Design Patterns SS 2015
Zusatzhausaufgabe

Die Aufgaben müssen von jedem Teilnehmer einzeln bearbeitet und abgegeben werden. Die Abgabe muss **bis spätestens Montag 03.08.2014 um 23:59 Uhr** über unser Hausaufgabenverwtungssystem **https://se.cs.uni-kassel.de/hms/** erfolgen (nicht per Email). Die Abgabe ist nur als einzelne *.zip Datei möglich. Daher müssen alle für eine Abgabe relevanten Daten zu einem solchen Archiv kombiniert werden.

Hinweise zur Abgabe:

- Die Hausaufgabe ist in Form von **einem** exportierten Eclipse Projekten abzugeben. Mit Hilfe der Eclipse Export Funktion ist es möglich mehrere Projekte **in einer zip-Datei** zu exportieren. Ist ein Projekt nicht korrekt exportiert, kann es bei der Korrektur nicht berücksichtigt werden (es bietet sich also an, den Import des exportierten Projektes auszuprobieren).

  WICHTIG: Benennen Sie ihre Projekte für diese Abgabe nach folgendem Schema:

  DPSS15_HA<a>_<b>_<Matrikelnummer>,

  wobei <a> für die aktuelle Hausaufgabe und <b> für die Aufgabennummer steht.
  Beispiel für Aufgabe 1:
  DPSS15_HA3_1_12345678.
- Libraries müssen in einen lib oder im Projekt eingefügt werden oder als weitere Projekt mit dem Namen DPSS15_HA<a>_libs_<Matrikelnummer>. Es darf keine Abhängigkeit zu anderen Projekten bestehen oder zu externen Libraries.

Allgemeines:
Orientiert euch für die Lösung der Aufgaben an den zugehörigen Übungen und Vorlesungen:
http://seblog.cs.uni-kassel.de/category/ss15/designpatternss15/

Zusammensetzung der Note:
Es wird n Hausaufgaben geben, von denen **n-2 Sinnvoll** bearbeitet werden müssen.
Sinnvoll bedeutet mindestens 50% der Punkte.
Am Ende ergibt sich die Note aus dem Durchschnitt n-2 besten Abgaben.
Zusätzlich wird es zwei Zusatzaufgaben geben mit denen fehlenden Abgaben ausgeglichen werden können.

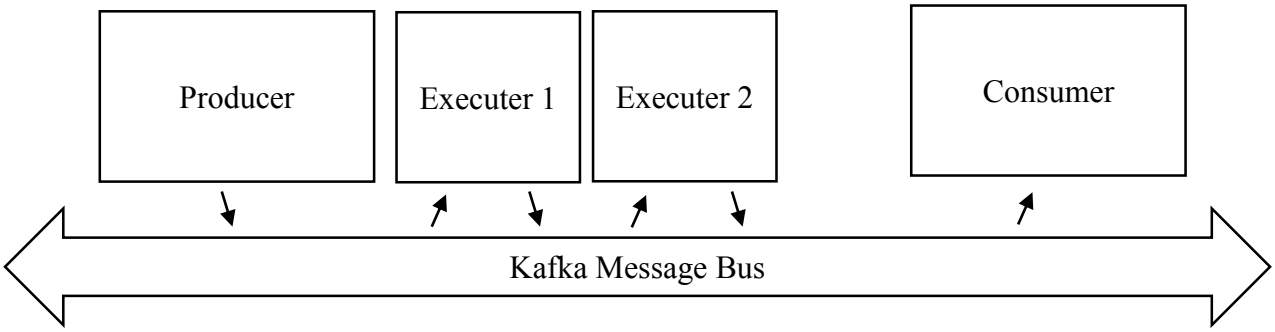
Die Zusatzaufgaben werden deutlich umfangreicher als normale Hausaufgaben.

Die Veranstaltung gilt als nicht bestanden, wenn mehr als **zwei Hausaufgaben nicht abgegeben** wurden oder der Mittelwert der zur Benotung herangezogenen Abgaben **weniger als 50%** beträgt.

Fragen bitte an hahn@uni-kassel.de

Zusatzaufgabe 1 Publish/Subscribe mit Kafka

https://kafka.apache.org (https://github.com/blackberry/Krackle/)



Erstellt mit Hilfe von Kafka ein lose gekoppeltes System, dass die folgenden Komponenten beinhaltet:

- Ein **Producer**, soll Zufallszahlen generieren und im Sekundentakt an einen Channel in den Message Bus schickt
    - Die Zufallszahl soll sich wie folgt berechnen:

value = Math.max(-10, Math.min(10, value + .8 * Math.random() - .4 + .2 * Math.cos(i += .2)));

    - achtete darauf, dass i nicht überlauft ;) beginnt mit i=0 und value = 0
    - Der zu verwendende Channel soll als Command Line Argument beim Start angegeben werden
- Ein **Executer** soll auf einen Channel horchen, eingehende Nachrichten wie folgt bearbeiten und anschließend auf einem anderen Channel ausgeben:

(inputValue) -> ((lastInputValue – inputValue) ; inputValue ; minValue ; maxValue)

    - Besitzt eine eingehende Nachricht mehrere Values, soll der erste Value der Liste verwendet werden
    - Beginnt mit lastInputValue = 0; minValue = 42; maxValue = -42
    - minValue und maxValue sollen entsprechend berechnet werden
    - lastInputValue soll entsprechend gesichert werden
    - Der zu verwendende Input- und Outputchannel sollen als Command Line Argument beim Start angegeben werden.
- Ein **Consumer** soll die eingehenden Nachrichten auf formatiert auf der Konsole ausgeben
    - Der zu verwendende Inputchannel soll als Command Line Argument beim Start angegeben werden.

(Alle Nachrichten sollen geeignet mit JSON kodiert werden)

Anschließend erstellt zwei Bashscripts, die folgende Systeme Starten:

System 1:

Producer --[channel1]--> Executer --[channel2]--> Consumer

System 2:

Producer --[channel23]--> Executer --[channel42]--> Executer --[channel1337]--> Consumer

Fragen bitte an hahn@uni-kassel.de

Zusatzaufgabe 2 Lose Kopplung

Schreibt einen weiteren Executer, der eine beliebige andere Berechnung ausführt.

Erstellt ein Bashscript, dass während eurer Systeme läuft einen alten Executer, mit einem neuen Executer austauscht. Der alter Executer Prozess wird einfach beendet (kill -9) und ein neuer gestartet.