

Minimales Java-Programm

```
public class SagHallo {
    public static void main(String[] args) {
        System.out.println("Hallo");
    }
}
```

int	Ganzzahl (+/-2Mrd)	12
long	Grössere Ganzzahlen	123456...
double	Gleitkommazahl	2.67
boolean	true und false	true
String	Beliebiger Text	"Morgen"
char	Einzelner Buchstabe	'a'

Variablen

Deklarieren	[Datentyp] <i>name</i> ;	int a;
Wert zuweisen	<i>name</i> = [Wert];	a = 12;
Deklarieren und zuweisen	[Datentyp] <i>name</i> = [Wert];	int a = 3;

+	Addition	/	Division
-	Subtraktion	%	Modulo
*	Multiplikation		

<i>name</i> ++	Wert plus 1
<i>name</i> --	Wert minus 1

&&	AND
	OR
!	NOT

<	Kleiner	>=	Grösser gleich
<=	Kleiner gleich	==	Gleich
>	Grösser	!=	Ungleich

<<	Linksschieben	&	Bitweise AND
>>	Rechtsschieben		Bitweise OR
		^	Bitweise XOR

Bedingungen

```
if (Bedingung) {
    ...
} else if (Bedingung) {
    ...
} else {
    ...
}
```

Schleifen

```
while (Bedingung) {
    ...
}
```

```
for (Startwert; Bedingung; Update) {
    ...
}
```

```
for (Listen-Element : Liste) {
    ...
}
```

Konsole

```
int zahl = 7;
System.out.println("Text");
System.out.println(zahl);
System.out.println("B: "+zahl+"m");
```

```
import java.util.Scanner;

Scanner ks = new Scanner(System.in);
int a = ks.nextInt();
double b = ks.nextDouble();
String s = ks.nextLine();
ks.close();
```

Kommentar

```
// Zeilenkommentar

/* Blockkommentar über
   mehrere Zeilen */
```

Dezimal-, Hexadezimal- und Binärsystem

0	0x00	0000	8	0x08	1000
1	0x01	0001	9	0x09	1001
2	0x02	0010	10	0x0A	1010
3	0x03	0011	11	0x0B	1011
4	0x04	0100	12	0x0C	1100
5	0x05	0101	13	0x0D	1101
6	0x06	0110	14	0x0E	1110
7	0x07	0111	15	0x0F	1111

Arrays (1D)

0	1	2	3	4
			5	

```
int [] alter = new int[5];
alter[3] = 5;
int zweiter = alter[1];
System.out.println(alter[2]);
```

```
int [] alter = {18, 32, 16, 5, 24};
for (int i=0; i<alter.length; i++) {
    System.out.println(alter[i]);
}
```

Arrays (2D)

	0	1	2	3	4
0					
1				9	
2	7				

```
int [][] a = new int [3][5];
a[1][3] = 9;
a[2][0] = 7;
```

```
for (int i=0; i<a.length; i++) {
    for (int j=0; j<a[i].length; j++) {
        System.out.print(a[i][j]);
    }
}
```

Statische Methoden und Attribute

```
public class Test {
    static int a = 7;

    public static void main(...) {
        int r = addA(3);
        System.out.println(r); //->10
    }

    static int addA(int b) {
        return a+b;
    }
}
```

Exception

```
try {
    // Hier könnte etwas passieren
} catch (Exception e) {
    // Wird im Fehlerfall ausgeführt
}
// Hier geht es in jedem Fall weiter
```

Array mit Objekten

```
Artikel [] all = new Artikel[2];
all[0] = new Artikel("Brot");
all[1] = new Artikel("Öl");
all[1].setName("Salz");
String n1 = all[1].getName();
```

```
for (int i=0; i<all.length; i++) {
    System.out.println(all[i].getName());
}
```

ArrayList

```
import java.util.ArrayList;
```

```
ArrayList<Artikel> artList = new ArrayList<Artikel>();
```

```
Artikel x = new Artikel("Brot");
artList.add(x);
Artikel y = artList.get(0);
artList.remove(x);
artList.add(0, new Artikel("Öl"));
artList.set(0, new Artikel("Tee"));
artList.remove(0);
```

```
for (int i=0; i<artList.size(); i++) {
    System.out.println(artList.get(i).getName());
}
```

```
for (Artikel a: artList) {
    System.out.println(a.getName());
}
```

HashMap

```
import java.util.HashMap;
```

```
HashMap<String, Artikel> art = new HashMap<String, Artikel>();
```

```
Artikel a1 = new Artikel("Salz");
art.put("G01", a1);
art.put("G02", new Artikel("Öl"));
art.remove("G01");
boolean has = art.containsKey("G02");
Artikel g = art.get("G02");
```

```
for (Artikel k: art.values()) {
    // etwas mit k machen...
}
```

```
for (String s: art.keySet()) {
    // etwas mit s machen...
}
```

Klassen und Objekte

```
public class Shop {
    public static void main(String[] args) {
        Artikel a1 = new Artikel("Ragout");
        a1.preis = 1.75;
        a1.setName("Ragusa");
        double preis = a1.preis;
        String name = a1.getName();
        a1.print();
    }
}
```

```
public class Artikel {
    private String name;
    public double preis;

    public Artikel(String name) {
        this.name = name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void print() {
        System.out.println(preis+" CHF");
    }
}
```

Vererbung

```
public class Buch extends Artikel {
    private String isbn;

    public Buch(String name, String isbn, double preis) {
        super(name); // Parent-Konstruktor
        this.preis = preis;
        this.isbn = isbn;
    }

    public String getIsbn() {
        return isbn;
    }

    // überschreiben
    public void print() {
        System.out.println(getName()+" "+isbn);
        super.print(); // print von Artikel
    }
}
```

```
public class Shop {
    public static void main(String[] args) {
        Artikel a2 = new Buch("Java","123",9.9);
        a2.print(); // geht
        Buch b = (Buch)a2; // Type-Cast nötig
        String isbn = b.getIsbn();
    }
}
```

private	nicht vererbt
public	vererbt
protected	vererbt

Interface

```
public class Box implements Produkt {
    private int gang;
    private String size;

    public Box(int gang, String size) {
        this.size = size;
        this.gang = gang;
    }

    public String getLagerOrt() {
        return Integer.toString(gang);
    }

    public String getSize() {
        return size;
    }
}
```

```
public interface Produkt {
    public String getLagerOrt();
}
```

```
ArrayList<Produkt> lager;
lager = new ArrayList<Produkt>();
lager.add(new Box(12,"M"));

for (Produkt a: lager) {
    System.out.println(a.getLagerOrt());
    if (a instanceof Box) {
        System.out.println(((Box)a).getSize());
    }
}
```

Generics

```
Kunde kunde = new Kunde("Herr", "Meier");
Safe<Kunde> safe = new Safe<Kunde>(kunde);
Kunde secret = safe.getSecret("zhaw");
```

Lambda Ausdrücke

```
public class LambdaExample {
    interface StringProcessor{
        char run(String t);
    }

    public static void main(...) {
        StringProcessor s1, s2;
        s1 = x -> x.charAt(0);
        s2 = x -> x.charAt(x.length()-1);

        String s = "Hallo";
        System.out.println(s1.run(s)); //H
        System.out.println(s2.run(s)); //o
    }
}
```

```
public class Safe<T> {
    private T secretObject;

    public Safe(T secretObject) {
        this.secretObject = secretObject;
    }

    public T getSecret(String pw) {
        if(pw.equals("zhaw")) {
            return secretObject;
        }
        return null;
    }
}
```

Regex

[abc]	a, b oder c	\d	[0-9]
[^abc]	alles ausser [abc]	\D	[^0-9]
[a-d]	a bis d	\s	Leerschlag
.	beliebiges Zeichen	\S	[^\s]
^	Zeilenanfang	\w	[a-zA-Z_0-9]
\$	Zeilenende	\W	[^\w]

String-Operationen

```
String text = "Hoi Hänsli, wie gehts?";
int length = text.length(); //22
char first = text.charAt(0); //H
boolean hasOi = text.contains("oi"); //true
int posWie = text.indexOf("wie"); //12
String part = text.substring(6,9); //nsl

boolean match = text.matches(".*H[a|ä]nsl.*");
String [] parts = text.split(",\\s");
String t1 = text.replaceFirst("H[o|e|i]", "Hallo");
String t2 = text.replaceAll("ä|ae", "a");
if (t1.equals(t2)) {System.out.println("gleich");}
```

XY	X dann Y
X Y	X oder Y
X?	Null oder einmal
X*	Null oder öfter
X+	Einmal oder öfter
X{n}	Genau n mal
X{n,}	n bis unendlich mal
X{n,m}	mindestens n mal, maximal m mal

Streams

```
import java.util.stream.Stream;

Stream.of("Hans", "Bea", "Pete")...
Stream.of(2, 4, 5, 1)...
```

```
int [] myArray = {4, 7, 3, 1};
Arrays.stream(myArray)...
```

```
ArrayList<Kunde> kunden = ...
kunden.stream()...
```

```
// quadratwerte ausgeben (9,36,49,25)
Stream.of(3,6,7,5).map(x -> {return x*x;})
    .forEach(System.out::println);
```

```
// zähle unterschiedliche Zahlen (3)
long c = Stream.of(3,7,4,3).distinct().count();
```

```
// Kundinnen sortiert nach Alter ausgeben
ArrayList<Kunde> kunden = ...
List<Kunde> kundinnen = kunden.stream()
    .filter(x -> x.getAnrede().equals("Frau"))
    .sorted(Comparator.comparing(Kunde::getAlter))
    .collect(Collectors.toList());
```

Statische Komponenten von Klassen

```
public class KreisMain {
    public static void main(...) {
        double r = 12.3;
        // Kein new!
        double flaeche = Kreis.area(r);
        double pi = Kreis.pi;
    }
}
```

```
public class Kreis {
    public static double pi = 3.14;

    public static double area(double r) {
        return r*r*pi;
    }
}
```

JSON Syntax

Element	Darstellung
Objekt	{...}
Array	[...]
Attribut	"Name": Wert
Mehrere	,

Wert	Beispiel
Zahl	12
Text	"Test"
Boolscher Wert	true
Nullwert	null

JSON-Import und -Export mit GSON

```
import java.io.*;
import com.google.gson.*;

public class GsonTest {
    public static void main(String[] args) {
        Gson gson = new GsonBuilder().create();
        try {
            FileReader read = new FileReader("myFile.json");
            Contacts b2b = gson.fromJson(read, Contacts.class);
            read.close();

            b2b.contacts.add(new Person("Sue", 43));

            FileWriter write = new FileWriter("myFile.json");
            gson.toJson(b2b, write);
            write.close();
        }
        catch (Exception e) {System.out.println(e);}
    }
}
```

```
public class Person {
    public String name;
    public int age;

    public Person(String n, int a) {
        name = n;
        age = a;
    }
}
```

```
import java.util.ArrayList;

public class Contacts {
    public String group;
    public ArrayList<Person> contacts;
}
```

```
{"group": "B2B",
 "contacts": [
   {"name": "Hans",
    "age": 34}
 ]}
```

```
{"group": "B2B",
 "contacts": [
   {"name": "Hans",
    "age": 34},
   {"name": "Sue",
    "age": 43}
 ]}
```

Pakete

```
package PaketC;
import PaketA.*;

public class Klasse3 {
    public static void main(String[] args) {
        Klasse1 k1 = new Klasse1();
        PaketB.Klasse2 k2 = new PaketB.Klasse2();
    }
}
```

```
package PaketA;
public class Klasse1 {
    //...
}
```

```
package PaketB;
public class Klasse2 {
    //...
}
```