Array in Groovy

Als Array wird eine Datenstruktur bezeichnet, in der mehrere Elemente des gleichen Typs gespeichert werden können. In der übertragenen Sicht, handelt es sich um einen Schubladenschrank, bei dem in jeder Schublade nur ein Element abgelegt werden kann. Auch die weiteren Eigenschaften eines Arrays lassen sich darauf übertragen: Bei einem Array ist bereits zu Beginn festgelegt ist, wie viele Elemente maximal gespeichert werden können. Die einzelnen Plätze des Arrays sind durchnummeriert, so dass unter Angabe der Nummer direkt auf die gespeicherten Elemente zugegriffen werden kann, bzw. sie an entsprechender Stelle abgelegt werden können. Die Nummerierung der Felder startet dabei in Groovy automatisch mit 0.

Um in Groovy genau diese Arrays nutzen zu können muss man teilweise auf die Funktionen des darunterliegenden Java zurückgreifen. Daher ist es immer nötig den Typ der zu speichernden Objekte, also den Klassennamen, mit anzugeben.

Um ein Array zu erzeugen, muss man entweder die Anzahl der Felder angeben oder die Werte der einzelnen Felder aufführen.

```
1 strArray = new String[5]
2 strArray2 = ["foo", "bar"] as String[]
3 zahlenArray = [4, 8, 7, 9, 31] as Integer[]
```

Um die Werte der Felder zu setzen, muss der Name des Array mit der Angabe der Feldnummer in eckigen Klammern angegeben werden.

```
4 strArray[0] = 'boo'
5 strArray[1] = 'far'
6 zahlenArray[3] = 15
```

Analog muss dieses auch genutzt werden, wenn man einen Wert aus einem Feld des Arrays nutzen möchte

```
7 println(strArray2[1]) //gibt bar aus
8 println(zahlenArray[0] + zahlenArray[4]) //gibt 35 aus
```

Die Anzahl der Felder eines Arrays ist immer in seinem Attribut »length« gespeichert.

```
9 println(strArray2.length) // gibt 2 aus
```

Neben dem hier aufgeführten eindimensionalen Arrays, kann man auch weitere Dimensionen nutzen. So wird im folgenden Beispiel ein 3 mal 4 Array für Zahlen erstellt und genutzt

```
10 mehrdim = new Integer[3][4]
11 mehrdim[1][2] = 3
```

Groovy bietet gegenüber Java noch eine Menge weiter Möglichkeiten bei der Nutzung von Array gegenüber Java. Diese werden an dieser Stelle aber nicht aufgeführt.