Der Konstruktor

Der Konstruktor ist eine besondere Methode, die nicht ausdrücklich aufgerufen werden kann, sondern automatisch in dem Moment ausgeführt wird, wenn ein Objekt der Klasse mit **new** erzeugt wird. Der konkrete Inhalt des Konstruktors kann vom Programmierer festgelegt werden – zum Beispiel, um Attributwerte vorgabemäßig zu setzen oder individuelle Werte beim Konstruktoraufruf zu übernehmen (siehe unten: Konstruktor-Argumente).

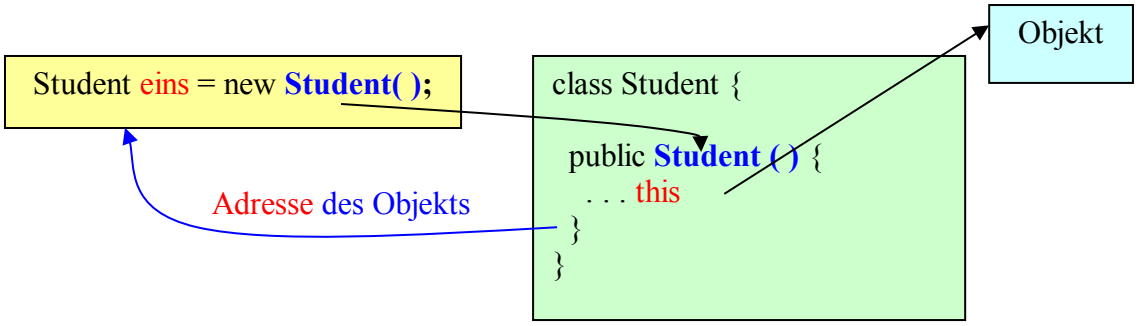
Ein Konstruktor hat folgende Merkmale:

- sein Name entspricht exakt dem Klassennamen,
- sein Zugriffskennzeichner ist public,
- ein Konstruktor hat *keinen* Rückgabetyp (auch nicht void).

```
public Student( ) {
      . . .
}
```

Ein Konstruktor darf deshalb keinen Rückgabewert haben, weil er durch den speziellen Ablauf des Objekterzeugungsprozesses bereits definiert ist. Der Rückgabewert ist nämlich nichts anderes als die Heap-ADRESSE des neu erzeugten Objekts, die an die Referenz zurückgegeben wird. Ein Konstruktor, der fälschlicherweise mit einem Rückgabetyp versehen wurde, wird vom Compiler nicht als solcher erkannt, sondern als „normale Methode“ behandelt.

Die Bildschirm-Ausgabe des folgenden Beispiel-Programms liefert identische Adressen. Dies beweist, dass die Objektadresse innerhalb der Klasse durch den **this**-Zeiger repräsentiert wird. Diese Adresse ist es, die im „unsichtbaren“ Konstruktor-Rückgabewert an die Referenz (Objektnamen) übergegeben wird. Generell gilt, dass alle Methoden einer Klasse nur die Adresse des jeweils „aktuellen Objekts“ (das die Methode aufgerufen hat) als Inhalt des this-Zeigers kennen.




```
class Student {

    public Student( ) {

        System.out.println( "THIS:     " + this );
    }
}

//////////////////////////////////////////////////

class StudentTest {

    public static void main(String [] args) {

        Student eins = new Student( );

        System.out.println( "Referenz: " + eins );
    }
}
```

```
THIS:      Student@7ced01
Referenz: Student@7ced01
```

In einer Klasse muss nicht zwingend ein Konstruktor vorhanden sein. Falls es keinen gibt, wird vom Compiler automatisch ein Standard-Konstruktor (ohne Übergabe-Argumente, mit leerem Rumpf) erzeugt. Dies erkennt man durch Dekompilieren einer Java-Klasse.

Achtung: Sobald es einen Konstruktor mit Parametern gibt (siehe unten), wird kein Standard-Konstruktor vom Compiler erzeugt.

Ablauf der Objekterzeugung

Bei der Objekterzeugung werden die Attribute prinzipiell **vor** der Abarbeitung des Konstruktors angelegt. Dies ist auch sinnvoll, denn ansonsten könnte im Konstruktorrumpf nicht mit ihnen gearbeitet werden. Es ist in Java möglich (im Gegensatz zu C++), Attribute ausdrücklich mit Anfangswerten zu belegen. Solche vor-initialisierte Attribute sind nicht objektspezifisch, sondern gelten für ALLE Objekte gleichermaßen. Natürlich können die Vorgabewerte später geändert werden und gelten dann nur für das jeweilige Objekt. Eine Initialisierung auf Attributebene ist nur in solchen Fällen empfehlenswert, wenn alle Objekte tatsächlich gleiche Attributwerte haben sollen. Dann ist es jedoch sinnvoller, diese mit dem Kennzeichner `static` zu versehen (siehe Kapitel Zugriffskennzeichner).

1. Attribute werden angelegt und initialisiert
2. Konstruktor wird abgearbeitet.

```
class Student {

    String name = "leer";

    public Student( ) {

        System.out.println( "Name ist " + name );
    }
}
////////////////////////////////////////////////

class StudentTest {

    public static void main(String [] args) {

        Student eins = new Student( );
        Student zwei = new Student( );
    }
}
```

```
Name ist leer
Name ist leer
```