

Frage: Können wir zur Erzeugung von Klassendiagrammen nicht einfach die Generierungsmöglichkeit von UMLet oder einem anderen UML-Tool nutzen.

Antwort: Da Sie eigentlich erst das Klassendiagramm entwickeln und dann den Code sollte das für die OOAD-Hausarbeit kaum möglich sein. Generell können natürlich Generatoren genutzt werden, dann sind sie aber verantwortlich dafür, dass die korrekte UML-Syntax eingehalten wird. UMLet versagt da leider in vielen Punkten, ähnliches gilt auch für die mir bekannten kommerziellen Werkzeuge.

Nachfrage: Ok, aber das sind ja nur Details, auf die bei Zeitmangel verzichtet werden kann?

Antwort: Nein. Der Sinn der UML ist, dass jede Person die Diagramme exakt gleich interpretiert bezüglich der dargestellten Informationen. Da abhängig vom Detaillierungsgrad Informationen weggelassen werden können, kann es allerdings immer noch Interpretationsmöglichkeiten geben. Da sie generell lernen sollen die Wünsche auftraggebender Personen zu respektieren und Ihre Freiheitsgrade in den Umsetzungsmöglichkeiten zu sehen, könnten nur solche Personen Anforderungen als unwichtige Details klassifizieren.

Frage: Wir haben Schwierigkeiten ein konkretes Hausarbeitsthema zu finden.

(s. auch FA06) Die Aufgabenstellung ist bewusst offen formuliert, damit sie ein Thema finden, das sie interessiert. Die Idee ist dann, dieses Framework oder die Bibliothek auszuprobieren und dann eine kleine Aufgabenstellung zu definieren, die sie dann mit OOAD-Mitteln von der Analyse bis zur Implementierung durchziehen. Alternativ entwickeln Sie ein großes Projekt und setzen dann nach der Use Case-Modellierung nur das erste Inkrement um.

Wem das noch zu unkonkret ist, für den ist der Vorschlag sich eine Oberflächentechnologie in Java anzusehen, dann auf einzelne Steuerungselemente wie Knöpfe und Tabellen zu konzentrieren und dann dazu eine Beispielaufgabe zu definieren. Beispiele für Technologien sind: Swing (alt, etabliert, fester Java-Bestandteil), JavaFX (genauer OpenJFX, war mal Teil von Java, muss jetzt selbst ergänzt werden; ist aber in der Java-Version in der KleukerSEU enthalten), Google Web Toolkit (GWT, Web-Applikationen mit Java). Der Tipp ist, wenn Sie kein interessantes Framework kennen, etwas Zeit in Google investieren, bei ein zwei Technologien schauen, ob man das Konzept versteht und Hello-World-artiges zum Laufen bekommt und sich dann eine kleine Aufgabenstellung auszudenken.

Eine einfachere Aufgabenstellung ist die Neuimplementierung z. B. von Teilen des Collection-Frameworks von Java und ein Vergleich mit existierenden Implementierungen. Bei solcher einer Aufgabenstellung ist es eher unwahrscheinlich ein sehr gutes aber auch ein sehr schlechtes Ergebnis zu erreichen.

Frage: Es gibt keine Präsentation?

Antwort: Richtig, anders als beim Projektbericht gibt es bei der Hausarbeit keine Präsentation, so dass mehr Zeit zur Erstellung des Textes zur Verfügung steht. Am Ende existiert ein Textdokument, das der Prof von vorne nach hinten liest und bewertet. Der Arbeitsaufwand ist identisch für alle Module mit Leistungspunkten.

Hinweis Ich werde einzelne Beispiellösungen für Blatt 7 in Ilias im Dateibereich der Vorlesung hinterlegen.

Frage: Wieviele Aufgabenblätter gibt es?

Antwort: 11 bewertete.

Frage: Müssen wir Sequenzdiagramme in der Hausarbeit nutzen und sollen wir sie generieren lassen.

Meine Standardantwort bei „müssen“ ist meist: Sie müssen das nicht machen und ich muss ihnen auch keine sehr gute Note geben. Konkret hängt es hier später von der von Ihnen gewählten Aufgabenstellung ab, ob Sequenzdiagramme und wie viele sinnvoll sind. Wenn der Fokus z. B. auf der Kommunikation von GUI-Elementen liegt, gibt es oft ein zentrales Prinzip, so dass ein Diagramm reicht. Da im Sequenzdiagramm nur alle relevanten Klassen sichtbar sein sollen, kann man das Diagramm generieren oder auch von Hand zeichnen. Ein Beispiel ist die Klasse Messages in der Aufgabe 18, die für das Verständnis der MVC-Idee sicherlich nicht relevant ist und weggelassen wird. Ähnliches gilt für die EinUndAusgabe-Klasse, wobei die eigentliche Ausgabe wieder sehr wichtig ist, aber auch einfach mit einem Pfeil von der View-Klasse zu Extern mit der Ausgabe auf dem Pfeil dargestellt werden könnte.

Beachten Sie, dass meine Vorgaben nur das enthalten können, bei deren vollständiger Abarbeitung ein Ergebnis herauskommt, das die durchschnittlichen Erwartungen erfüllt. Das entspricht der Basisnote 3.0.

Das ist nebenbei der Grund, warum ich nicht noch detaillierter in die Aufgabenstellung einsteige. Würde ich das machen, wäre es immer schwerer eine (sehr) gute Note zu erreichen. Sehr gute Leistungen entstehen nie durch Pflichterfüllung.

Zum Clonen hier noch ein kleines weiteres Beispiel, warum eine einfache rekursive Nutzung von clone() nur unter bestimmten Randbedingungen zum Ergebnis führt.

```

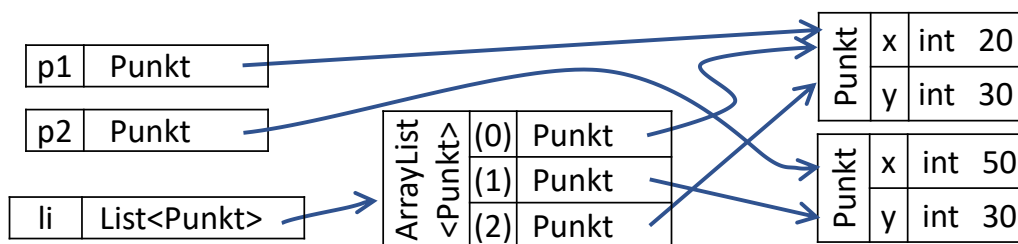
6 public class Beispiel {
7
8     public static List<Punkt> kopieren(List<Punkt> liste){
9         List<Punkt> ergebnis = new ArrayList<>();
10        for(Punkt e:liste) {
11            if (e == null) {
12                ergebnis.add(null);
13            } else {
14                ergebnis.add(e.clone());
15            }
16        }
17        return ergebnis;
18    }
19
20    public static void main(String[] s) {
21        List<Punkt> li = new ArrayList<>();
22        Punkt p1 = new Punkt(20,30);
23        Punkt p2 = new Punkt(50,30);
24        li.add(p1);
25        li.add(p2);
26        li.add(p1);
27        System.out.println("" + (li.get(0) == li.get(2)));
28        List<Punkt> li2 = kopieren(li);
29        System.out.println("" + (li2.get(0) == li2.get(2)));
30    }
31 }

```

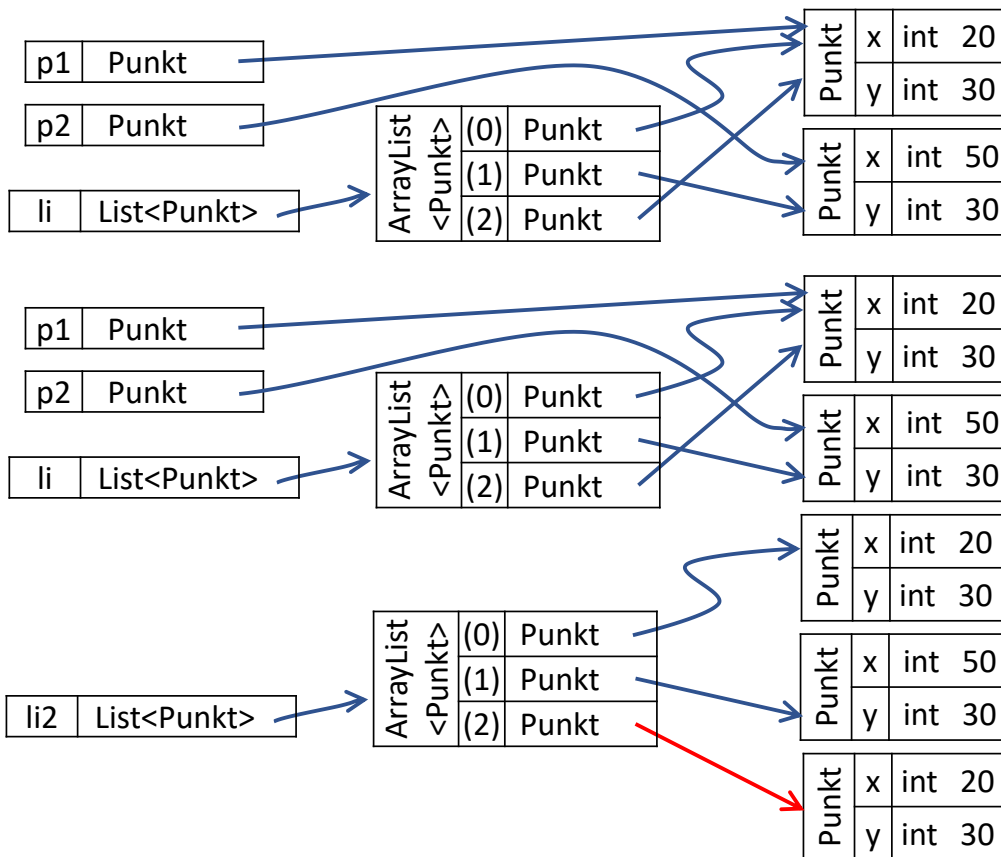
Problems @ Javadoc Console Declaration Coverage

<terminated> Beispiel [Java Application] C:\kleukersSEU\java\bi
true
false

Die Methode kopieren() kopiert einfach jedes existierende Element. In der Liste li befinden sich identische Elemente an den Positionen 0 und 2, wie auch das folgende Objektspeicherdiagramm zeigt.



Beim Erstellen der Kopie stehen dann nur gleiche Objekte an den Positionen 0 und 2.



Als Lösung kann man sich schon geclonete Objekte z. B. in einer zu übergebenden HashMap merken und vor jedem Clonen prüfen, ob das Objekt schon geclonet wurde und ggfls. dieses als Ergebnis liefern. Die Klassen XMLEncoder und XMLDecoder stellen dies durch Referenzen in XML sicher, wie der folgende Ausschnitt zeigt. Alternativ können hier Binär-Kopien zu schnellen sauberen Clone-Ergebnissen führen.

```

<void property="Liste1">
  <void method="add">
    <object class="entity.Punkt" id="Punkt0">
      <void property="x">
        <int>20</int>
      </void>
      <void property="y">
        <int>30</int>
      </void>
    </object>
  </void>
  <void method="add">
    <object class="entity.Punkt">
      <void property="x">
        <int>50</int>
      </void>
      <void property="y">
        <int>30</int>
      </void>
    </object>
  </void>
  <void method="add">
    <object idref="Punkt0"/>
  </void>
</void>

```

Frage: Wir haben es mit Streams beim Clonen versucht, da ging aber gar nichts.

Achtung, es gibt zumindest zwei Varianten, in denen in Java der Begriff Stream benutzt wird. Die klassische Variante ist auch in den Beispiellösungen zum Clonen enthalten und nutzt Klassen wie `FileInputStream`, `FileOutputStream` oder die 16-Bit-Varianten `FileReader` und `FileWriter`. Dies sind alles typische Klassen die eine Dateibearbeitung ermöglichen.

Der Stream-Begriff ist dann in anderer Form bei der Einführung von Lambda-Ausdrücken und der funktionalen Programmierung in Java erneut relevant, z.B.

```
int[] arr= {9,7,3,1};  
Arrays.stream(arr)  
    .forEach(i -> System.out.println(i));
```

Es können u. a. Collections in eine Stream-Verarbeitung gesteckt werden, mit der jedes Objekt des Streams schrittweise in der danach beschriebenen Form verarbeitet wird. Diese Streams können dabei beliebig lang sein und sehr unterschiedliche Quellen haben. Das Thema „Einführung in die reaktive Programmierung in Java anhand eines Beispiels“ ist in diesem Zusammenhang auch für Hausarbeiten wählbar.

Diese kurze Übersicht soll zur Vermutung führen, dass die beiden Stream-Begriffe in der Nutzung direkt auf wesentlich andere Anwendungsszenarien zielen, was ein sinnvoller Basisgedanke ist. In einer abstrakteren Sichtweise stellt sich die Frage, warum dann der gleiche Begriff gewählt wurde. Die Vermutung, dass jede Art von Byte-Stream auch als funktionaler Stream betrachtet werden kann, ist dabei sinnvoll. Es gibt also auch funktionale Möglichkeiten Dateien zu lesen und zu schreiben, was allerdings beim Clone meiner Erfahrung nach nicht hilft (?).

```
public static void main(String[] args) {  
    try(Stream<Path> projekt = Files.list(Path.of("../"))) {  
        projekt.filter(Files::isDirectory)  
            .forEach(System.out::println);  
    } catch (IOException e) {  
        System.out.println("Exception: " + e);  
    }  
}
```