

Ein kleines Programm...

```
int x = 5;
```

```
int y = x;
```

```
x = x + 2;
```

- Welchen Wert hat x?
- Welchen Wert hat y?

Noch ein kleines Programm...

```
BösesMonster b1 = new BösesMonster(50, 100, 5);
```

```
BösesMonster b2 = b1;
```

```
b1.bewege();
```

```
b2.bewege();
```

- Welches Ergebnis erwartet ihr?

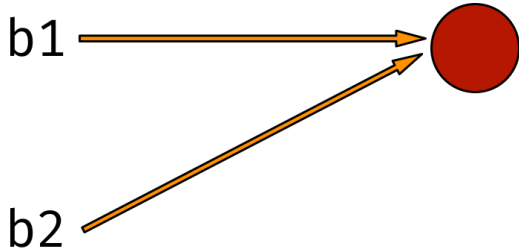
Referenzen

- Der Operator **new** erzeugt ein Objekt der angegebenen Klasse und liefert eine **Referenz** auf dieses Objekt zurück
 - Eine Referenz ist ein Verweis/ein “Zeiger” auf das eigentliche Objekt im Speicher
- Genau genommen ist also bspw. **b1** kein Objekt der Klasse **BösesMonster**, sondern eine Referenz auf ein Objekt der Klasse **BösesMonster**
- Die Zuweisung **BösesMonster b2 = b1** kopiert also nur die in **b1** gespeicherte Referenz.
 - **b1** und **b2** verweisen auf das selbe Objekt!

- `BösesMonster b1 = new BösesMonster(50, 100, 5);`



- `BösesMonster b2 = b1;`



- Referenzen können als Parameter an Methoden übergeben werden
 - Bereits bekannt, z. B. aus der Klasse `Schatztruhe`:

```
void nimmAuf(Schatz schatz) {  
    // ...  
}
```

- Referenzen können auch ein Attribut einer Klasse sein
 - Die Klasse `Spiel` verfügt bspw. über ein Attribut `private Held held;`
- Referenzen erlauben es, Beziehungen zwischen Objekten zu realisieren

- Eine Referenz muss nicht zwingend auf ein existierendes Objekt verweisen, sondern kann auch “ins Leere” zeigen
- In diesem Fall hat die Referenz den Wert **null** (nicht 0)
- Ein Zugriff auf Attribute/Methoden einer null-Referenz führt zu einer NullPointerException:

```
Held h;
```

```
h.geheHoch(); // Fehler zur Laufzeit!
```

Aufgabe 1

- Das Spiel soll so erweitert werden, dass der Held einen weiteren Helden als Begleiter bekommt
 - Der Begleiter soll sich auf einem Feld neben dem Helden befinden
 - Er soll immer genau die gleichen Bewegungen machen wie der eigentliche Held
- Umsetzung:
 - Die Klasse **Held** muss um eine Referenz auf den Begleiter, also ein weiteres Objekt vom Typ **Held**, erweitert werden
 - Ein weiteres Objekt vom Typ **Held** muss erzeugt werden
 - Dieser neue Held muss dem schon vorhandenen als Begleiter zugewiesen werden
 - Die Methoden zum Bewegen des Helden müssen so angepasst werden, dass sich der Begleiter mitbewegt
- Zusatzaufgabe:
 - Auch der Begleiter soll noch einen Begleiter bekommen...

- Die Referenz `this` verweist immer auf das aktuelle Objekt
 - Kann benutzt werden, um Attribute von gleichnamigen Parametern zu unterscheiden:

```
class Beispiel {  
    private int x;  
    public void setX(int x) {  
        this.x = x; // x = x ginge nicht!  
    }  
}
```