Objektorientierung

zusammengestellt von J. Rau /01.2020 unter Verwendung von https://mein-javablog.de

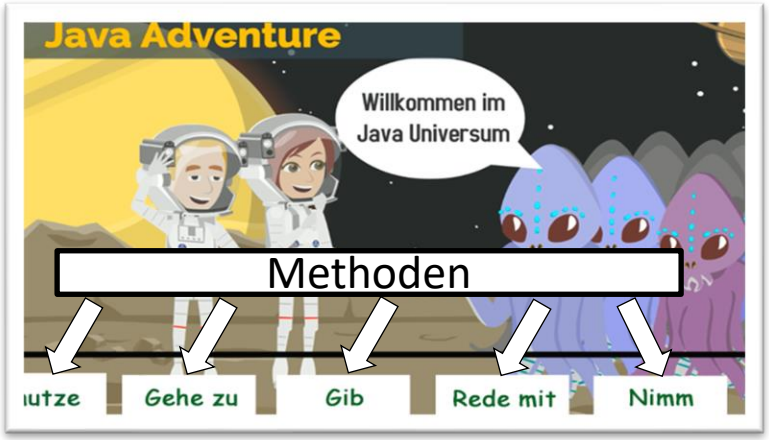
Ein Held wird in eine Welt programmiert. Er muss mit Personen sprechen, muss Gegenstände einsammeln und diese benutzen. Und der Held muss Rätsel lösen.
All diese Dinge: Welt, Held, Gegenstände, Personen sind **Objekte.**

Objekte stehen in einem Programm zu Beginn nicht zur Verfügung, sie **müssen erst erzeugt werden**.

Für ein Objekt benötigst du einen Bauplan: für den Helden, für die Gegenstände usw. Diese **Baupläne sind** deine **Klassen**.

Eine **Klasse** besteht aus **Methoden, Konstruktoren** und **Variablen.**

Objekte müssen etwas können, dafür werden **Methoden** genutzt.

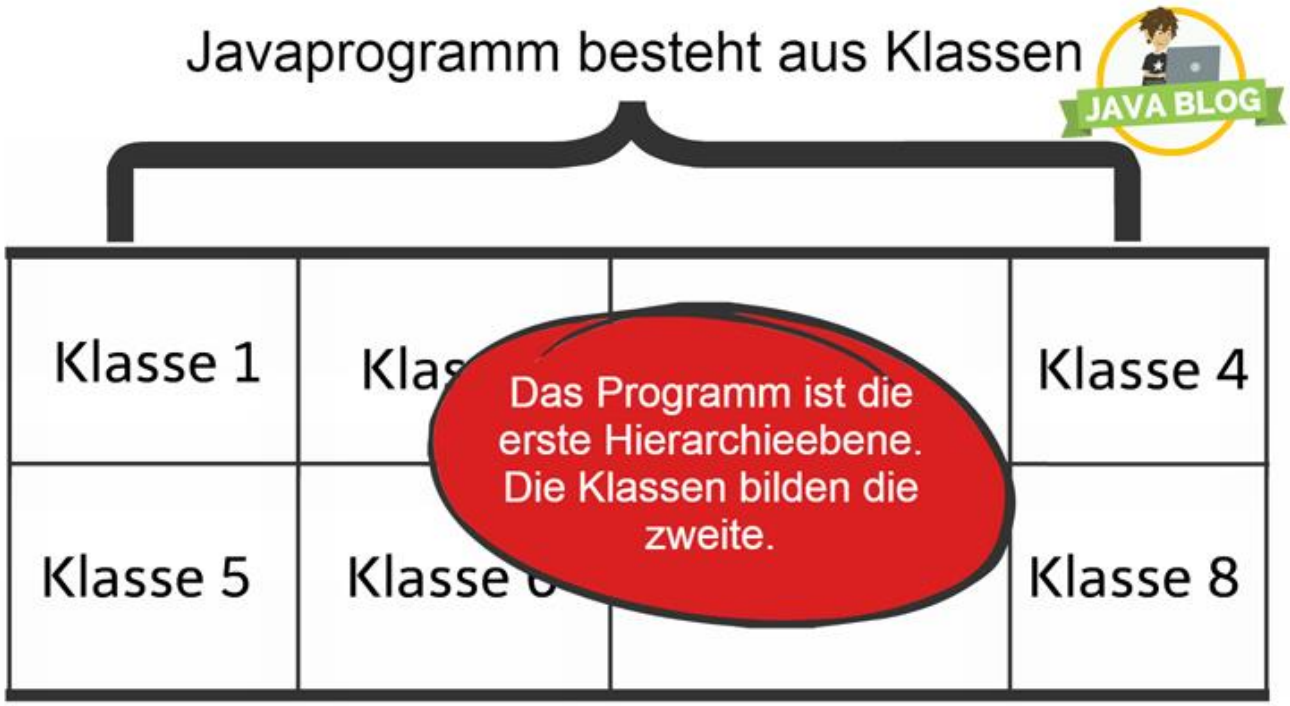


Neben den Objektmethoden gibt es **statische Methoden**. Das sind Klassenmethoden und wichtig für das gesamte Programm, z.B.

Starte das Spiel.
Erschaffe dieses Objekt.
Beende das Spiel.

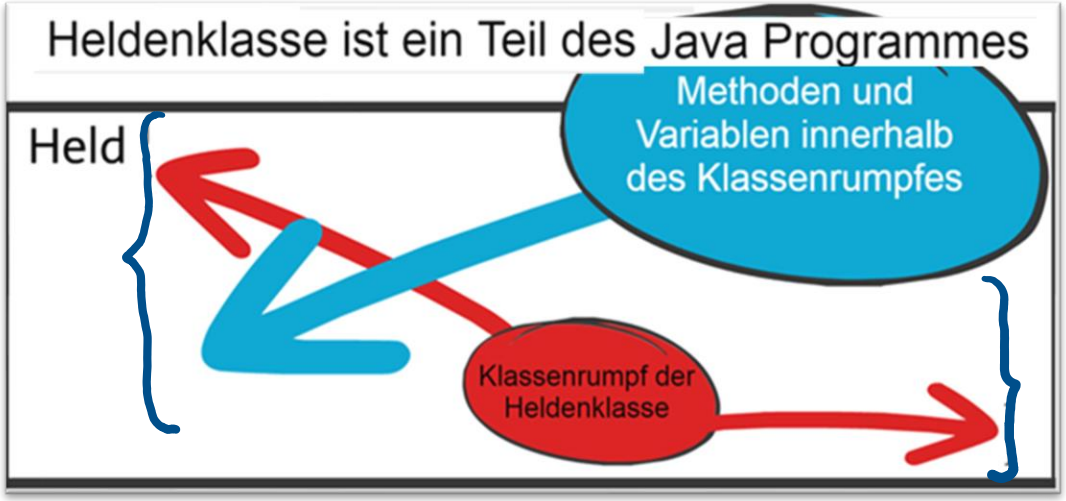
Variable sind Objektvariable. Das sind die Eigenschaften bzw. Attribute der Objekte.

Alle Java Klassen sind Bestandteile eines Java Programmes.
Jedes Java Programm besteht aus **Klassen**, sie sind die **Bausteine eines Programmes**.



Der innere Aufbau von Java Klassen

Eine Java Klasse besteht aus zwei Teilen: dem **Klassenkopf** und dem **Klassenrumpf**.



Zusammenfassung

Eine **Programm** besteht aus **Klassen,** die miteinander kommunizieren.

Eine **Klasse** besteht aus **globalen Variablen, Methoden** und **Konstruktoren.**

Klassen sind die Modelle von Objekten, die sie erzeugen.

Globale Variable: **Attribut (Eienschaft)** eines Objekts

Methoden: **Fähigkeiten** eines Objekts, **Statusrückgabe** des O.
Beziehungen zwischen Objekten herstellen

Konstruktor: **Erzeugen** eines Objekts im Startzustand.

Verwendung von Variablen

1. Instanzvariable bzw. globale Variable

Diese sind **im gesamten Programm gültig** und werden so deklariert:

I. Datenspeicher wird reserviert: **Datentyp varName**

IM KLASSENKOPF

II. Der Variablenwert wird zugewiesen: **varName =** variablenWert

IN EINER METHODE/KONSTRUKTOR

Wird kein Wert zugewiesen: varWert ist automatisch 0 (Zahlen) oder null (String).

Verwendung von Variablen

2. lokale Variable

Diese werden nur **temporär** benötigt und sind **nur in Laufzeit der Methode/bzw. im Konstruktor gültig** und werden so deklariert:

Datentyp varName = variablenWert

INNERHALB EINER METHODE/KONSTRUKTOR

Der lokalen Variablen muss sofort ein Wert zugewiesen werden, auch wenn er 0 oder null ist.

Einfache Datentypen --> Kleinschreibung!!

Java kennt vier Gruppen einfacher Datentypen:

I. **logischer Typ** boolean

II. **Ganzzahltyp** int, byte, short, long

III. **Gleitpunkttyp** double; float

IV. **Zeichentyp** char

Ganzzahl - integer → **int**
Deklaration: int alter; alter=15;
von -2.147.483.648 bis 2.147.483.647
für größere Werte → long

Gleitpunktzahl - double → **double**
Deklaration: double groesse; groesse=1.82;

Einzelsymbol - charakter → **char**
Deklaration: char a; a='€';

Wahrheitswert - boolean → **boolean**
Deklaration: boolean superHeld; superHeld=true; oder superHeld=false;

Objektypen mit eigenen Klassen --> Großschreibung!!

Text - String → **String**

Deklaration: String held; held= "Moritz";

Liste - Arraylist → **ArrayList**

Deklaration: ArrayList <Helden> a; a.add= "Moritz";

Eigenes Objekt - Schwert → **Schwert**

Deklaration: Schwert excali; excali.stichZu ( ) ;

Parameter

Um Werte für Variablen einzugeben und diese an Variablen zu übergeben, gibt es Parameter (temporäre **Übergabe-Werte**).

Parameter sind **nur innerhalb der Methode oder des Konstruktors gültig** und werden in zwei Schritten verwendet:

I. Parameter im Methodenkopf (--> Signatur) festlegen

II. Parameter-Wert an eine Variable übergeben.

Signatur einer Methode

… methodenName (**Datentyp** param1Name**, Datentyp** param2Name**,…)**

IM METHODEN/KONSTRUKTOR-KOPF

```
{
    variable1 = param1Name;
    variable2 = param2Name;
    ÜBERGABE AN DIE VARIABLE
}
```

```
public class Heldenklasse
{       //hier Attribute (globale Variablen)festlegen
        int staerke, gewinn;
        String name, geschlecht;
        public Heldenklasse()
        { //der Konstruktor erzeugt und initialisiert Objekte, kann auch leer sein
                name="Herbert"; geschlecht="männlich"; staerke=20;
        }
        public void kaempfe()
        { //Vorlage einer verändernden Methode, die etwas tut ...
          gewinn= staerke-2;
        }
        public int speichereStaerke()
        {   //Vorlage einer sondierenden Methode, die den Status einer
            Variablen zurückgibt → return ...
                staerke=staerke + gewinn;
                return staerke;
        }
}
```

Klassenname beginnt mit Großbuchstaben

Variablenname beginnt mit Kleinbuchstaben

M.-name beginnt mit Kleinbuchstaben

Jeder Datentyp ist möglich, aber die return-Variable muss vom selben Typ sein.

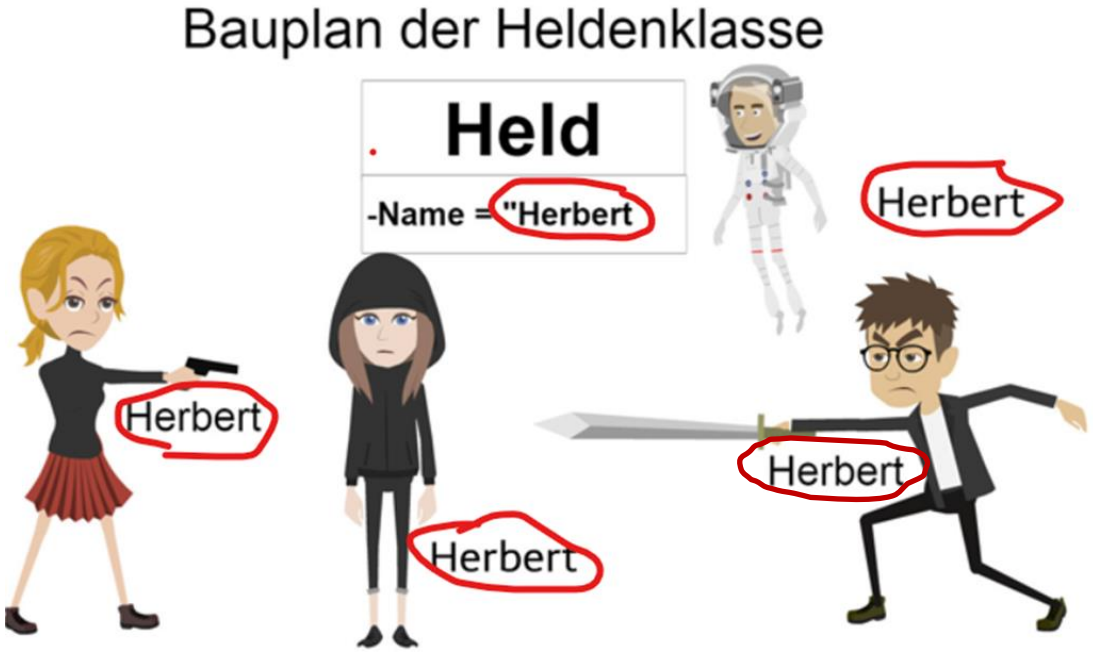
Der Konstruktor

Jetzt musst also ein Objekt erschaffen, mit den festgelegten Eigenschaften und Methoden.

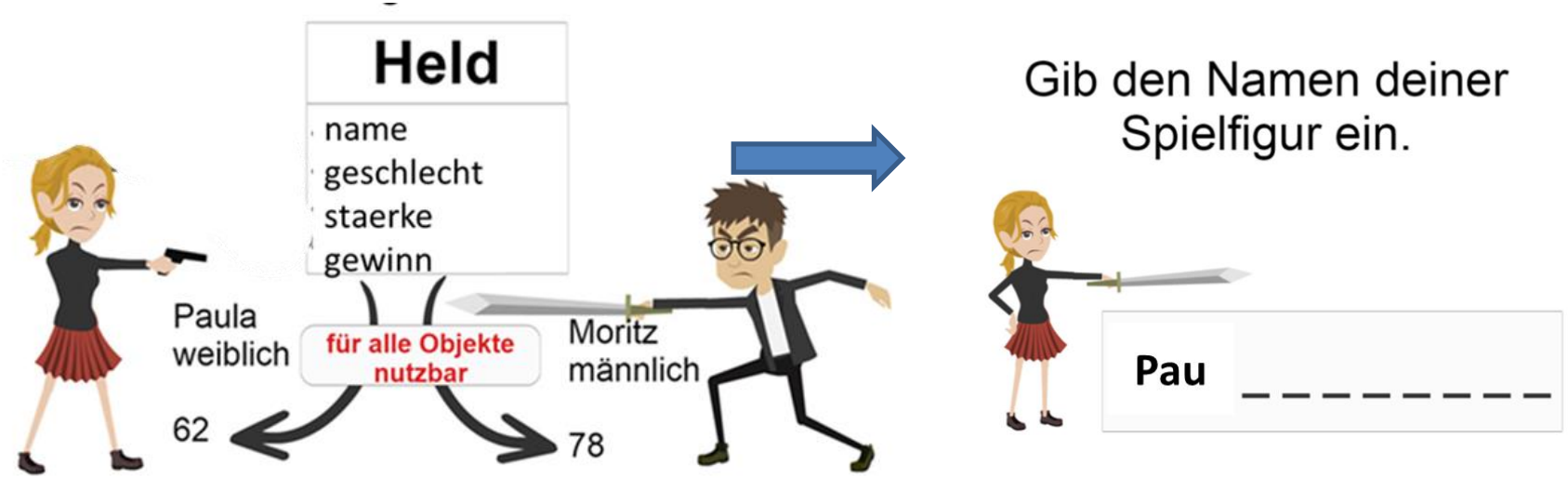
Konstruktoren sind die Erzeuger eines Objekts mit den Starteigenschaften.

Alle Eigenschaften und alles was dein Held von Anfang an kann, wird im Konstruktor festgelegt.

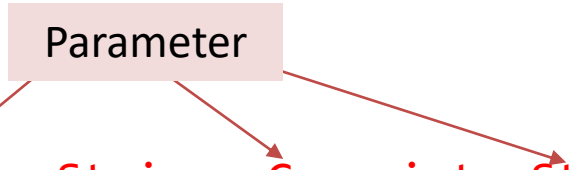
In dem Klassenbeispiel **heißen alle Helden Herbert**, weil der Konstruktor nur Herberte erzeugt…



Ein Konstruktor soll aber für alle Helden nutzbar sein...



Ein Konstruktor mit Parametern…




```
public Heldenklasse(String eName, String eGes, int eStaerke)
{   //der Konstruktor erzeugt und initialisiert Objekte,
        name=eName; geschlecht=eGes; staerke=eStaerke;
}
```

lies: Die Variable name ergibt sich aus dem Eingabenamen.