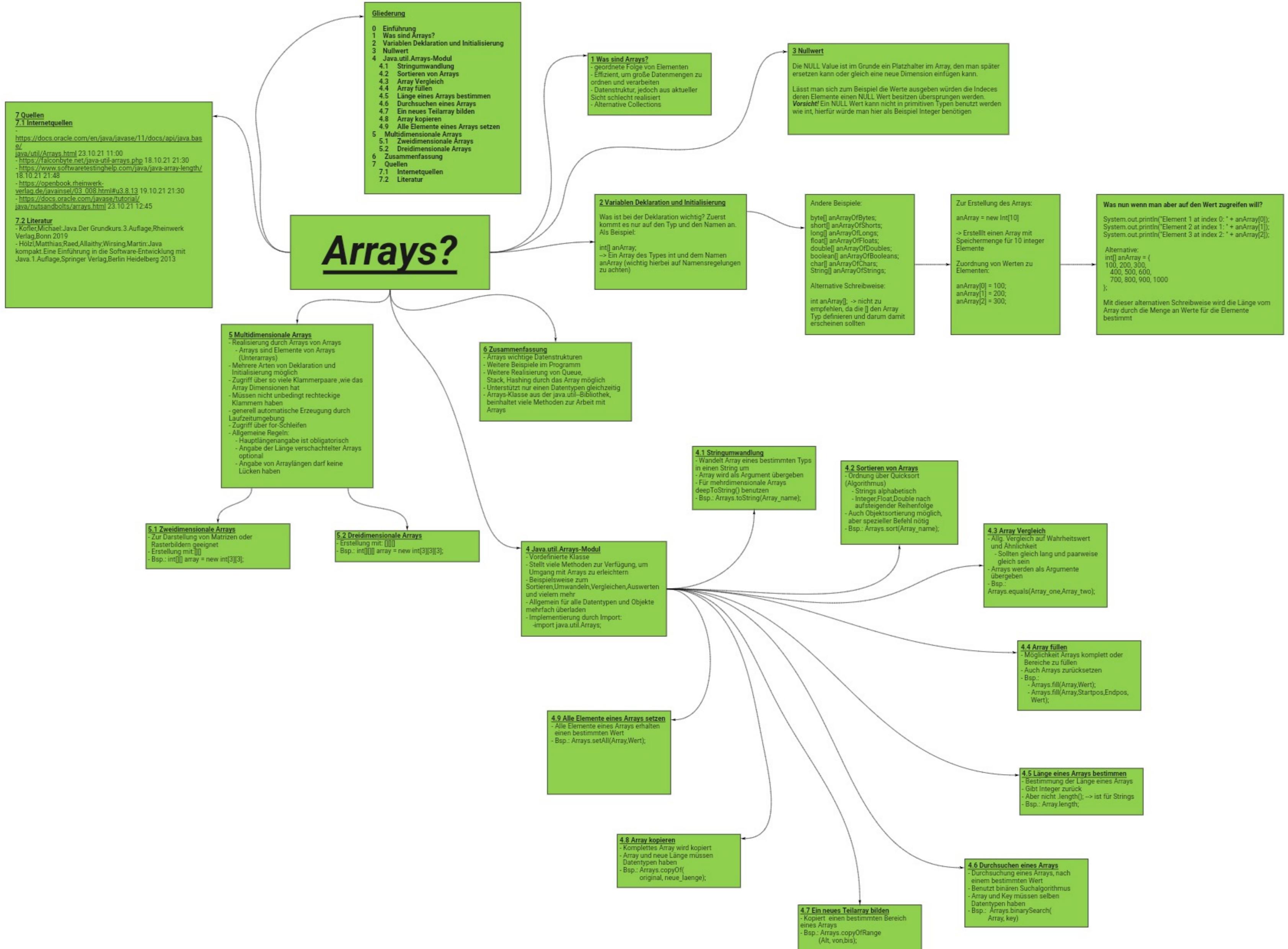


# Arrays?



# Gliederung

- 0 Einführung
- 1 Was sind Arrays?
- 2 Variablen Deklaration und Initialisierung
- 3 Nullwert
- 4 Java.util.Arrays-Modul
  - 4.1 Stringumwandlung
  - 4.2 Sortieren von Arrays
  - 4.3 Array Vergleich
  - 4.4 Array füllen
  - 4.5 Länge eines Arrays bestimmen
  - 4.6 Durchsuchen eines Arrays
  - 4.7 Ein neues Teilarray bilden
  - 4.8 Array kopieren
  - 4.9 Alle Elemente eines Arrays setzen
- 5 Multidimensionale Arrays
  - 5.1 Zweidimensionale Arrays
  - 5.2 Dreidimensionale Arrays
- 6 Zusammenfassung
- 7 Quellen
  - 7.1 Internetquellen
  - 7.2 Literatur

# 1 Was sind Arrays?

- geordnete Folge von Elementen
- Effizient, um große Datenmengen zu ordnen und verarbeiten
- Datenstruktur, jedoch aus aktueller Sicht schlecht realisiert
- Alternative Collections

## 2 Variablen Deklaration und Initialisierung

Was ist bei der Deklaration wichtig? Zuerst kommt es nur auf den Typ und den Namen an.  
Als Beispiel:

→  
`int[] anArray;`

→ Ein Array des Types int und dem Namen  
anArray (wichtig hierbei auf Namensregelungen  
zu achten)

## Andere Beispiele:

```
byte[] anArrayOfBytes;  
short[] anArrayOfShorts;  
long[] anArrayOfLongs;  
float[] anArrayOfFloats;  
double[] anArrayOfDoubles;  
boolean[] anArrayOfBooleans;  
char[] anArrayOfChars;  
String[] anArrayOfStrings;
```

## Alternative Schreibweise:

int anArray[]; -> nicht zu empfehlen, da die [] den Array Typ definieren und darum damit erscheinen sollten

Zur Erstellung des Arrays:

`anArray = new Int[10]`

-> Erstellt einen Array mit Speichergröße für 10 integer Elemente

→ Zuordnung von Werten zu Elementen:

`anArray[0] = 100;`  
`anArray[1] = 200;`  
`anArray[2] = 300;`

## Was nun wenn man aber auf den Wert zugreifen will?

```
System.out.println("Element 1 at index 0: " + anArray[0]);  
System.out.println("Element 2 at index 1: " + anArray[1]);  
System.out.println("Element 3 at index 2: " + anArray[2]);
```

Alternative:

```
→ int[] anArray = {  
    100, 200, 300,  
    400, 500, 600,  
    700, 800, 900, 1000  
};
```

Mit dieser alternativen Schreibweise wird die Länge vom Array durch die Menge an Werte für die Elemente bestimmt

### 3 Nullwert

Die NULL Value ist im Grunde ein Platzhalter im Array, den man später ersetzen kann oder gleich eine neue Dimension einfügen kann.

Lässt man sich zum Beispiel die Werte ausgeben würden die Indeces deren Elemente einen NULL Wert besitzen übersprungen werden.

**Vorsicht!** Ein NULL Wert kann nicht in primitiven Typen benutzt werden wie int, hierfür würde man hier als Beispiel Integer benötigen

## 4 Java.util.Arrays-Modul

- Vordefinierte Klasse
- Stellt viele Methoden zur Verfügung, um Umgang mit Arrays zu erleichtern
- Beispielsweise zum Sortieren, Umwandeln, Vergleichen, Auswerten und vielem mehr
- Allgemein für alle Datentypen und Objekte mehrfach überladen
- Implementierung durch Import:
  - import java.util.Arrays;

## 4.1 Stringumwandlung

- Wandelt Array eines bestimmten Typs in einen String um
- Array wird als Argument übergeben
- Für mehrdimensionale Arrays  
deepToString() benutzen
- Bsp.: `Arrays.toString(Array_name);`



## 4.2 Sortieren von Arrays

- Ordnung über Quicksort  
(Algorithmus)
  - Strings alphabetisch
  - Integer,Float,Double nach  
aufsteigender Reihenfolge
- Auch Objektsortierung möglich,  
aber spezieller Befehl nötig
- Bsp.: `Arrays.sort(Array_name);`



## 4.3 Array Vergleich

- Allg. Vergleich auf Wahrheitswert und Ähnlichkeit
  - Sollten gleich lang und paarweise gleich sein
- Arrays werden als Argumente übergeben
- Bsp.:

```
Arrays.equals(Array_one,Array_two);
```

## 4.4 Array füllen

- Möglichkeit Arrays komplett oder Bereiche zu füllen
- Auch Arrays zurücksetzen
- Bsp.:
  - `Arrays.fill(Array,Wert);`
  - `Arrays.fill(Array,Startpos,Endpos,Wert);`



## 4.5 Länge eines Arrays bestimmen

- Bestimmung der Länge eines Arrays
- Gibt Integer zurück
- Aber nicht .length(); --> ist für Strings
- Bsp.: Array.length;

## 4.6 Durchsuchen eines Arrays

- Durchsuchung eines Arrays, nach einem bestimmten Wert
- Benutzt binären Suchalgorithmus
- Array und Key müssen selben Datentypen haben
- Bsp.: `Arrays.binarySearch(Array, key)`



## 4.7 Ein neues Teilarray bilden

- Kopiert einen bestimmten Bereich eines Arrays
- Bsp.: `Arrays.copyOfRange  
(Alt, von,bis);`

## 4.8 Array kopieren

- Komplettes Array wird kopiert
- Array und neue Länge müssen Datentypen haben
- Bsp.: `Arrays.copyOf(  
original, neue_laenge);`



## 4.9 Alle Elemente eines Arrays setzen

- Alle Elemente eines Arrays erhalten einen bestimmten Wert
- Bsp.: `Arrays.setAll(Array,Wert);`



## 5 Multidimensionale Arrays

- Realisierung durch Arrays von Arrays
  - Arrays sind Elemente von Arrays  
(Unterarrays)
- Mehrere Arten von Deklaration und Initialisierung möglich
- Zugriff über so viele Klammerpaare ,wie das Array Dimensionen hat
- Müssen nicht unbedingt rechteckige Klammern haben
- generell automatische Erzeugung durch Laufzeitumgebung
- Zugriff über for-Schleifen
- Allgemeine Regeln:
  - Hauptlängenangabe ist obligatorisch
  - Angabe der Länge verschachtelter Arrays optional
  - Angabe von Arraylängen darf keine Lücken haben

## 5.1 Zweidimensionale Arrays

- Zur Darstellung von Matrizen oder Rasterbildern geeignet
- Erstellung mit: [] []
- Bsp.: int[][] array = new int[3][3];



## 5.2 Dreidimensionale Arrays

- Erstellung mit: `[][][]`
- Bsp.: `int[][][] array = new int[3][3][3];`



## 6 Zusammenfassung

- Arrays wichtige Datenstrukturen
- Weitere Beispiele im Programm
- Weitere Realisierung von Queue, Stack, Hashing durch das Array möglich
- Unterstützt nur einen Datentypen gleichzeitig
- Arrays-Klasse aus der java.util-Bibliothek, beinhaltet viele Methoden zur Arbeit mit Arrays

## 7 Quellen

### 7.1 Internetquellen

- <https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/Arrays.html> 23.10.21 11:00  
- <https://falconbyte.net/java-util-arrays.php> 18.10.21 21:30  
- <https://www.softwaretestinghelp.com/java/java-array-length/> 18.10.21 21:48  
- [https://openbook.rheinwerk-verlag.de/javainsel/03\\_008.html#u3.8.13](https://openbook.rheinwerk-verlag.de/javainsel/03_008.html#u3.8.13) 19.10.21 21:30  
- <https://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html> 23.10.21 12:45

### 7.2 Literatur

- Kofler, Michael: Java. Der Grundkurs. 3. Auflage, Rheinwerk Verlag, Bonn 2019
- Hölzl, Matthias; Raed, Allaithy; Wirsing, Martin: Java kompakt. Eine Einführung in die Software-Entwicklung mit Java. 1. Auflage, Springer Verlag, Berlin Heidelberg 2013