Inhalt

[**Klassen** 2](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.30j0zll)

[**Objekte** 2](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.1fob9te)

[**Methode** 2](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.3znysh7)

[Main- Methode 3](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.2et92p0)

[Was kommt in die Main Methode 3](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.tyjcwt)

[**Parameter** 3](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.3dy6vkm)

[**Datentypen** 3](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.1t3h5sf)

[**Konstructor** 4](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.4d34og8)

[**User Input** 4](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.2s8eyo1)

[**If Abfragen** 5](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.17dp8vu)

[**Verschachtelte If Bedingungen** 6](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.3rdcrjn)

[**Schleifen** 8](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.26in1rg)

[For Schleifen 8](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.lnxbz9)

[While Schleifen 8](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.35nkun2)

[Do While Schleifen 8](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.1ksv4uv)

[**Setter& Getter (Setzer& Holer)** 8](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.44sinio)

[Setter 8](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.2jxsxqh)

[Getter 9](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.z337ya)

[**Arrays** 10](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.3j2qqm3)

[Mehrdimensionale Arrays 10](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.1y810tw)

[Arraylist 10](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.4i7ojhp)

[**Static** 10](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.2xcytpi)

[**Private, Public, Default, Protected** 10](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.1ci93xb)

[**This** 11](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.3whwml4)

[**Vererbung** 11](https://docs.google.com/document/d/1mERX70idNPQIK9VMXrED03qMCniiGj9g/edit#heading=h.2bn6wsx)

Zusammenfassung

**Klassen**

* Beschreibt, welche Attribute/ Eigenschaften z.B ein Mensch haben kann, ohne diesem einen Wert zuzuweisen 🡪 z.B Augenfarbe, Größe,Geschlecht
* Beschreibt, welche Methoden/Funktionen ein Mensch haben kann 🡪 z.B Sprechen
* **Klassen werden immer großgeschrieben**

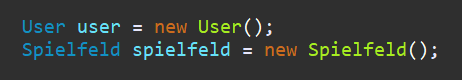
Klassen beim Schiffe versenken: GUI, Steuerung, Spieler, Spielfeld, Schiff



**Objekte**

* Objekt wird von Klasse abgeleitet

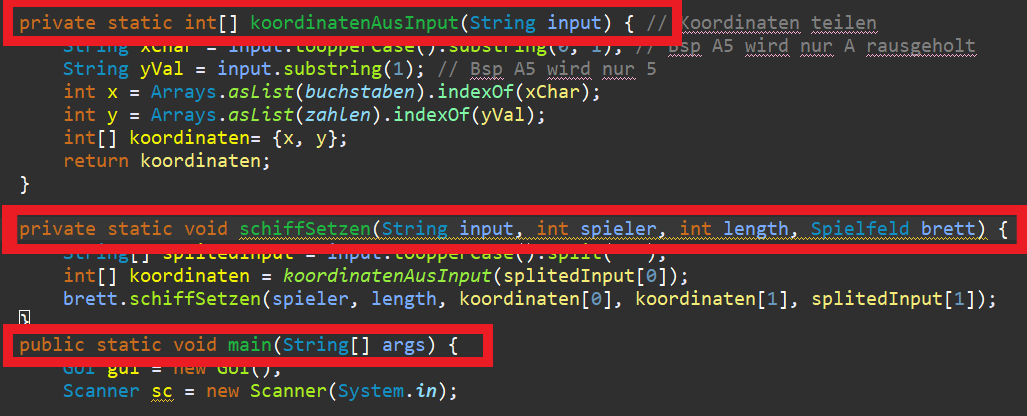
🡪jedes Objekt gehört zu einer Klasse, besitzt Attribute, Methoden dieser Klasse

Ein Objekt erzeugen wir mit Hilfe des new-Operators.

**Methode**

=Funktionen

Eine Methode ist die Anleitung, wie und was gemacht wird

****

Main- Methode

Eine Methode main() muss jede Java-Anwendung besitzen. Sie stellt den Einstiegspunkt in die Ausführung einer Java-Anwendung dar und muss die Signatur public static void main(String[] args) besitzen.



* public weil von außerhalb der Klasse auf sie zugegriffen wird.
* static weil sie aufgerufen wird ohne dass vorher ein Objekt einer Klasse gebildet wurde.
* void weil sie keinen Rückgabewert besitzt.
* der Parameter String[] args, der auch String args[] geschrieben werden darf, ist ein Array, das die vom Aufruf entgegen genommenen Kommandozeilenparameter enthält.

Was kommt in die Main Methode

**Parameter**

(Parameter in der Klammer der Methode)

Methoden können Parameter haben, mit denen zusätzliche Informationen angegeben werden können.

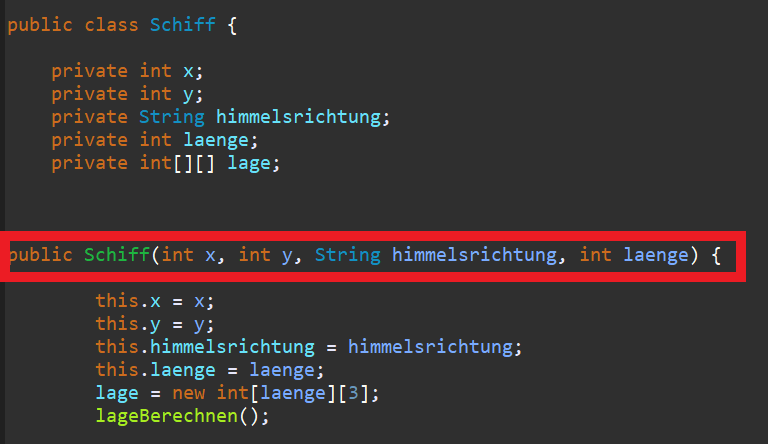
**Datentypen**

|  |  |  |
| --- | --- | --- |
| Typ | Beschreibung | Beispiel |
| **boolean** |  | True, false |
| **char** | Einzelnes Zeichen | z.B Geschlecht (m, w, d) |
| **int** | Ganze Zahl  (darf keine 0 am Anfang haben | Alter, Studienjahr |
| **float** | Fließkommazahl  (auf 7 Nachkommastellen begrenzt) | 3,14159f |
| **double** | Kommazahl  (mehr als 7 Nachkommastellen) | 3,141594789274 |
| **string** | Zeichenkette | “Dies ist ein String” |
| **short** | Kurze ganze Zahlen |  |
| **long** | Riesige ganze Zahlen |  |
| **byte** | Sehr kleine Zahlen | -128 bis 127 |

**Konstruktor**

Konstruktoren sind spezielle methodenähnliche Klassenstrukturen, die den Namen ihrer Klasse tragen und beim Erzeugen von Objekten der Klasse über das Schlüsselwort new aufgerufen werden

Ein Konstruktor ist eine Methode, die automatisch beim Erstellen eines neuen Objektes einer Klasse aufgerufen wird und dazu dient, das Objekt zu initialisieren



Der Konstruktor in Java ist somit für das Initialisieren von Werten geeignet und für das setzen von Werten die für das erstellte Objekt relevant sind.

Dazu muss man die Werte die man einem Objekt übergeben möchte dem **Konstruktor** als Parameter übergeben und sie in ihm setzen.

**User Input**

**Nutzer** (User) soll etwas eingeben können, benötigen einen Scanner

* Scanner muss importiert werden

Imports müssen immer über der Klasse eingefügt werden!

Damit wir diese Methoden nun verwenden können, müssen wir eine Variable vom Typen Scanner erzeugen.

Um etwas eingeben zu können, verwenden wir System.in.

Nun erzeugen wir eine Klasse, importieren Scanner und sorgen dafür, dass die Main Methode in der Klasse eingfügt ist, außerdem fügen wir in diese main Methode folgenden Code ein:



1. ein (Referenz-)Variable (Scanner namederVariable) vom Typen Scanner anlegen

🡪scannerVariable

1. neues Objekt erzeugen (new Scanner(...) mit dem Inhalt den der Nutzer eingibt (user input).

Die Main Methode sieht dann so aus, wenn wir zusätzlich eine Zeile vorne anfügen, die dem Nutzer ausgibt, was zu tun ist:

**If Abfragen**

|  |  |
| --- | --- |
| **Mit Alternative** | **Ohne Alternative** |
| **If** (<Anweisung>) {  }  **else** {<Anweisung>  }  Beispiel  **If** (erg > 100) {  System.out.println(„big“);  } **else** {  System.out.println(„small“);  } | **If** (<Anweisung>) {  }    Beispiel  **If** (alter >= 18) {  Zugang = true;  } |

**Verschachtelte If Bedingungen**

Beispiel Schiffe versenken

public class Panjutorials {

public static void main(String[] args) {

int iq = 135;

if(iq<100){

System.out.println("Schlechter als der Durchschnitt.");

}else{

System.out.println("Besser als der Durchschnitt");

if(iq>130){

System.out.println("Gehört zu den intelligentesten 2% der Bevölkerung.");

}else{

System.out.println("Aber nicht intelligent genug für Mensa.");

}

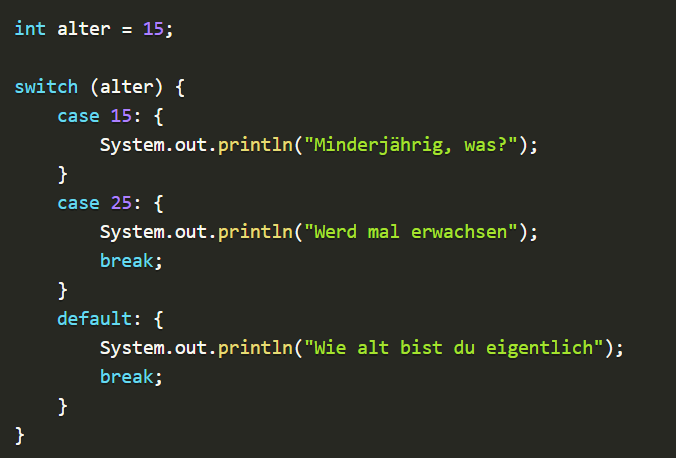
}

}

}

Hierbei wird zuerst abgefragt, ob die Variable iq kleiner ist als 100, wenn dies der Fall ist, soll "Schlechter als der Durchschnitt" ausgegeben werden. Wenn nicht, soll ausgegeben "Besser als der Durchschnitt" und außerdem soll abgefragt werden, ob die Variable größer ist als 130, wenn das der Fall ist, soll ausgegeben werden: Gehört zu den intelligentesten 2% der Bevölkerung. Wenn nicht, dann: "Aber nicht intelligent genug für Mensa."

**Switch& Case**



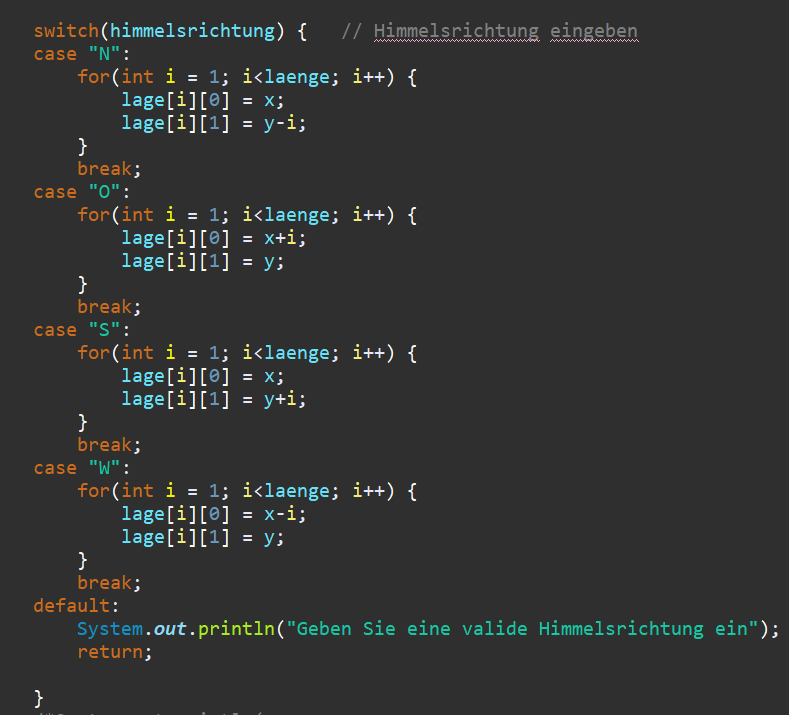
In runden Klammern nach **switch** Variable eingeben, die man untersuchen möchte. (Variable muss sich in Integer umwandeln lassen)

Dann folgen geschweifte Klammern und die "cases", dann die Zahl/String die man überprüfen möchte gefolgt von einem Doppelpunkt.

Folgend, in geschweiften Klammern, der Code den man ausführen möchte.

* Break nicht vergessen, sonst werden alle „cases“ die nach erfülltem Fall kommen aufgerufen

Beispiel Schiffe versenken



**Logische Operatoren**

|  |  |
| --- | --- |
| > | größer als |
| < | kleiner als |
| >= | größer gleich |
| <= | kleiner gleich |
| == | gleich |
| != | ungleich (! nicht) |
| && | und |
| || | oder |

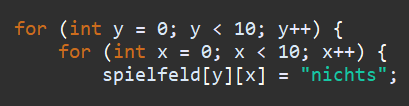
**Schleifen**

Schleifen werden verwendet, um eine Handlung mehrfach durchführen zu können

For Schleifen

Die For-Schleife nimmt man immer dann, wenn man die Anzahl der benötigten Schleifen-Durchläufe schon im Voraus kennt.

Es wird erst geprüft (wenn Bedingung wahr) dann wird sie ausgeführt

//for

for (int i = 0; i <= 1000; i = i+2) {

System.out.println (“countToThousant:” +i);

}

While Schleifen

Die While-Schleife wird so lange durchlaufen, bis die Bedingung ein **False** ergibt. Dabei steht die Bedingung am Anfang, ist sie also schon davor „falsch“, dann wird die Schleife kein einziges Mal ausgeführt, sondern übersprungen.

Do While Schleifen

Wenn man weiß, dass die Schleife auf jeden Fall **mindestens** einmal ausgeführt werden soll, dann kann man statt einer While-Schleife auch eine **Do-While-Schleife** einsetzen. Eine Do-While-Schleife ist im Grunde genommen nichts anderes, als eine While-Schleife bei der die Schleifen-Bedingung am Ende der Schleife und nicht am Anfang steht. Do wird geschaut ob man reinkommt dann geht es in die while Schleife hinein. Also mind 1 mal

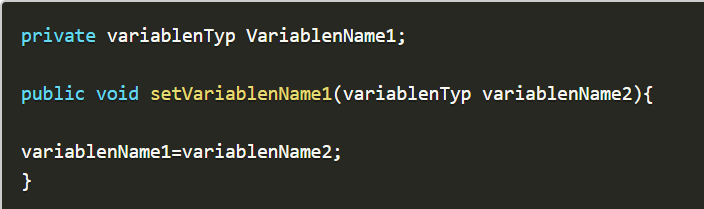
**Setter& Getter (Setzer& Holer)**

Zugriff auf private Attribute von außen NUR über Zugriffsmethoden möglich!

🡪Zugriffsmethode (gewährt "Zugang" zu privaten Attr./Meth.)

Setter

Methode, die die Veränderung einer Objektvariablen ermöglicht.

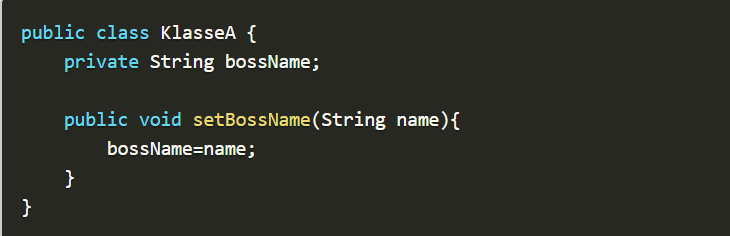


Dadurch, dass wir nun private vor den VariablenTypen einfügen, ist diese Variable nur in der selben Klasse sichtbar in der sie angelegt wurde. Außerdem sagen wir, dass die Setter Methode public sein soll, und void als Rückgabewert, also keinen, hat. Die Methode nennen wir set + den Variablennamen.

Innerhalb der runden Klammern erzeugen wir eine Variable vom gleichen Variablentypen wie die zu setzende Variable.

In den geschweiften Klammern sagen wir nun, dass die Variable die wir privat angelegt haben gleich der Variable in unseren runden Klammern sein soll.

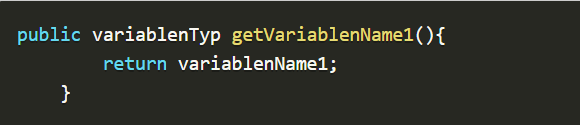
Beispiel:



Wie ihr sehen könnt, haben wir die Variable bossName außerhalb einer Methode, sie somit **global,** angelegt.

Jetzt können wir die Variable zwar von der KlasseB aus setzen, jedoch sie noch nicht in ihr anzeigen lassen. Um das zu können, müssen wir in unserer KlasseA noch einen **Getter** hinzufügen.

Getter



In der **Getter** - Methode ist es wichtig, dass der Variablentyp dem Typen des Wertes entspricht, den wir haben wollen. Innerhalb der {} haben wir lediglich return und den Namen der Variable die die Methode zurückliefern soll.

public String getBossName(){

   return bossName;

}

**Arrays**

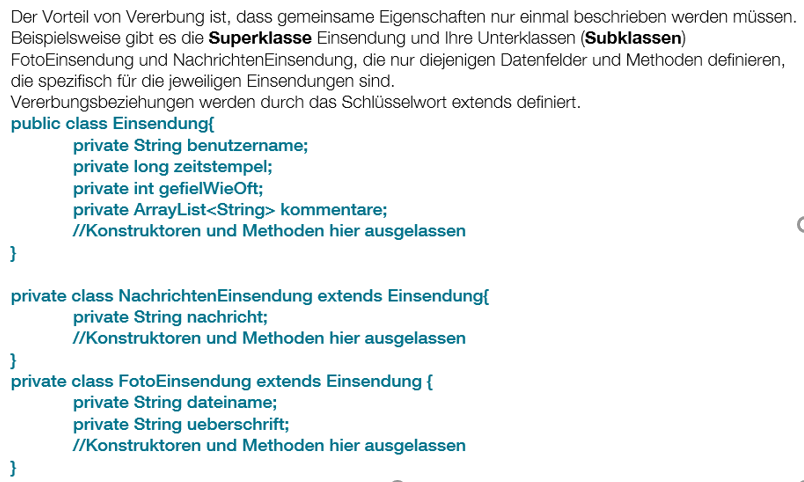
Es wird von 0 gezählt

Mehrdimensionale Arrays

Man kann verschiedenviele Dimensionen mit **Arrays** darstellen.

Arraylist

Public class Baum{



**Static**

Wird genutzt sodass schon ein gewisser Zustand herrscht bevor überhaupt ein Objekt erstellt wird.

Methoden und Objekte ohne static sind erst verfügbar wenn diese auch erzeugt werden.

Objekte nutzen diese Static Klasse dann gemeinsam. static wird eingesetzt wenn für alle Objekte etwas gelten soll, z.b. Gravitation.

**Private, Public, Default, Protected**

Eigt alles Private wenn es geht🡪sonst public verwenden

Kann nur innerhalb von dem gewissen Ding zugreifen beiPrivate

🡪immer wenn public aufpassen , dass so wenig wie möglich nach außen geht

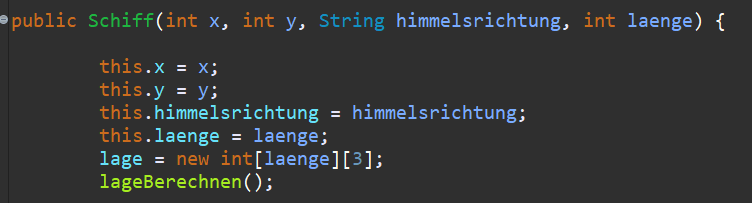
Klasse kann man besser umbauen wenn private

Baum tollebaum=new Baum(namen;“Tanne“);

**This**

this.a = abcd

--> this überweist den Parameter aus einer Methode hier abcd an das Datenfeld, sodass dieser Parameter auch von anderen Klassen genutzt werden kann.



Objekterzeugung

Student neuerStudent1=new Student();

**Vererbung**