Arrays in Java unterscheiden sich dadurch von Arrays in anderen Programmiersprachen, dass sie *Objekte* sind. Obwohl dieser Umstand in vielen Fällen vernachlässigt werden kann, bedeutet er dennoch:

- dass Array-Variablen *Referenzen* sind,
- dass Arrays Methoden und Instanz-Variablen besitzen,
- dass Arrays zur Laufzeit erzeugt werden.

Arrays aus primitiven Datentypen: Deklaration und Initialisierung:

```
int[] prim ;
boolean[] bollArray ;


prim = new int[5] ;
boolArray = new boolean[3] ;
prim[0] = 2 ;
prim[1] = 3 ;
prim[2] = 5 ;
prim[3] = 7 ;
prim[4] = 11 ;
boolArray[0] = true ;
boolArray[1] = false ;
boolArray[2] = true ;

int[] prim = {2, 3, 5, 7, 11}
boolean[] bollArray = {true, false, true}
```

Zugriff auf Array-Elemente:

```
System.out.println("Elemente von prim: " + prim[0] + " "
   + prim[1] + " " + prim[2] + " " + prim[3] + " " + prim[4])
System.out.println("Anzahl Elemente von prim: "
          + prim.length )
System.out.println("Anzahl Elemente von prim: "
          + prim.length )
System.out.println(("Anzahl Elemente von boolArray: "
          + boolArray.length)
```

Mehrdimensionale Arrays

```
int[][] quadArray = new int[2][3] ;
quadArray[0][0] = 1 ;
quadArray[0][1] = 2 ;
quadArray[0][2] = 3 ;
```

Arrays aus Objekten: Deklaration und Initialisierung:

```
Object[] objArray ;
Auto[] autoArray ;
Student[] studentArray ;


objArray = new Object[5] ;
autoArray = new Auto[3] ;
studentArray = new Student[4] ;


objArray[0] = new String("Beispiel") ;
objArray[1] = "Zeichenkette" ;
objArray[2] = new Auto( ... ) ;
objArray[3] = new Student( ... ) ;
objArray[4] = new BeliebigeKlasse( ... ) ;

autoArray[0] = new Auto( ... ) ;
autoArray[1] = new Auto( ... ) ;
// autoArray[2] = new Student( ... ) ;   Compilerfehler!!!

studentArray[0] = new Student( ... ) ;
studentArray[1] = new Student( ... ) ;
studentArray[2] = new Student( ... ) ;
studentArray[3] = new Student( ... ) ;
// studentArray[4] = new Student( ... ) ;   Laufzeitfehler !!!
```