Neue Themen für die 808

Februar 2025

SMART INDUSTRY CAMPUS

Plan für die Woche

Montag	Dienstag	Mittwoch	Donnerstag	Freitag
• Wiederholung Lambdas und Predicate • ArrayLists • ArrayLists und Lambdas	• Wrapper-Klassen • Block an Quizfragen (15-20)	• Java class structure • Command line • Selbständiges Vorbereiten zur Prüfung anhand Lernplans	• Statische Variablen und Methoden • Selbständiges Vorbereiten zur Prüfung anhand Lernplans	• Features vergleichen • Block an Quizfragen (15-20)

SMART INDUSTRY CAMPUS

Plan für heute

- Klassenstruktur (inklusive packages und import statements)
- Commandline
- Selbsteinschätzung nochmal ausfüllen
- eines der schwächeren Themen nochmal lernen

SMART INDUSTRY CAMPUS

Klassenstruktur

SMART INDUSTRY CAMPUS

Aufbau einer Java-Klasse

- package-Deklaration, wenn die Java-Datei in einem package ist
- import-Anweisung, wenn nötig (z.B. importieren von ArrayList)
- Klassendefinition
- Attribute, Methoden

SMART INDUSTRY CAMPUS

Packages

- helfen den Code zu strukturieren und Kollisionen von Klassennamen zu vermeiden
- eine Datei kann **KEINE** oder nur **EINE** package-Deklaration enthalten
  - Eine Datei gehört zu einem Package
  - Der package-Name muss angegeben werden

```
package;          //invalide
```

- package-Deklaration muss vor der Klassendefinition und import-Anweisungen stehen

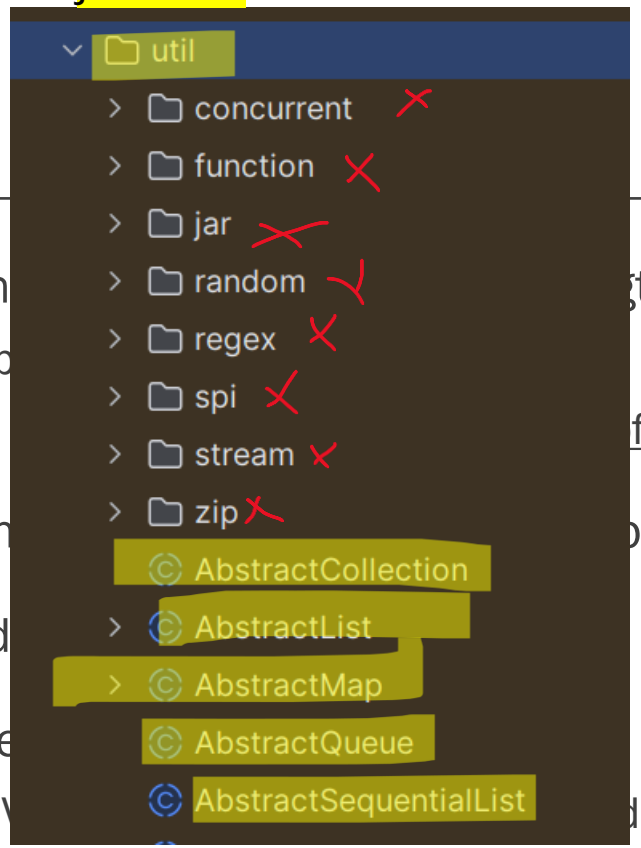
SMART INDUSTRY CAMPUS

Imports

- wenn eine Klasse aus einem anderen Package benötigt wird, muss sie importiert werden
  - import java.util.List; für spezifische Klassen.
  - import java.util.*; für alle Klassen eines Pakets (nicht empfohlen wegen schlechter Lesbarkeit).
- es dürfen keine oder mehrere import-Anweisungen geben
- es werden keine Unterordner importiert!

Imports

java.util.



- wenn eine Klasse aus ein … gt wird, muss sie importiert werden
  - import java.util.List; für sp…
  - import java.util.*; für alle … fohlen wegen schlechter Lesbarkeit).
- es dürfen keine oder meh… pen
- es werden keine Unterord…
- es können keine Methode…
  - Statische Methoden oder … den

SMART INDUSTRY CAMPUS

Imports

- wenn eine Klasse aus einem anderen Package benötigt wird, muss sie importiert werden
  - import java.util.List; für spezifische Klassen.
  - import java.util.*; für alle Klassen eines Pakets (nicht empfohlen wegen schlechter Lesbarkeit).
- es dürfen keine oder mehrere import-Anweisungen geben
- es werden keine Unterordner importiert!
- es können keine Methoden importiert werden
  - Statische Variablen können importiert werden

Imports

- wenn eine Klasse au tiert werden
  - import java.util.List;
  - import java.util.*; fü Lesbarkeit).
- es dürfen keine oder
- es werden keine Unt
- es können keine Me
  - Statische Variablen

```
import static java.lang.Integer.MAX_VALUE;
import static java.lang.System.*;

public class Test {
    public static void main(String[] args) {
        out.println(MAX_VALUE);
    }
}
```

SMART INDUSTRY CAMPUS

Klassendefinition

- innerhalb einer Datei darf nur eine öffentliche (public) Klasse geben
  - Diese muss den gleichen Namen wie die Datei haben
- innerhalb einer Datei dürfen weitere Klasse definiert werden, solang sie nicht public sind
  - Auch Enums und Interfaces können innerhalb einer Datei definiert werden
- Schleifen, System.out.print(),... müssen innerhalb einer Methode stehen!

Klassendefinition

- innerhalb einer Date
  - Diese muss den gleic
- innerhalb einer Date sie nicht public sind
  - Auch Enums und Inte
- Schleifen, System.ou en!

```
public class Test {

    for(int i = 0; i<10; i++){

    }

    public static void main(String[] args) {
        out.println(MAX_VALUE);
    }
}
```

SMART INDUSTRY CAMPUS

Command Line

SMART INDUSTRY CAMPUS

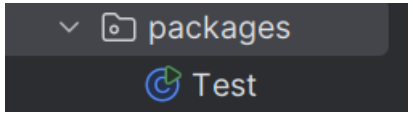
Kommandozeile

- kann zur Ausführung von Java-Programmen dienen
- zuerst muss eine Datei mittels javac Dateiname.java kompiliert werden
  - Daraus resultiert eine .class-Datei
  - Die Datei wurde in bytecode umgewandelt
- zum Ausführen der Datei nutzt man den Befehl: java Klassenname
- sowohl javac als auch java können großgeschrieben werden
  - Bytecode wird ausgeführt
  - Javac und Java
  - JAVAC und JAVA

Ausführen von Code in Packages

- das Ausführen von Code, welche in packages sind, ist etwas umständlicher
  - Javac –d . packagePath/Dateiname.java
  - Java packageName.Klassenname

-d steht für Destination und gibt an, wohin die kompilierten .class-Dateien gespeichert werden sollen




```
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> javac -d .  packages/Test.java
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> java packages.Test
2147483647
```

SMART INDUSTRY CAMPUS

Codevorführung

Komma kann genutzt werden, um Argumente zu trennen

```java
public class StringCommando {
    public static void main(String[] args) {
        System.out.println(args[0] + args[1]);
    }
}
```

```
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> javac -d.  packages/StringCommando.java
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> java packages.StringCommando Hallo mein Name ist Anita
Hallomein
```

```
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> java packages.StringCommando "Hallo mein Name ist" Anita
Hallo mein Name istAnita
```

```
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> java packages.StringCommando "Hallo mein", name
Hallo meinname
```

```
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> java packages.StringCommando "Hallo mein" :  name
Hallo mein:
```

SMART INDUSTRY CAMPUS

Exkurs: JAR-Dateien

- Java Archive (JAR) ist ein komprimiertes Archiv, das mehrere Java-Klassen, Ressourcen und Meta-Daten enthält
- im Wesentlichen: ZIP-Archiv mit einer bestimmten Struktur, das Java-Programme in einer einzigen Datei bündelt

SMART INDUSTRY CAMPUS

Exkurs: JAR-Dateien

- erleichtert die Verteilung von Java-Anwendungen
  - bspw. Bibliotheken zur Verbindung mit Datenbanken (mysql-connector.jar)
- reduziert Dateigröße (wie ZIP-Dateien)

SMART INDUSTRY CAMPUS

Exkurs: JAR-Dateien

- fertige Java-Programme können als .jar – Datei bereitgestellt werden
- viele Java-Bibliotheken sind als .jar-Datei veröffentlicht und können über Maven oder Gradle automatisch eingebunden werden
  - Manuell ebenfalls möglich

SMART INDUSTRY CAMPUS

Exkurs: JAR-Dateien

- Erstellen und Ausführen einer .jar-Datei:
  - c -> create
  - f -> file
  - e -> entry point (Ort der main-Methode)

Built-Systeme nehmen das für einen ab!

```
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> jar cfe myJar.jar packages.Test packages/Test.class
PS C:\Users\Tomme\IdeaProjects\ErstesProjekt\src> java -jar myJar.jar
2147483647
```

SMART INDUSTRY CAMPUS

Selbstbestimmtes Lernen

Lernplan erstellen

- erstellt euch einen Lernplan für heute und morgen
  - Soll eine Wiederholungseinheit beinhalten
  - Soll eine Übungseinheit beinhalten