# Grundbegriffe Programmieren Cheat Sheet

by Srabudc via cheatography.com/84490/cs/19964/

## Objekt (object)

| | |
|---|---|
| Objektinstanz | *Instanz einer Klasse* oder auch *erzeugtes Objekt* |
| Zustand eines Objekts | Alle aktuellen Werte der Instanzvariablen einer Objektinstanz |

Der Begriff *Objekt* bezeichnet eine **Instanz einer Klasse**.

## Klasse (class)

| | |
|---|---|
| Kopf einer Klasse | enthält unter anderem den Namen der Klasse |
| Innenteil der Klasse | enthält die Implementierung der Klasse mit<br>* Datenfeldern (*fields*)<br>* Konstruktoren (*constructors*)<br>* Methoden (*methods*) |
| Deklaration | Definition der Datenfelder |
| Zugriffsmodifikator (*access modifier*) | Der Zugriffsmodifikator bestimmt die Sichtbarkeit. Verwendet werden oft `private` oder `public`. |

Die Klasse beschreibt eine bestimmte Art von Objekten.
Eine Klasse bestimmt, welche **Methoden** und **Datenfelder** eine **Objektinstanz** haben wird. Sie "existiert" nur einmal als Vorgabe der Struktur (Bauplan / Formular).
Es können beliebig viele **Objektinstanzen** basierend auf der Definition der Klasse erstellt werden.

## Datentypen

| | |
|---|---|
| `int` | ganze Zahl, $-2^{31}$ bis $2^{31}-1$ |
| `String` | Text |
| `float` | Fliesskommazahl mit beschränkter Genauigkeit |
| `double` | Fliesskommazahl mit höherer Genauigkeit |
| `boolean` | `true` oder `false` |

Für Java sind die Datentypen im Kapitel 4 der The Java® Language Specification definiert.

## Lebensdauer von ...

| | |
|---|---|
| Klassenvariablen | ist für die Programmausführung unbeschränkt |
| Instanzvariablen | ist auf die Lebenszeit der Objektinstanz beschränkt |
| Parametern | ist auf die Ausführungszeit der Methode beschränkt |
| lokalen Variablen | ist auf die Ausführungszeit des Code-Blocks beschränkt |

## Datenfelder (fields)

| | |
|---|---|
| Deklaration | Definition der Datenfelder (*field declaration*) |
| Sichtbarkeit | Datenfelder sind in der Regel `private` und damit nur innerhalb der eigenen Klasse direkt zugänglich |
| Attribut | Wird oft als Datenfeld implementiert |

```
private String title = "Ein erster Titel";
```

Siehe auch **Variable**.

## Konstruktoren (constructors)

Die Konstruktoren setzen den **Anfangszustand eines Objekts** (initialisieren das Objekt). Der Aufbau ist **ähnlich der einer Methode** und die Sichtbarkeit ist in der Regel `public`.
Der Name des Konstruktors muss der Klassenname sein. Im Gegensatz zu Methoden haben Konstruktoren keinen Rückgabewert.

```
public Fracti on(int numerator, int denomi nator)
```

## Methoden (methods)

| | |
|---|---|
| Methoden | sind für die Interaktion mit einem Objekt |
| Parameter | sind Werte, die an eine Methode (oder Konstruktor) übergeben werden |
| Signatur | umfasst den Methodennamen und die Parameter einer Methode.<br>Viele Sprachen unterstützen beliebig viele Parameter und maximal einen Rückgabewert. |
| Methodenkopf | Enthält neben der Signatur auch den Rückgabewert und die Sichtbarkeit, z.B. `private int doit();` |
| Rückgabewert | = Aufrufergebnis |
| Datentyp | gibt den Wertebereich von Parametern und Rückgabewert vor<br>Beispiele sind `int`, `boolean`, `...` und Klassen |
| `void` | wird im Programmcode an der Stelle eines Datentyps bei Rückgabewert angegeben, wenn eine Methode keinen Rückgabewert liefern wird |
| Inspektoren | Sondierende Methoden. Beschränkt auf ein Datenfeld sind es Getter-Methoden, z.B. `getPrice` |
| Mutatoren | Verändernde Methoden. Beschränkt auf ein Datenfeld sind es Setter-Methoden, z.B. `setPrice` |




Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

## Parameter

| | |
|---|---|
| aktueller Parameter | Beim Aufruf übergebener Wert. Zum Beispiel `500` in `new Ticket aut oma t(500);` |
| formaler Parameter (*formal parameter*) | Definierter Name des Parameters. Zum Beispiel `ticket preis` in `public Ticket aut oma t(int ticket preis);` |

* Werden durch Komma getrennt zwischen Klammern "(" und ")" im Kopf des Konstruktors oder der Methode definiert
* haben einen Namen und einen Datentyp

`int diceNu mber, Colour diceColour`

## Namensgebung

| | |
|---|---|
| Datenfelder | Beginnen mit Buchstaben, gefolgt von Buchstaben, Zahlen und "_" (kein "-"). |
| Clean Code für Datenfelder | Mit Kleinbuchstaben beginnen und Camel Case verwenden. `reifen druck`, `koerpe rgr oesse`, `geldEi nbe zahlt`, `preisI nFr anken` |
| Methodennamen | Verb, beginnt mit Kleinbuchstaben. `moveLeft`, `print` |
| Setter-Methoden | Methoden, die typischerweise ein Datenfeld auf einen neuen Wert setzten. `setColour`, `setName` |
| Getter-Methoden | Methoden, die typischerweise den Wert eines Datenfelds abfragen. `getName`, `isValid` |
| Konstanten | Alles Grossbuchstaben und "_". `MAX_RADIUS`, `FIRST_ PLAYER` |

## Kommentar (comment)

| | |
|---|---|
| Einfache Kommentare | Bis zum Zeilenende mit `//` (*end-of-line comment*) oder über mehrere Zeilen mit `/* ... */` (*traditional comment*) |
| Javadoc-Kommentare | Beginnen mit zwei Sternen: `/** ... */` und werden in der Regel für Klassen, Konstruktoren und public Methoden geschrieben. |

```
int value; // Kommentar in einer Zeile
/**
 * Kommentar für Javadoc
 */
```

## Variable (variable)

| | |
|---|---|
| Datenfelder | sind **Instanzvariablen** oder **Klassenvariablen** |
| Instanzvariable (*instance variable*) | existiert pro Objekt und ist durch das Fehlen des Schlüsselwortes `static` als Instanzvariable definiert `private String name;` |
| Klassenvariable (*class variable*) | existiert nur einmal und wird mit dem Schlüsselwort `static` als Klassenvariable gekennzeichnet `public static int size;` |
| Parameter (*parameters*) | werden bei Methoden oder Konstruktoren übergeben |
| lokale Variablen (*Local variables*) | werden in einem Code-**Block** (z.B. innerhalb einer Methode) definiert und sind nur innerhalb dieses Blocks verfügbar |

Variablen **speichern Daten** und haben einen Datentyp, siehe *Kinds of variables* in der JLS



Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com