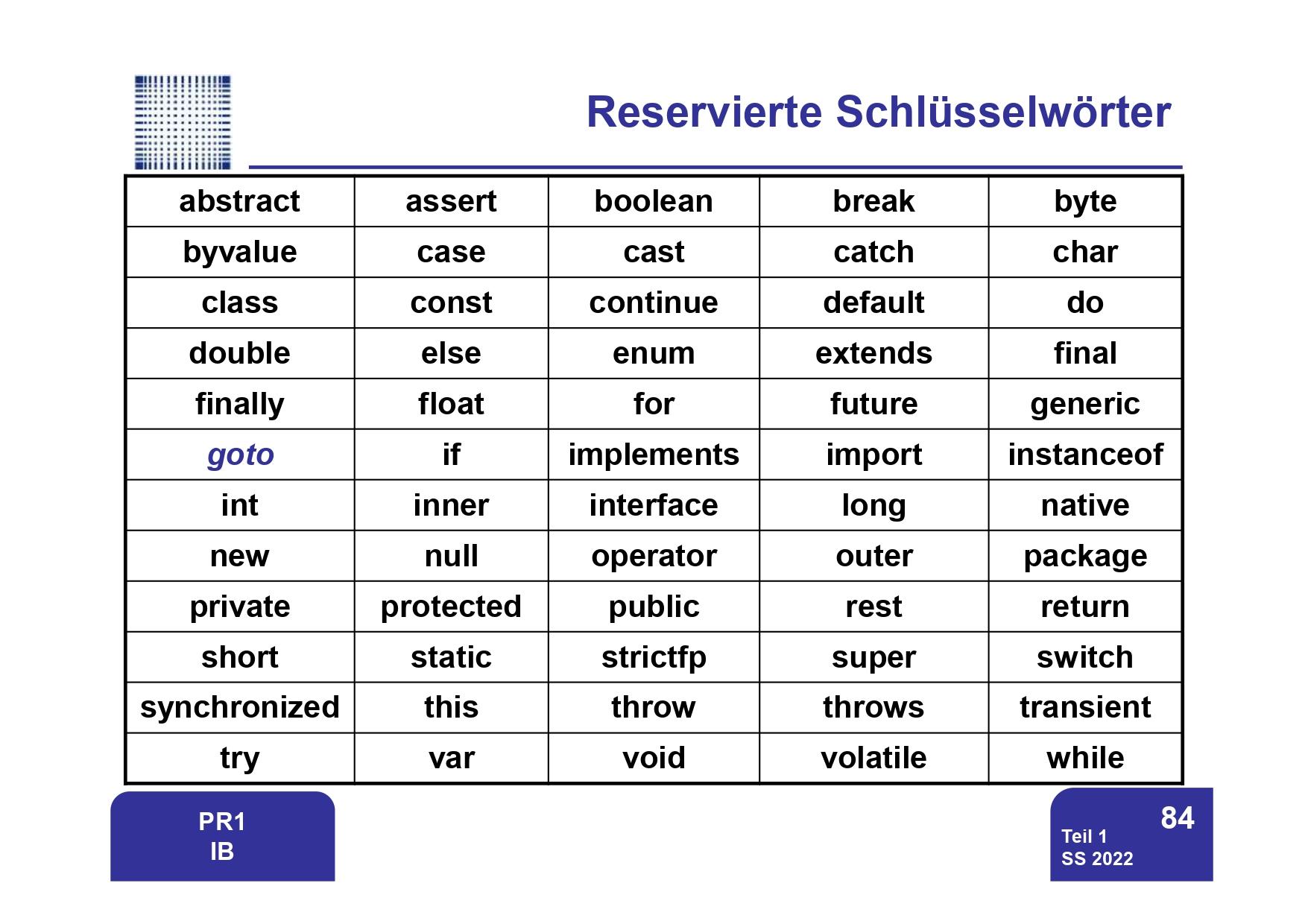
# Syntax Variable

| **Python** | **<variableName> = <variableParameters>** |
| --- | --- |
| **Java** | : **<variableType> <variableName> = <variableParameters>** |
| **JavaScript** | : **var/let/const <variableName> = <variableParameters>** |

**Taboo Variable names:**



*from "Vorlesungsskript SS 2022, Teil 1" of PR1 - Klaus*

# 

- end

# Variable

## Numeric Type

For the Numeric type keywords, u must add an b for byte (120b), (float b = 3,14f)

Because the system has the Integer and the Double as default, without the letters

**Integer / Ordinal (?) / Kardinal (?) Type [ganzzahlige Datentypen]:**

* byte (number, 8 bits)
  + [3 digit] sign255 == 2^8 -1)
  + from -128b to 127b == from -2^7 to 2^7 -1
* short (number, 16 bits
  + [5 digit] sign65.535)
  + from -32.768 to 32.767
* int (number, 32 bits
  + [10 digit] sign4.294.967.295)
  + from 2.147.483.648 to 2.147.483.647
* long (number, 64 bits) heigth\_1 = 180; // = 0
  + [20 digit] sign18.446.744.073.709.551.615
  + from –9.223.372.036.854.775.808 to 9.223.372.036.854.775.807,
  + (ca. -9\*10^18 to 9\*10^18)

**Character**

* char (a character, 8 bits) gender = 'm'; // = ‘\000’

**Floating Type [Fließ-/Gleitkomma Zahlen] : [...]**

Fließkomma zahlen sollte man nicht vergleicht da die nicht genau sind (Foglie 112 -113)

* float (float number, 32 bits)
  + Mantis (?)
  + 7 signifikante Stellen
    - float a = 123456789;

a = a + 1;

= 1,23456792E8

* + from 1.40239846\*10^-45 to 3.40282347\*10^38
* double (float number, 64 bits) = 1.80; // = 0.0

  + from 4.94065645841246544 \* 10^-324 to 1.79769313486231570 \* 10^308

For floating numbers u can also make double.POSITIVE\_INFINITY or the NEGATIVE one

The answer will be “Infinity” of “-Infinity”

**NaN = Not a Number** [kein Zahl]

0d / 0d = NaN ⇒ you can divide 0 by 0

Float.NaN and Double.NaN = NaN

Float.POSITIVE\_INFINITY + Float.NEGATIVE\_INFINITY = NaN

Once a Variable is NaN, none of an operation(Relational Operator) can change it.

Also: NaN ≠ NaN

but: Float.isNaN = 0f / 0f

## Boolean

boolean (true or false, 1 byte) // = false

boolean a = false | ist das gleiche als | !a

2 Boolean with different Operators

* If == is used on 2 boolean keywords
  + The result is true if both are true or false
  + If they are not identical, the result will be false
* If != is used, then it is the opposite. The Operator ^ is same as !=
* If ! is used, then change the result (from true to false, from false to true)
* & not short-circuiting - boolean logical AND
  + only if both boolean are true, the result will be true
* && short-circuiting - logical AND
  + same function as &, but it works only if the first one boolean is already true

(if the first one is not true), the result will always be false

* | not short-circuiting - boolean logical OR
  + result true, if one of the is true
* || short-circuiting - logical OR
  + same as &&, but the first one must be false

**Composite** Type:

* String (text) name = "Rusilowski" // = Null

## String

next().charAt(0)

| **Python** | **Java** | **JavaScript** |
| --- | --- | --- |
| **.lower()** | **<variableName>.toLowerCase();** | **- // -** |
| **.upper()** | **<variableName>.toUpperCase();** | **- // -** |
|  | **<variableName> += “<addText>”;** |  |
| **- // -** | **<variableName>.replace(‘<oldChar>’,’<newChar>’);** |  |

# Array

Merke: Die Länge eines Arrays ist fix und kann nicht nachträglich geändert werden (höchstens neu initialisiert werden) da dabei gehen alle alten Inhalte verloren.

//initialisiert eine Array mit 10 ints // es fängt immer von 0, und geht bis n-1:

int[] ArrayA = new int[10];

ArrayA[0] = …

…

ArrayA[9] = …

//ein Array mit vorgegebenen Werten

int[] y = {1, 1, 2, 3, 5, 8, 13};

&

z = new int[] { 0, 8, 15 };

))

public static void print(Int[][] matrix) {} = (idk)

in JShell:

int[] würfel = new int[5];

würfel ⇒ int[5] { 0, 0, 0, 0, 0 }

würfel[3]

$2 ==> 0

würfel[3] = -123

$3 ==> -123

würfel

würfel ==> int[5] { 3, 0, 0, 6, 4 }

Empty String Array: public String[] product = new String[]{};

r2.products = new String[]{"Carne", "Patate"};

System.out.println(r2.products[0] + " " + r2.products[1]);

String[] increaseArray = new String[products.length + 1];

## Double Array

**variableType[row][col]**

**variableType[][] variableName = {{a1, a2, a3},{b1, b2, b3},{c1, c2, c3},{d1, d2, d3}}**

row1 - row2 - row3 - row4

**variableType[][] variableName = {{a1, a2, a3},{b1, b2, b3},{c1, c2, c3},{d1, d2, d3}}**

col1 - col2 - col3

**variableName.length** // the Number of Array, within the main Array = 4

**variableName[position].length** // the number of value inside the Array at the mentioned position

* + **Math.rMath.random()** - will generate a number between 0 and < 1. **andom()** - will generate a number between 0 and < 1.

## Char

<stringVariable>**.toCharArray();**

## Remove Elements (js)

[9 Ways To Remove 🗑️ Elements From A JavaScript Array 📇[Examples]](https://love2dev.com/blog/javascript-remove-from-array/)

* pop - Removes from the **end** of an Array
* shift - Removes from the **beginning** of an Array
* splice - removes from a specific Array **index**
* filter - allows you to programatically remove elements from an Array

[Array.prototype.splice() - JavaScript | MDN](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/splice)

delete <varName>;

<varName>.remove();

– end

## Tipp:

1. If you divide two Integers, the answer will be an Integer “without number after the commas”.
2. Even if you create a Double, by dividing two Integers, the answer will be n.0.

Bsp.:

int a = 3

int b = 2

double A = 3

double B = 2

1. a / b  
   ⇒ 1

double Z1 = a / b;  
⇒ 1.0

1. a / B or A / b

⇒ 1.5

double Z2 = (double) (a / b);

⇒ 1.5

int x = (int) (3 / 4.0)

.split()

String s = "Hallo Welt wie geht es Dir";

var words = s.split(" ");

System.out.println(words[i]); // to outprint with a loop

-end

# Dictionary (js)

[JavaScript Basics: How to create a Dictionary with Key/Value pairs | Chris Pietschmann](https://pietschsoft.com/post/2015/09/05/javascript-basics-how-to-create-a-dictionary-with-keyvalue-pairs)

function abc(){

var myList = {

FirstName: "Chris",

"one": 1,

1: "some value"

};

console.log(myList.FirstName);

console.log(myList["one"]);

console.log(myList[1]);

}

Add new Key with Value

itemList["Bone"] = false;

-end

# Cast

Cast =^ Conversion of the variants into others variants.

Use the int x = (int) ... only for smaller Data Types

Not this one:

byte ⇒ short

byte ⇒ int

byte ⇒ long

short ⇒ int

short ⇒ long

int ⇒ long

char ⇒ short

char ⇒ int

char ⇒ long

Double into Integer

int x = (int) (double var);

Integer/Double into String:

String.valueOf(int or double var);

Integer.toString(-//-);

Spring into Integer

Integer.parseInt(String var);

Char into Integer

int a = (int) b

If u divide 2 int, only the integral number will show up (if the program didn’t crash)

Wenn man ein Wert in Byte mit einem Wert von Short addiert, das Ergebnis wird Short sein, da es im Byte nicht passt,

Falls es auch im Short nicht passen würde, dann wird dieses Zahl konvertiert zu der gleichen Datentype die angewendet wurde (hier Short)

Wenn eine operation durchgeführt wird, und der Datentype an seine Grenzen kommt, fängt der Loop wieder.

byte a = (byte) (-128 - 1) = 127

byte b = 127!

b =b + (byte)1;

**Primitive Type**

Ist der Zieltyp “größer” als der Typ des Wertes, dann ist der Cast unnötig, Java macht den Cast implizit:

byte ⇒ short ⇒ int ⇒ long ⇒ float ⇒ double

chat ⇒ int

**Numeric Type**

the integer number can not record floatings numbers:

integer = floating ⇒ error

int a = 1.0; ⇒ error

but the floating numbers can cast the integers:

floating = integer ⇒ floating

double b = 1; ⇒ b = 1.0

**Floating Type**

Die Konvertierung erfolgt Schrittweise:

double a;

a = 3.0f + 4;

⇒ a = 3.0f + 4.0f;

⇒ a = 7.0f;

⇒ a = 7.0d;

# Operators

Only the integral numbers are associative

And only if the keywords are the same

a = 8

b = 27

c = 64

----------------------------------------------

a / b / c = x

a / c / b = x

8 | 27 | 64 = 13.824 | 0,0046296…

8 | 64 | 27 = 13.824 | 0,0046296…

----------------------------------------------

c / a / b = y

c / b / a = y

64 | 8 | 27 = 13.824 | 0,296…

64 | 27 | 8 = 13.824 | 0,296…

## Comments

| **Python** | **#** this is a single line comment | **‘’’** multi line comment **‘’’** |
| --- | --- | --- |
| **Java** | **//** this is a single line comment | **/\*** multi line comment **\*/** |
| **JavaScript** |  |  |
| **Html** | **< ! - -** Both single & multi line **- - >** | -//- |

## Print

| **Python** | **print()** |
| --- | --- |
| **Java** | **System.out.println();** |
| **JavaScript** | **console.log();** |

## Timer

public static void main(String[] args) throws InterruptedException {

System.*out*.print("test");

Thread.*sleep*(3000);

System.*out*.print('\r');

System.*out*.print("lulz");

}

## Pre- & Postfix

a = 1, b;

b = a++; ⇒ a = 2; b = 1

b = ++a; ⇒ a = 2 ;b = 2

.length

for (int i = 0; i < array.length; i++)

but

System.out.println( a.length() );