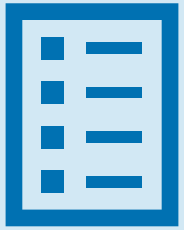
JAVA Programmierung

ECLIPSE & JAVA 8

Themenübersicht



01

Objekte

1. Generelles
2. Garbage Collector
3. Beispiele

02

Aufgabe

1. Aufgabe
2. Lösung

01 Objekte

1. *Generelles*
2. *Garbage Collector*
3. *Beispiele*

1.1 Objekte

Generelles

1.1 Objekte

- Generelles

Klassen dienen als Vorlage (Baupläne) für Objekte. Konstruktoren definieren den Bauplan und der „new“ Operator löst diesen Bauauftrag aus. Am ende bilden sie mit dieser Kombination Instanzen einer bestimmten Klasse. Man spricht daher auch von der Instanziierung. Am Ende wird somit der Klassenname zum Typ der Variable.

```
Person person = new Person();
```

Letztendlich dienen Objekte zur Speicherung von Klassen spezifischen Eigenschaften innerhalb des Lebenszyklus eines Programms. Bei dem Beispiel Person könnte das der Name und Alter sein, welche Attribute (Eigenschaften) darstellen.

```
public class Person {

    private String name;
    private int alter;
```

1.1 Objekte

- Generelles

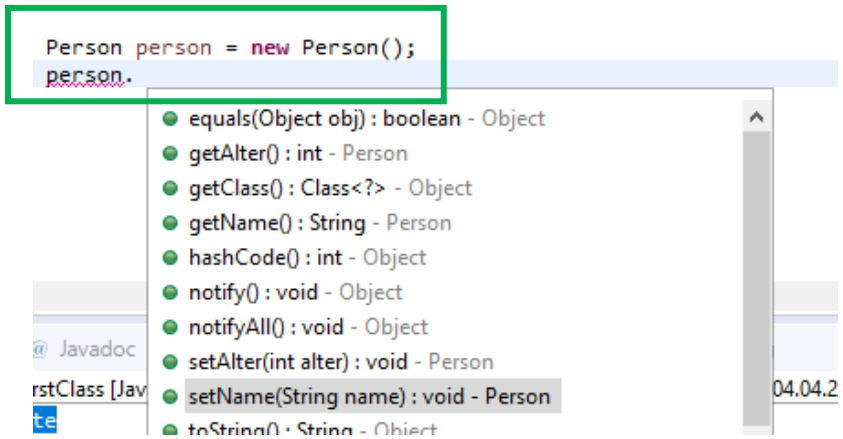
Ein wichtiger Bestandteil der OOP (Objekt Orientierten Programmieren) ist das manipulieren des jeweiligen Objektes. Den Mechanismus hierfür kennen Sie schon => Methoden. Um die Analogie zum Bauplan herzustellen

=> Während des Baus sollen Änderungen vorgenommen werden

Methoden der Klasse zum Manipulieren

```
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public int getAlter() {
    return alter;
}
public void setAlter(int alter) {
    this.alter = alter;
}
```

Anwenden der jeweiligen Methode



1.1 Objekte

- Generelles

Besondere Methoden in Java sind Getter und Setter. Diese manipulieren Attribute (Eigenschaften) des jeweiligen Objekts.

- Der Getter ermöglicht es, auf ein Attribut, auch Außerhalb der Klasse, zuzugreifen
- Der Setter ermöglicht es, Attribute, auch Außerhalb der Klasse, zu manipulieren

```
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public int getAlter() {
    return alter;
}
public void setAlter(int alter) {
    this.alter = alter;
}
```

Generell sind diese Methoden dazu gedacht, auf Eigenschaften (Attribute), auch bei eingeschränkter Sichtbarkeit (z.B. private => nur innerhalb der Klasse Sichtbar), außerhalb der Klasse Zugreifbar oder Manipulierbar zu machen.

1.2 Objekte

Garbage Collector

1.2 Objekte

- Garbage Collector

Der Garbage Collector dient zur Bereinigung des Speicher der JVM (Java Virtuellen Machine). Dieser Unterteilt sich grob in 2 Ebenen, den Heap und Stack.

Als Erstes muss mit erwähnt werden das der Garbage Collector nicht immer das macht was man will. Selbst bei einem expliziten Auffordern entscheidet dieser selbst, wann der Auftrag ausgeführt wird. Da dieser von der JVM gestellt wird, erfolgt der Aufruf über System.

```
System.gc();
}
```

Beim Ausführen einer Anwendung teilt die JVM den Speicher in Heap und Stack Speicher auf. Alles was während der Ausführung erstellt oder Aufgerufen wird => Methoden, Variablen, Objekte oder auch einfache Operationen landen entweder im „Heap Space“ oder dem „Stack Memory“.

1.2 Objekte

- Garbage Collector

Stack Memory :

Dieser speichert Methoden spezifische primitive Werte, sowie Verweise auf Objekte, auf die von der Methode verwiesen wird, welche sich im Heap befindet.

Der Stack (Stapel) arbeitet nach dem Prinzip des LIFO => Last In First Out. Sobald eine neue Methode aufgerufen wird landet ein Block auf dem Stapel. Dieser enthält primitive Werte und Objekt Verweise. Nach beenden der Methode wird der Block im Stack gelehrt und Platz gemacht.

- Sollte der Speicher voll sein, wirft er eine *java.lang.StackOverFlowError*
- Variablen existieren innerhalb des Speichers nur so lange wie diese Ausgeführt wird
- Speicherplatz wird automatisch nach beenden der Methode freigegeben

1.2 Objekte

- Garbage Collector

Heap Space :

Dieser speichert Objekte zur Laufzeit. Im Heap Space werden neue Objekte erstellt, und die Referenzen zu diesen im Stack.

Der Heap kann in kleiner Teile zerlegt werden. Hier eine kleiner Einblick

- **Young Generation** - dient der Zuordnung neuer Objekte. Diese lagern dort und „altern“ im Sinne des Lebenszyklus eines Programms. Sollte dieser voll sein, wird der Garbage Collector aufgerufen
- **Old Generation** - Objekte welche in der Young Generation gespeichert sind, erhalten einen Schwellenwert für ihr Alter, ist dieser erreicht wird es in die Old Generation verschoben
- **Permanete Generierung** - hier werden Laufzeitklassen und JVM Metadaten abgelegt

1.2 Objekte

- Garbage Collector

Heap Space :

- Sobald der Heap Speicher voll ist, gibt es einen *java.lang.OutOfMemoryError*
- Im Gegensatz zum Stack wird der Heap Speicher nicht automatisch freigegeben. Sie benötigen den Garbage Collector um abgelaufene Objekte zu löschen. Um Objekte für den Garbage Collector vorzubereiten müssen diese auf null gesetzt werden.
- Im Vergleich zum Stack ist der Heap nicht threadsicher und muss deshalb Synchronisiert werden. Hierzu im Thema Threads mehr.

1.3 Objekte

Beispiele

1.3 Objekte

- Beispiel

```
public static void main(String[] args) {

    Kuchenstück kuchenstück = new Kuchenstück(100);
    ArrayList<Kuchenstück> kuchenStueckListe = new ArrayList<>();
    for(int i = 0; i< 12; i++) {
        kuchenStueckListe.add(new Kuchenstück(10 * (i +1)));
    }
    Kuchen kuchen = new Kuchen(kuchenStueckListe);
```

1.3 Objekte

- Beispiel

```
2  public class GarbageCollection {
3      public void finalize() {
4          System.out.println("der GC war da");
5      }
6
7      public static void main(String args[]) {
8
9          GarbageCollection s1 = new GarbageCollection();
.0         GarbageCollection s2 = new GarbageCollection();
.1         s1 = null;
.2         s2 = null;
.3         System.gc();
.4     }
.5 }
```

Um Objekte für den Garbage Collector frei zu geben müssen diese zuvor auf null gesetzt werden, da sonst ihr Lebenszyklus noch nicht beendet ist.

02 Aufgaben

1. *Aufgabe*
2. *Lösung*

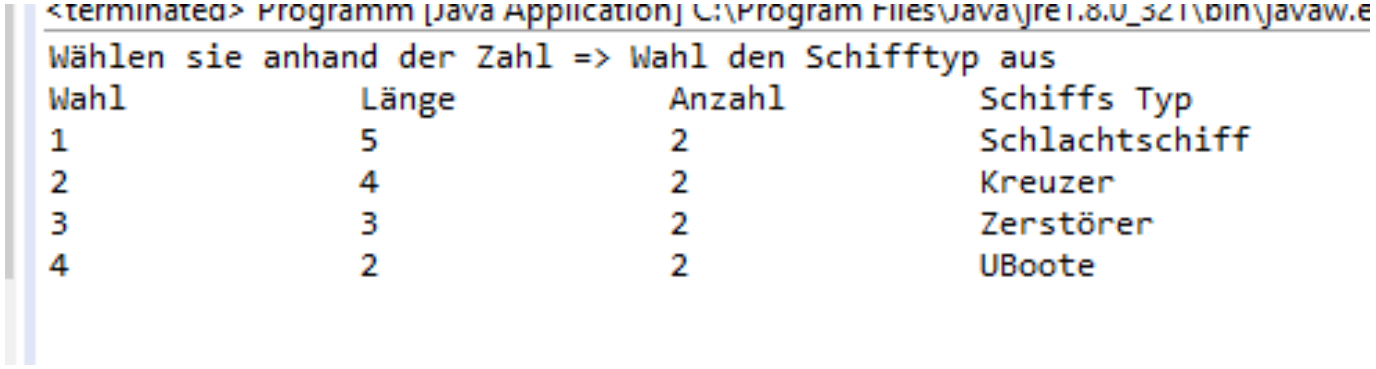
2.1 Aufgaben

Aufgabe

2.1 Aufgabe

Baustein Projekt =>
Implementiere die ersten Schritte der Spiellogik

- **Abfrage über die Konsole welcher Schiffstyp platziert werden soll**
  - **Aktualisieren der noch verfügbaren Anzahl der Schiffe**



2.1 Aufgaben

Lösung

2.2 Lösung

BausteinProjekt_V4.7z => in der Cloud

VIELEN DANK!