1,2,3,4,5
,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,Rumpf,1,2,3,4,5,6,7,8,9,10,11,12,13
,14,15,16,17,18,19,20,Kettenantrieb,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,
18,19,20,Arm,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,Arm,1,2,3,4,
5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
Threadee-ID124,Mitarbeiter-
ID231,Auge,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,Auge,1,2,3,4,5
,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,Rumpf,1,2,3,4,5,6,7,8,9,10,11,12,13
,14,15,16,17,18,19,20,Kettenantrieb,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,
18,19,20,Arm,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,Arm,1,2,3,4,
5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20

Ausführung

Zu bedenken sind die im Beispiel angeführten Argumente. Diese können mit eigenem Code oder mit einer CLI-Library implementiert werden (z.B. [2]).
Alle Argumente sind verpflichtend und die Anzahl muss positiv sein. Die obere Grenze soll sinnvoll festgelegt werden. Vergessen Sie auch nicht auf die Ausgabe der Synopsis bei einer fehlerhaften Eingabe! Sollten Sie zusätzliche Argumente benötigen sind diese erst nach einer Rücksprache implementierter.

java tgm.sew.hit.roboterfabrik.Simulation --lager /verzeichnis/zum/lager --logs /verzeichnis/zum/loggen --lieferanten 12 --monteure 25 --laufzeit 10000

Resources

[1] http://logging.apache.org/log4j/2.0/manual/configuration.html
[2] http://commons.apache.org/sandbox/commons-cli2/manual/index.html

Gruppenarbeit

1-2,3-12,4-11,6-23,7-10,8-19,9-17,13-21,14-18,15-16,20-22

Change-Request (2013-10-04 um 11:00)

Der Spielzeugroboter erhält neue Teile! Jeder Threadee erhält nun eine zusätzliche Antenne und zwei Greifer für die Arme. Der Lagermitarbeiter wird aufgewertet - alle Lieferanten müssen ihre Teile dem Lagermitarbeiter übergeben und dürfen nicht mehr selbständig ins Lager schreiben. Abgabe wurde auf Donnerstag, 10. Oktober 23:55 verschoben.

Arbeitsverteilung :

Geplant:

Aufgabe	Aufwand [h]	Wer
Teil Arm Auge Rumpf Kettenantrieb	1	Andi
Fabrik	3	Andi
FileQueue	3	Jakob
Kunde	1	Andi
Lager	3	Jakob
LagerMitarbeiter	2	Jakob
Lieferant	1	Jakob
MontageMitarbeiter	3	Andi
Sekretariat	1	Andi
Simulation	2	Jakob
SpielzeugRoboter	1	Andi
Einlesen CLI Library	3 x 2	Andi Jakob

Arbeitsaufwand Vogt

Task	Date	From	To	Duration
Uml-Session	2013-09-20	16:00	18:40	1:20
Uml-Session2	2013-09-26	13:30	14:30	1:00
JavaDocs Kommentare	2013-09-26	14:40	16:40	2:00
Andi IDE einrichten	2013-09-27	10:00	11:20	1:20
Simulation/Fabrik UML	2013-09-27	13:20	13:40	1:20
Teil	2013-09-28	16:00	18:00	2:00
Montagemitarbeiter	2013-10-02	15:00	19:00	4:00
Kunde	2013-10-02	19:00	20:00	1:00
SpielzeugRoboter	2013-10-03	15:00	16:15	1:15
JUnit teil	2013-10-03	16:15	17:15	1:00
Fabrik	2013-10-03	17:15	19:45	2:30
Verbesserungen	2013-10-03	19:45	21:00	1:15
NeueTeile	2013-10-08	13:30	15:00	1:30
Montagemitarbeiter	2013-10-08	15:00	16:30	1:30
Spielzeugroboter	2013-10-08	16:30	17:30	1:00
JUNIT	2013-10-08	17:30	19:00	1:30
Fabrik	2013-10-09	14:30	17:00	2:30
Verbesserungen	2013-10-09	17:00	19:00	2:00
		Insgesamt		30

Arbeitsaufwand Klepp

Task	Date	From	To	Duration
Uml-Session	2013-09-20	16:00	18:40	1:20
Uml-Session2	2013-09-26	13:30	14:30	1:00
JavaDocs Kommentare	2013-09-26	14:40	16:40	2:00
Andi IDE einrichten	2013-09-27	10:00	11:20	1:20
Simulation/Fabrik UML	2013-09-27	13:20	13:40	1:20
Ant Buildfile	2013-09-27	13:40	14:10	0:30
Ant	2013-09-30	21:40	01:15	3:35
Junit FileQueue	2013-10-01	13:00	15:15	2:15
FileQueue	2013-10-01	15:15	17:08	1:53
Logging	2013-10-01	17:08	20:50	3:42
Lager	2013-10-03	19:00	19:45	0:45
Verwaltung	2013-10-03	19:45	19:55	0:10
Lieferant	2013-10-03	20:10	20:40	0:30
LagerMitarbeiter	2013-10-03	20:40	21:20	0:40
Simulation	2013-10-03	21:20	21:30	0:10
Logging LagerMitarbei	2013-10-03	21:30	22:15	0:45
Ant JavaDoc	2013-10-03	22:15	22:35	0:20
Fabrik	2013-10-03	22:35	22:55	0:20
FileQueue 1/2 brainafk	2013-10-08	13:30	19:15	5:45
Lager	2013-10-09	08:40	08:50	0:10

Ingesamt 30

Persönlicher Fortschritt der Gruppe

Wir haben am Fr 20.9.13 damit begonnen ein UML Diagramm zu erstellen.

Nach ein paar Meinungsverschiedenheiten haben wir uns auf ein UML geeinigt(Siehe UML Dokument) und die Arbeitsaufteilung erstellt.

Jeder sollte ca. 15STD arbeiten um fertig zu werden
Meistens haben wir uns Mo-Fr nach dem Unterricht zusammengesetzt und gearbeitet.

Wir sind einigermaßen gut mit den einzelnen Klassen vorangekommen aber am Schluss sind ein paar Fehler endstanden in die wir einige Zeit investieren mussten. (Siehe Arbeitsaufwand).

Das Programm war am Do 3.10.13 nicht fertig da es einige Bugs gab . Nach dem Change Request haben wir die meisten Bugs gelöst und die neuen Aufgaben wurden ins Programm implementiert. Am Do den 10.10.13 konnten wir alle schwerwiegenden Fehler aus dem Programm entfernen und das Fertige Projekt abgeben. Wir haben jeweils ca. 30 Std gebraucht.