Design Pattern SS 2013
Hausaufgabe 3

UNIKASSEL VERSITÄT

Die Aufgaben müssen einzeln bearbeitet und abgegeben werden. Die Abgabe muss **bis spätestens Montag, den 06.05.2013 um 23:59 Uhr** über unser Hausaufgabenabgabesystem **http://seblog.cs.uni-kassel.de/dpss13/** erfolgen. Die Abgabe ist nur als einzelne *.zip oder *.jar-Datei möglich. Daher müssen alle für eine Abgabe relevanten Daten zu einem solchen Archiv kombiniert werden.

Hinweise zur Abgabe:

- Die Hausaufgabe als exportiertes Eclipse Projekt (*.zip, nicht den gesamten Workspace) abgeben. Das kann mit Hilfe der Eclipse Export Funktion durchgeführt werden. Ist das 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 ihr Projekt für diese Abgabe nach folgendem Schema:

  DPSS13_HA<a>_<Matrikelnummer>,

  wobei <a> für die aktuelle Hausaufgabe steht.
  Beispiel:
  DPSS13_HA10_12345678.

Allgemeines
Orientieren Sie sich für die Lösung der Aufgaben an den zugehörigen Übungen und Vorlesungen:
http://seblog.cs.uni-kassel.de/category/currentterm/design-patterns/

Die Benotung beruht auf den Hausaufgaben.

Hierfür werden die gesamten Hausaufgaben minus zwei vom Studenten abgegebenen Hausaufgaben addiert, die mit mehr als 50 % bewertet wurden. Nicht abgegebene Hausaufgaben oder Betrugsversuche bekommen 0 Punkte.

Die Prüfung gilt als nicht bestanden, wenn alle Hausaufgabenpunkte weniger als **50 %** ergeben oder der Student mehr als **zwei Hausaufgaben nicht abgibt**.

Fragen bitte an Stefan.Lindel@cs.uni-kassel.de oder george@uni-kassel.de

Aufgabe 1 (Strategy Pattern) (3P)

- Baut ein Unisystem.
- Es sollen folgende Elemente vorhanden sein: Person, Strategie.
- Jede Strategie hat eine int Variable *value* die mit getValue() abgefragt werden kann.
- Es gibt vier Strategien, die nacheinander ausgetauscht werden sollen:
  - Vorlesung besuchen (value = 1)
  - Kaffee trinken (value = 2)
  - Lernen (value = 3)
  - ins Schwimmbad gehen (value = 4)
- Es soll ein sinnvoller Test geschrieben werden.

Fragen bitte an Stefan.Lindel@cs.uni-kassel.de oder george@uni-kassel.de

Aufgabe 2 (Hook Pattern) (6P)

- Baut ein Grillveranstaltungssystem mittels Hook Pattern
- In diesen Grillsystem gibt es die Klassen: Veranstaltung(Party), Person, Transaction und die entsprechenden Klassen für das Pattern.
- Die Methode für das Hook-Pattern:
  - paymentMovements(Person person, String event, float value)
    Beispiel für den Aufruf:
    Veranstaltung v = new Veranstaltung();

    ...

    v.paymentMovements(albert, "Bier", 3.99);
    v.paymentMovements(albert, "Würstchen", 9.99);
- Es sollen folgende Events (Strategien) implementiert und getestet werden:
  - Es soll eine Strategie nach dem Hook-Pattern entwickelt werden, in der die einzelne Transaction in die Liste der Transactions der Person hinzugefügt wird.
  - Eine zweite Strategie soll im Anschluss jedes Mal die Summe der bisherigen Transactions bilden und in der Person speichern.

  (Es wäre zusätzlich denkbar, dass die Strategien asynchron arbeiten und die Summe nicht nach jeder Transaction sondern immer nach einer gewissen Zeit gebildet wird. Z.B. bei einer langen Einkaufsliste mit 42 Artikeln.)

Fragen bitte an Stefan.Lindel@cs.uni-kassel.de oder george@uni-kassel.de

Aufgabe 3 (Chain of Responsibility) (3P)

- Baut euer Beispiel aus der Aufgabe 1 in eine Chain of Responsibility, inklusive aller Anpassungen der Strategieklassen, um.
- Ob eine Strategie benutzbar ist, hängt von der Note der letzten Hausaufgabe (Attribut in Person) ab.
    - Vorlesung besuchen (>= 50 %)
    - Kaffee trinken (== 100 %)
    - Lernen (>= 70 %)
    - ins Schwimmbad gehen(<50 %)
- Es soll ein sinnvoller Test geschrieben werden.

Fragen bitte an Stefan.Lindel@cs.uni-kassel.de oder george@uni-kassel.de