

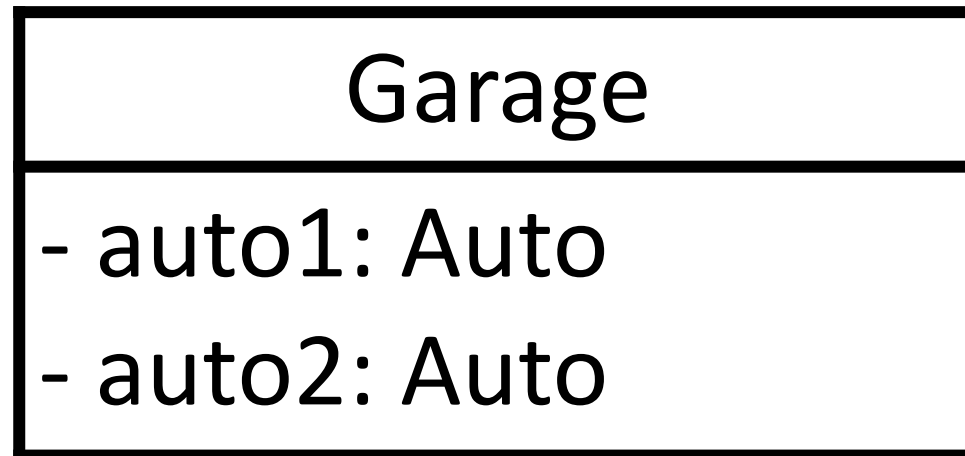
Felder in der Informatik



Quelle: http://www.hotelducommerce.lu/uploads/pics/kornfeld_1024x768_03.jpg, aufgerufen am 16.01.2017

Garage

Stelle dir vor, du programmierst eine Garage.
Jedes Objekt der Klasse Garage referenziert 2
Objekte der Klasse Auto:



bisher...

... wird für jedes Objekt eine eigene Variable deklariert.


Kleine Anzahl von Objekten -> überschaubarer Aufwand

Große Anzahl von Objekten -> großer Aufwand

Szenario: Unsere Garage wird zum Parkhaus

Beispiel Tiefgarage Fürstenried-West

P+R Tiefgarage Fürstenried-West



Neurieder Straße 40
81475 München

Am gleichen Standort: [Parkplatz Fürstenried-West](#)

Höchstparkdauer: 24 Stunden
Stellplätze: 214
Einfahrtshöhe: 2,10 m
ÖPNV-Anbindung: U3

Belegung (Prognose)

Mo.-Fr.	06:00 bis 07:00 Uhr	●
Mo.-Fr.	07:00 bis 08:00 Uhr	●
Mo.-Fr.	08:00 bis 16:00 Uhr	●
Mo.-Fr.	16:00 bis 18:00 Uhr	●
Mo.-Fr.	18:00 bis 06:00 Uhr	●
Sa. und So.	06:00 bis 06:00 Uhr	●

214 Stellplätze
-> 214 Attribute zum Abspeichern von Autos!

Lösung:

Wir benötigen eine „***Datenstruktur***“!

1. Sie soll ***viele Elemente gleichen Datentyps*** fassen können.
2. Der ***Zugriff auf jedes einzelne Element*** muss möglich sein.

Deklarien von Feldobjekten

Notation in Schreibweise analog zu normalen Objekten, nur mit [] hinter dem Datentyp:

```
Auto[] autos; //Feld mit Elementen vom Typ Auto
```

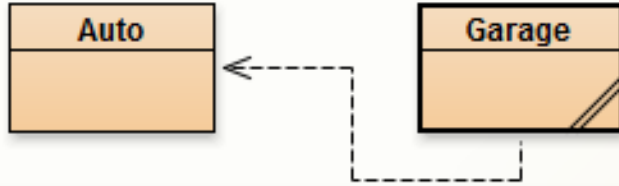
Die angehängten Klammern enthalten die Anzahl der Feldelemente.

```
autos = new Auto[214]; //Feld mit 214 Elementen
```

Zugriff auf die Länge des Elements:

```
int anzahlParkplaetze = autos.length;
```

Mal sehen, was sich getan hat



Garage - AutoGarage

Class Edit Tools Options

Compile Undo Cut Copy Paste Find... Close

```
public class Garage
{
    Auto[] autos = new Auto[214];
}
```

The inspector window is titled 'garage1 : Garage'. It displays a field 'Auto[] autos' with a value of 214. A yellow highlight is around the field name. To the right of the field is a small diagram showing a dot connected to an arrow. Below the field are buttons for 'Inspect', 'Get', 'Show static fields', and 'Close'.

Mal sehen, was sich getan hat

The image shows an IDE interface with several components:

- Class Diagram:** Located at the top left, it shows two classes: `Auto` and `Garage`. A dashed arrow points from `Garage` to `Auto`, indicating a dependency or inheritance relationship.
- Source Code:** Below the diagram, the source code for the `Garage` class is visible:

```
public class Garage  
{  
    Auto[] autos = new Auto[214];  
}
```
- Inspection Window (autos):** A red-bordered window titled `autos : Auto[]` is open on the right. It displays the state of the `autos` array:

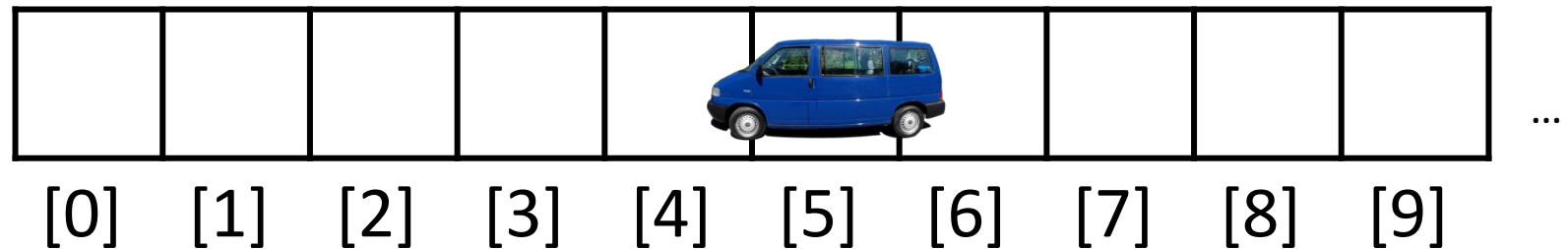
Index	Value
int length	214
[0]	null
[1]	null
[2]	null
[3]	null
[4]	null
[5]	null
[6]	null

Buttons for `Inspect`, `Get`, `Show static fields`, and `Close` are present.
- Inspection Window (garage1):** A red-bordered window titled `garage1 : Garage` is open at the bottom left. It shows the `Auto[] autos` field with a visual representation of an array and a pointer. Buttons for `Inspect`, `Get`, `Show static fields`, and `Close` are present.

null bedeutet, hier wurde noch kein Objekt initialisiert

Zugriff auf die Elemente

Zum *Initialisieren* und *Verwenden* der einzelnen Elemente können wir über deren *Position* Zugriff erlangen:



Initialisieren des Elements mit Index 5:

```
autos[5] = new Auto(„VW“, „T4“, 85, 1997); //Element 5 steht an 6. Stelle!
```

ACHTUNG! Der Informatiker zählt 0, 1, 2, 3...

Die Länge des oben sichtbaren Elements ist 10, es beinhaltet die Elemente 0 bis 9!

Übung

- a) ***Instanziiere*** ein Array vom typ int mit 7 Feldern. Gib jedem Element den Wert seines Indizes. (*Index = Nummer des Elements*)

- b) ***Verwende*** eine While-Schleife für eine automatische Zuweisung der Werte.

- c) ***Implementiere*** eine Methode, die automatisch alle Elemente des Arrays auf der Konsole ausgibt.