



From good to great!

Gutes kann noch besser werden,
Sortiments- und Platzierungsoptimierung in einer neuen Dimension.

<https://github.com/HInformatikAG>



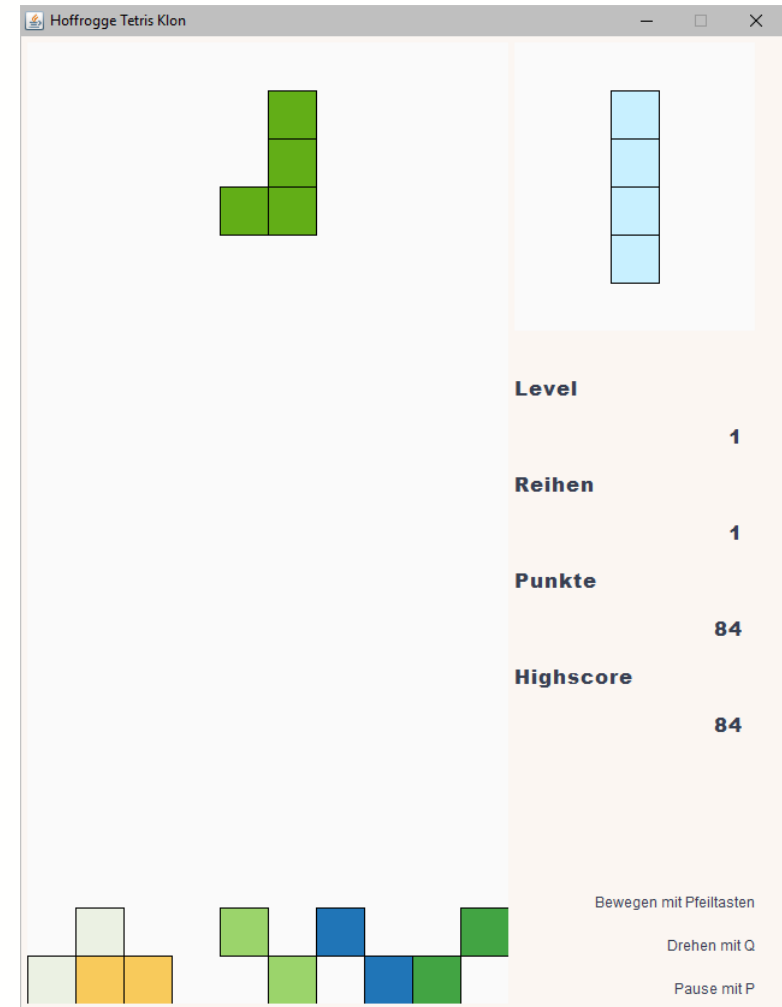
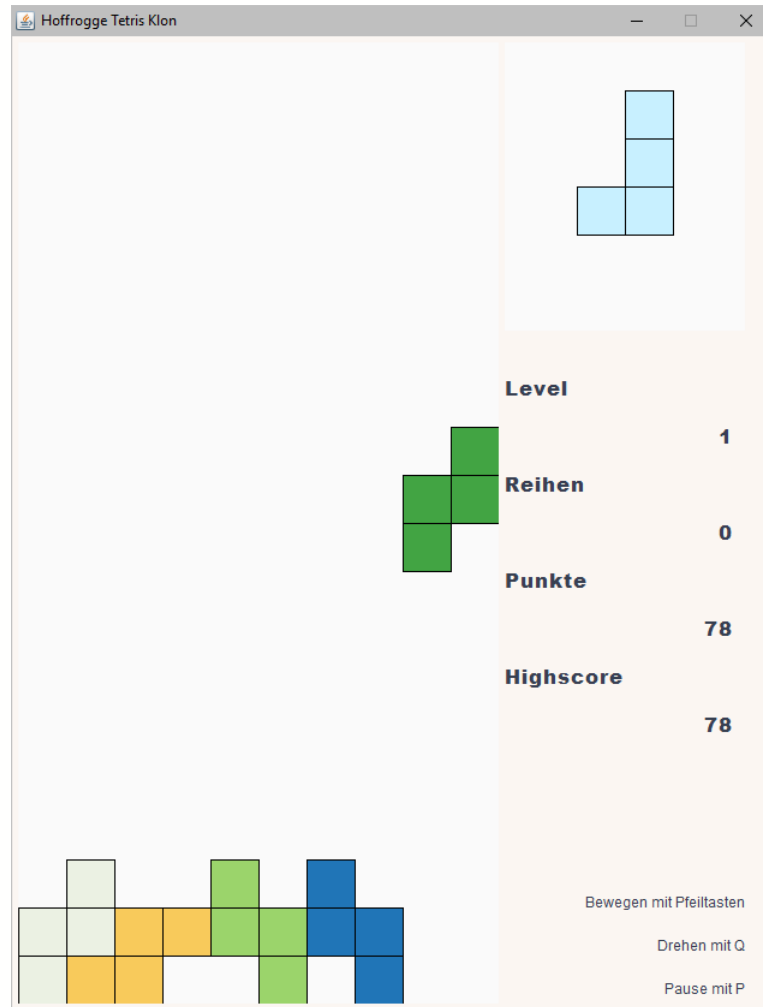
it-ag@hoffrogge.com

Lehreinheit 2

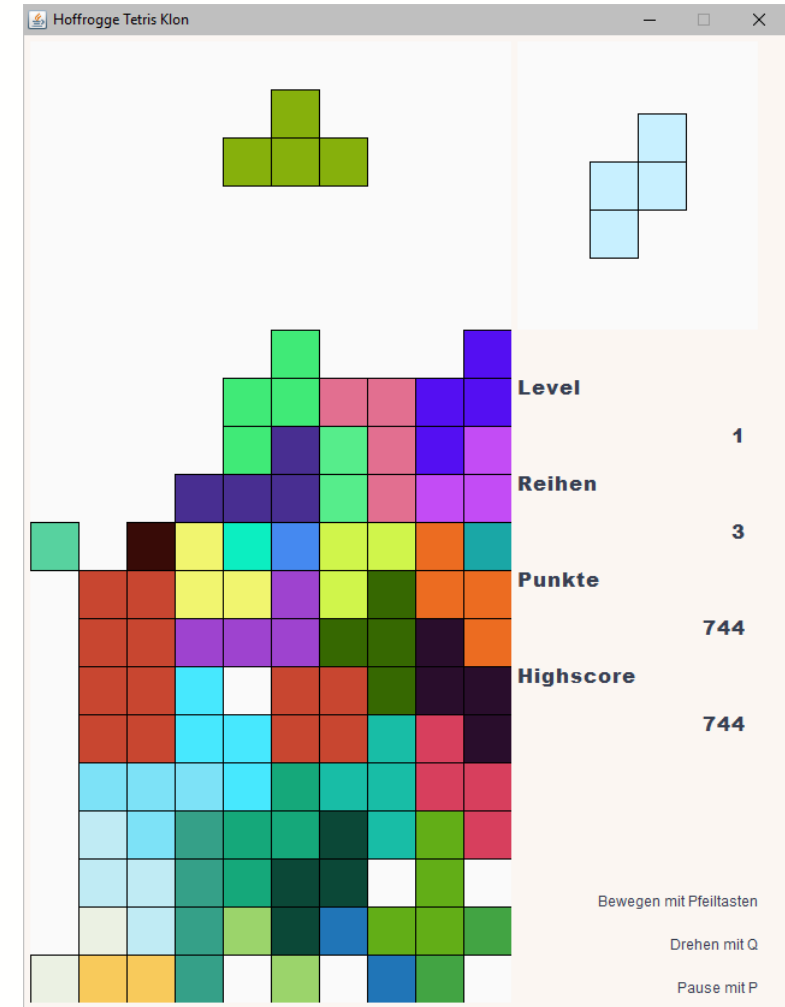
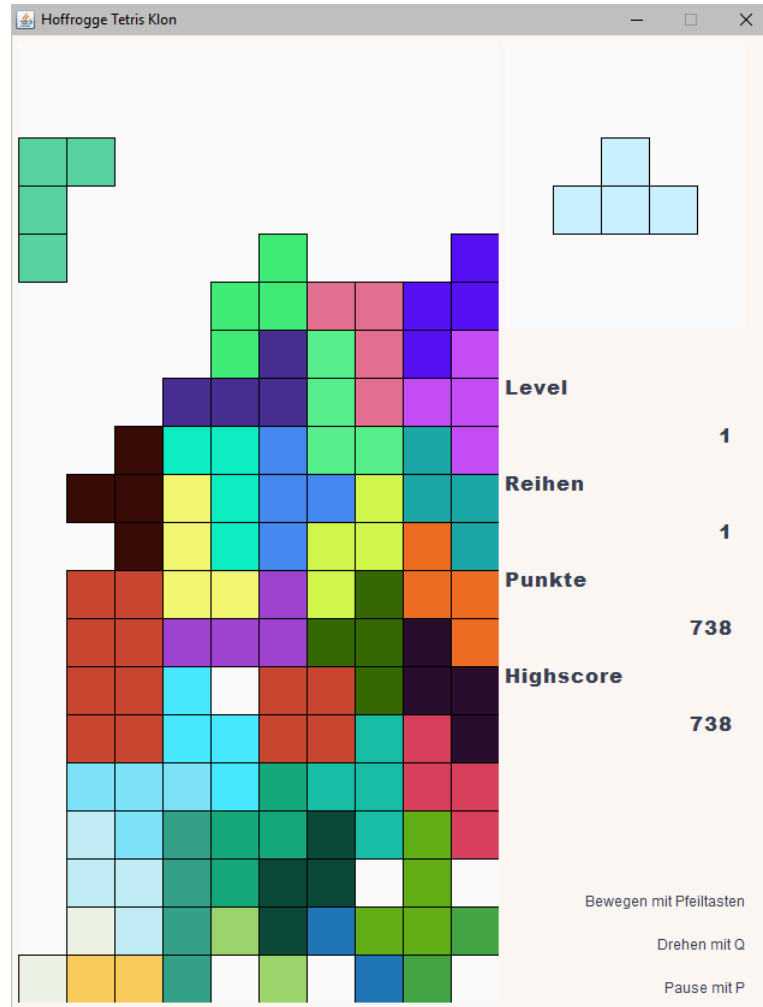
Ziele

1. `public static void main(String[] args)`
2. Hello World
3. Grundlagen
4. Klasse
5. Instanz/Objekt
6. Package
7. Import
8. Variablen
9. Konstruktor
10. Methoden
11. Übung

Was steckt in Tetris alles drin?



Es gibt viel mehr Spielfeldsituationen als die Tetrominos selbst



Aller Anfang ist gar nicht so schwer

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

- Jedes Java Programm benötigt zum Ausführen eine „Main-Methode“
 - `public static void main(String[] args)`
 - Einstiegspunkt/Startpunkt eines Java-Programms
 - Diese Methode wird von der Java Virtual Machine aufgerufen

Grundlagen

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

```
public class HelloWorld {  
  
    /* Startpunkt des Programms */  
    public static void main(String[] args) {  
        System.out.println("Hallo, Welt :");  
    }  
}
```


Grundlagen

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

Eine Java Klasse

```
public class HelloWorld {  
  
    /* Startpunkt des Programms */  
    public static void main(String[] args) {  
        System.out.println("Hallo, Welt :");  
    }  
}
```

Grundlagen

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



Start des Javaprogramms

```
public class HelloWorld {
```

```
    /* Startpunkt des Programms */
```

```
    public static void main(String[] args) {  
        System.out.println("Hallo, Welt :");
```

```
    }  
}
```

Grundlagen

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

```
public class HelloWorld {
```

```
    /* Startpunkt des Programms */
```

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

```
        System.out.println("Hallo, Welt :)");
```

```
    }
```

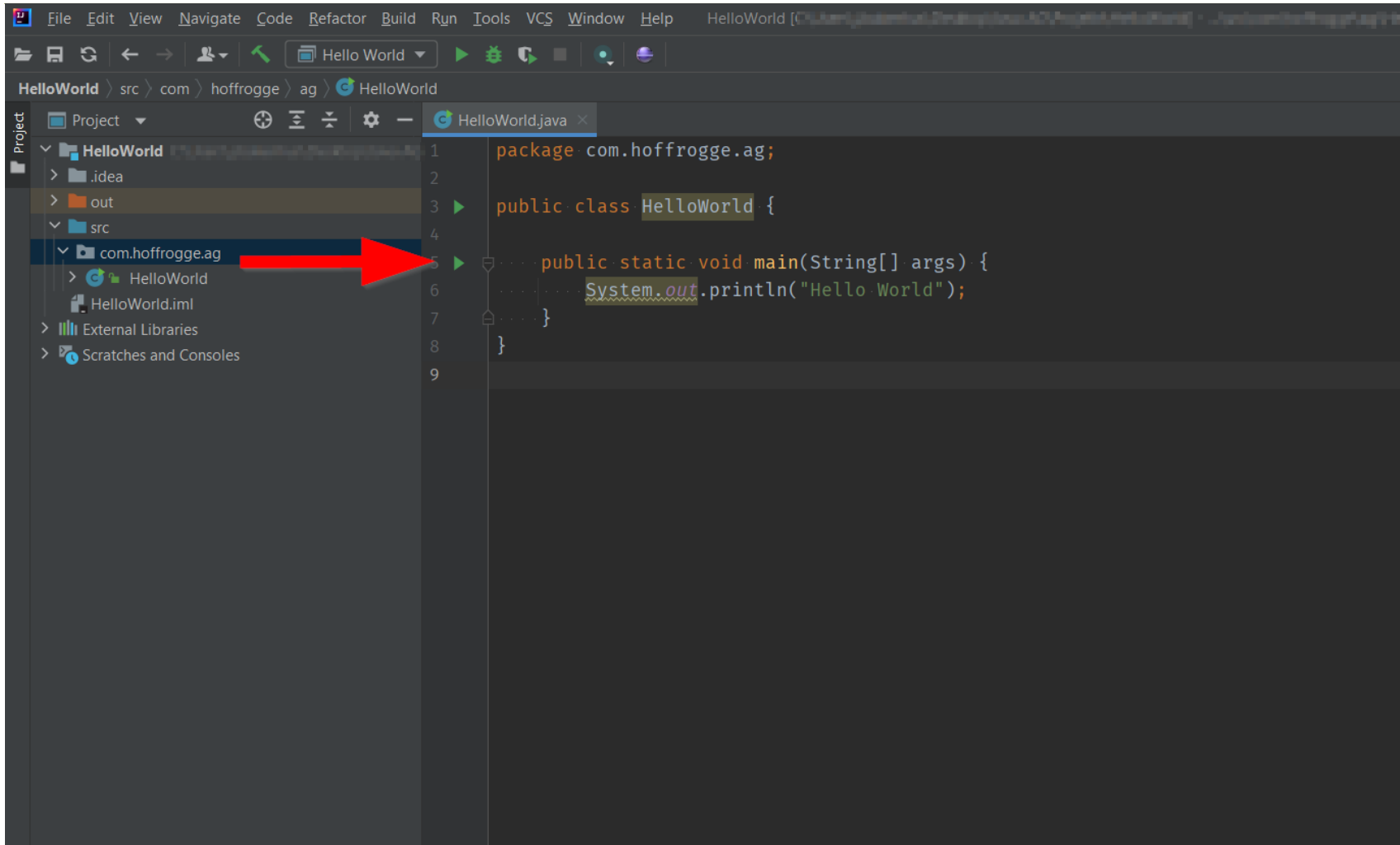
```
}
```



Parameter

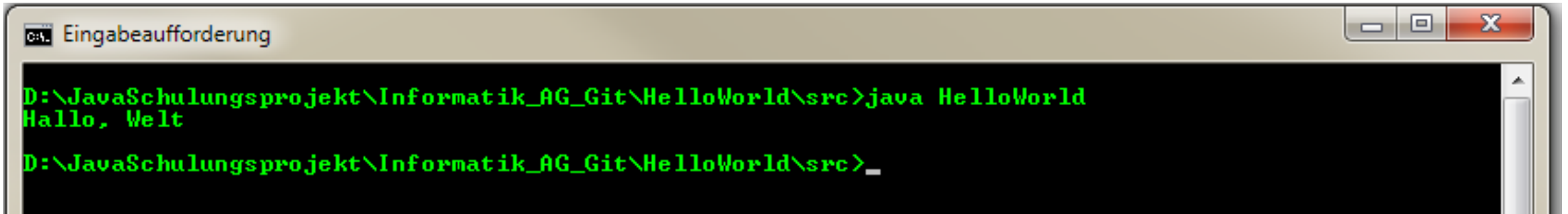
Javaprogramm ausführen

In IntelliJ IDEA



Javaprogramm ausführen

In der Konsole



```
C:\> D:\JavaSchulungsprojekt\Informatik_AG_Git\HelloWorld\src>java HelloWorld
Hallo, Welt
D:\JavaSchulungsprojekt\Informatik_AG_Git\HelloWorld\src>_
```

The screenshot shows a Windows command prompt window with the title 'Eingabeaufforderung'. The command prompt is open at the directory 'D:\JavaSchulungsprojekt\Informatik_AG_Git\HelloWorld\src'. The user has entered the command 'java HelloWorld', and the output is 'Hallo, Welt'. The prompt is currently at 'D:\JavaSchulungsprojekt\Informatik_AG_Git\HelloWorld\src>_'.

Javaprogramm ausführen

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

- Wir nutzen IntelliJ IDEA
- AUFGABE:
 - Suche die Datei HelloWorld.java im Projekt Lehreinheiten im Package com.hoffrogge.lehreinheiten.lehreinheit02 und starte das Programm

Basics erklärt

Instanz/Objekt

- Eine Klasse definiert Eigenschaften und Methoden
- Beispiel: die Klasse Wuerfel hat die Eigenschaften
 - anzahlSeiten
 - beschreibung
- Wir wollen viele Wuerfel haben
- Deswegen erzeugen wir Objekte/Instanzen der Klasse Wuerfel
- Diese Objekte können wir unterscheiden
 - Eine Manipulation eines Objekts lässt andere Objekte unberührt

Basics erklärt

Instanz/Objekt

- Eine Klasse definiert Eigenschaften und Methoden
- Beispiel: die Klasse Wuerfel hat die Eigenschaften
 - anzahlSeiten
 - beschreibung
- Wir wollen viele Wuerfel haben
- Deswegen erzeugen wir Objekte/Instanzen der Klasse Wuerfel
- Diese Objekte können wir unterscheiden
- Eine Manipulation eines Objekts lässt andere Objekte unberührt

Basics erklärt

Aufbau einer Klasse

```
package com.hoffrogge.lehreinheit02;
```

```
import java.util.Random;
```

```
public class Wuerfel {  
  
    private int anzahlSeiten;  
    private String beschreibung;  
  
    public Wuerfel() {  
  
    }  
  
    public int wuerfeln() {  
        // Zufallszahl  
    }  
}
```

Beispiel

Basics erklärt

Aufbau einer Klasse

```
package com.hoffrogge.lehreinheiten.lehreinheit02;
```



Package (Struktur)

```
import java.util.Random;
```

```
public class Wuerfel {
```

```
    private int  anzahlSeiten;
```

```
    private String beschreibung;
```

```
    public Wuerfel() {
```

```
    }
```

```
    public int wuerfeln() {
```

```
        // Zufallszahl
```

```
    }
```

```
}
```

Basics erklärt

Package

- Packages ermöglichen eine strukturierte Organisation der Klassen
- Und somit des ganzen Programms
- Man *kann* ohne Packages arbeiten
 - Nachteile
 - Unübersichtlich, dadurch hoher Organisationsaufwand
 - Fördert Kopplung
 - Fehlerbehaftet
 - Verhindert mehrere Klassen gleichen Namens

Basics erklärt

Package

- nur kleinbuchstaben
- Domainname rückwärts für Packages
 - **package** `com.hoffrogge.lehreinheiten.lehreinheit02;`

Basics erklärt

Aufbau einer Klasse

```
package com.hoffrogge.lehreinheiten.lehreinheit02;
```

```
import java.util.Random;
```



```
public class Wuerfel {
```

```
    private int    anzahlSeiten;
```

```
    private String beschreibung;
```

```
    public Wuerfel() {
```

```
    }
```

```
    public int wuerfeln() {
```

```
        // Zufallszahl
```

```
    }
```

```
}
```

Basics erklärt

Import

- Importiert eine fremde Klasse in die aktuelle Klasse
- Somit kann man die Werte und Methoden einer fremden Klasse benutzen
- Fördert natürlich auch Kopplung, aber vollständig ist Kopplung nicht zu vermeiden

Basics erklärt

Aufbau einer Klasse

```
package com.hoffrogge.lehreinheiten.lehreinheit02;
```

```
import java.util.Random;
```

```
public class Wuerfel {
```



Name der Klasse

```
    private int    anzahlSeiten;
```

```
    private String beschreibung;
```

```
    public Wuerfel() {
```

```
    }
```

```
    public int wuerfeln() {
```

```
        // Zufallszahl
```

```
    }
```

```
}
```

Basics erklärt

Klasse

- Eine Klasse kapselt Eigenschaften und Methoden
- Die Eigenschaften lassen sich mit Methoden manipulieren
- Klasse Wuerfel mit den Eigenschaften
 - anzahlSeiten
 - beschreibung

Basics erklärt

Aufbau einer Klasse

```
package com.hoffrogge.lehreinheiten.lehreinheit02;
```

```
import java.util.Random;
```

```
public class Wuerfel {
```

```
    private int   anzahlSeiten;  
    private String beschreibung;
```

```
    public Wuerfel() {
```

```
    }
```

```
    public int wuerfeln() {  
        // Zufallszahl  
    }
```

```
}
```



Eigenschaften/Variablen

Basics erklärt

Variablen

- Variablen (primitiv oder komplex) speichern einen Wert
- Variablen können meistens manipuliert werden, um den Wert zu ändern oder Berechnungen durchzuführen
- Variablen haben einen Gültigkeitsbereich, je nachdem, wo sie definiert sind
 - Das kann die ganze Klasse oder in einer Methode sein
 - Beispiele in `com.hoffrogge.lehreinheiten.lehreinheit02.VariablenScope`

Basics erklärt

Datentypen – primitive Datentypen

Typname	Größe ^[1]	Wrapper-Klasse	Wertebereich	Beschreibung
boolean	undefiniert ^[2]	java.lang.Boolean	true / false	Boolescher Wahrheitswert, Boolescher Typ ^[3]
char	16 bit	java.lang.Character	0 ... 65.535 (z. B. 'A')	Unicode-Zeichen (UTF-16)
byte	8 bit	java.lang.Byte	-128 ... 127	Zweierkomplement-Wert
short	16 bit	java.lang.Short	-32.768 ... 32.767	Zweierkomplement-Wert
int	32 bit	java.lang.Integer	-2.147.483.648 ... 2.147.483.647	Zweierkomplement-Wert
long	64 bit	java.lang.Long	-2^{63} bis $2^{63}-1$, ab Java 8 auch 0 bis $2^{64}-1$ ^[4]	Zweierkomplement-Wert
float	32 bit	java.lang.Float	$\pm 1,4\text{E}-45$... $\pm 3,4\text{E}+38$	32-bit IEEE 754, es wird empfohlen, diesen Wert nicht für Programme zu verwenden, die sehr genau rechnen müssen.
double	64 bit	java.lang.Double	$\pm 4,9\text{E}-324$... $\pm 1,7\text{E}+308$	64-bit IEEE 754, doppelte Genauigkeit

Quelle: https://de.wikibooks.org/wiki/Java_Standard:_Primitive_Datentypen

Basics erklärt

Aufbau einer Klasse

```
package com.hoffrogge.lehreinheiten.lehreinheit02;
```

```
import java.util.Random;
```

```
public class Wuerfel {
```

```
    private int    anzahlSeiten;
```

```
    private String beschreibung;
```

```
    public Wuerfel() {
```

```
    }
```

```
    public int wuerfeln() {
```

```
        // Zufallszahl
```

```
    }
```

```
}
```



Konstruktor

Basics erklärt

Konstruktor

- Ein Konstruktor erstellt eine Instanz einer Klasse
- Konstruktoren können Standardwerte nutzen
 - oder über Parameter die Eigenschaften des Objektes setzen
- `Wuerfel wuerfel = new Wuerfel();`
 - Erstellt einen Wuerfel mit Standardwerten für `anzahlSeiten` und `beschreibung`
- `Wuerfel wuerfel = new Wuerfel(6);`
 - Erstellt einen Wuerfel mit `anzahlSeiten` gleich 6
- `Wuerfel wuerfel = new Wuerfel(„Spielwürfel“);`
- Konstruktoren müssen definiert werden

Basics erklärt

Aufbau einer Klasse

```
package com.hoffrogge.lehreinheiten.lehreinheit02;
```

```
import java.util.Random;
```

```
public class Wuerfel {
```

```
    private int    anzahlSeiten;
```

```
    private String beschreibung;
```

```
    public Wuerfel() {
```

```
    }
```

```
    public int wuerfeln() {
```

```
        // Zufallszahl
```

```
    }
```

```
}
```



Eine Methode

Basics erklärt

Methoden

- Eine Methode manipuliert in der Regel die Eigenschaften einer Instanz einer Klasse (Setter)
- oder macht diese Eigenschaften zugänglich (Getter)
- oder führt eine Aufgabe durch, die meistens von den Eigenschaften der Instanz abhängt

- Eigenschaften manipulieren über Methoden
 - `wuerfel.setAnzahlSeiten(6);`
 - Setzt die Anzahl der Seiten des Würfels auf 6
 - `wuerfel.wuerfeln();`
 - würfelt eine zufällige Zahl
 - `wuerfel.getAnzahlSeiten();`
 - gibt an, wie viele Seiten der Würfel hat

Wiederholung & Übung

Komplexe Datentypen

- Wuerfel.java

- Kann viele primitive und komplexe Daten enthalten
- Zuweisung in einer beliebigen Klasse:
 - `Wuerfel wuerfel = new Wuerfel();`
 - `Wuerfel wuerfel = new Wuerfel(6);`
 - `Wuerfel wuerfel = new Wuerfel(6, "Wuerfel mit 6 Seiten");`

Wiederholung & Übung

Datentypen - Deklaration (Zuweisung) von Variablen

- `int anzahlSeiten; // implizite Zuweisung, es wird der Standardwert genutzt`
- `int anzahlSeiten = 6; // explizite Zuweisung, es wird der Wert nach dem = genutzt`
- `anzahlSeiten = 10; // existiert die Variable schon, muss der Typ nicht noch einmal geschrieben werden`
- `anzahlSeiten = 8 / 2; // in einer Variable kann auch das Ergebnis einer Berechnung stehen`
- `int multiplikator = 2;`
- `anzahlSeiten = 3 * multiplikator; // in der Variable steht jetzt 6`

Wiederholung & Übung

Instanz/Objekt

- Eine Klasse definiert Eigenschaften und Methoden
- Beispiel: die Klasse Wuerfel hat die Eigenschaften
 - anzahlSeiten
 - beschreibung
- Wir wollen viele Wuerfel haben
- Deswegen erzeugen wir Objekte/Instanzen der Klasse Wuerfel
- Diese Objekte können wir unterscheiden
 - Eine Manipulation eines Objekts lässt andere Objekte unberührt

Wiederholung & Übung

Übung

- Öffne die Klasse WuerfelUebung.java im Projekt Lehreinheiten im Package com.hoffrogge.lehreinheiten.lehreinheit02
- Bearbeite die Aufgaben in der Klasse

Vielen Dank!

