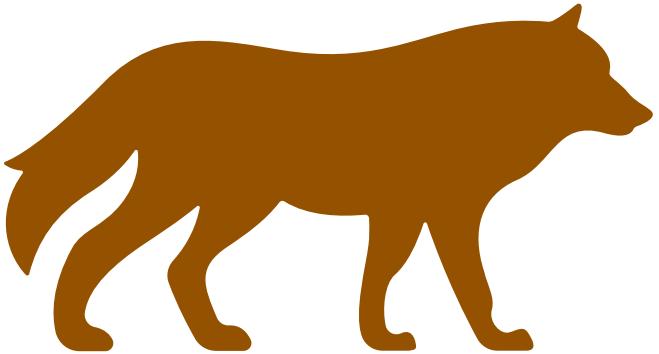
Folge 1.4
Klassen, Objekte und Methoden

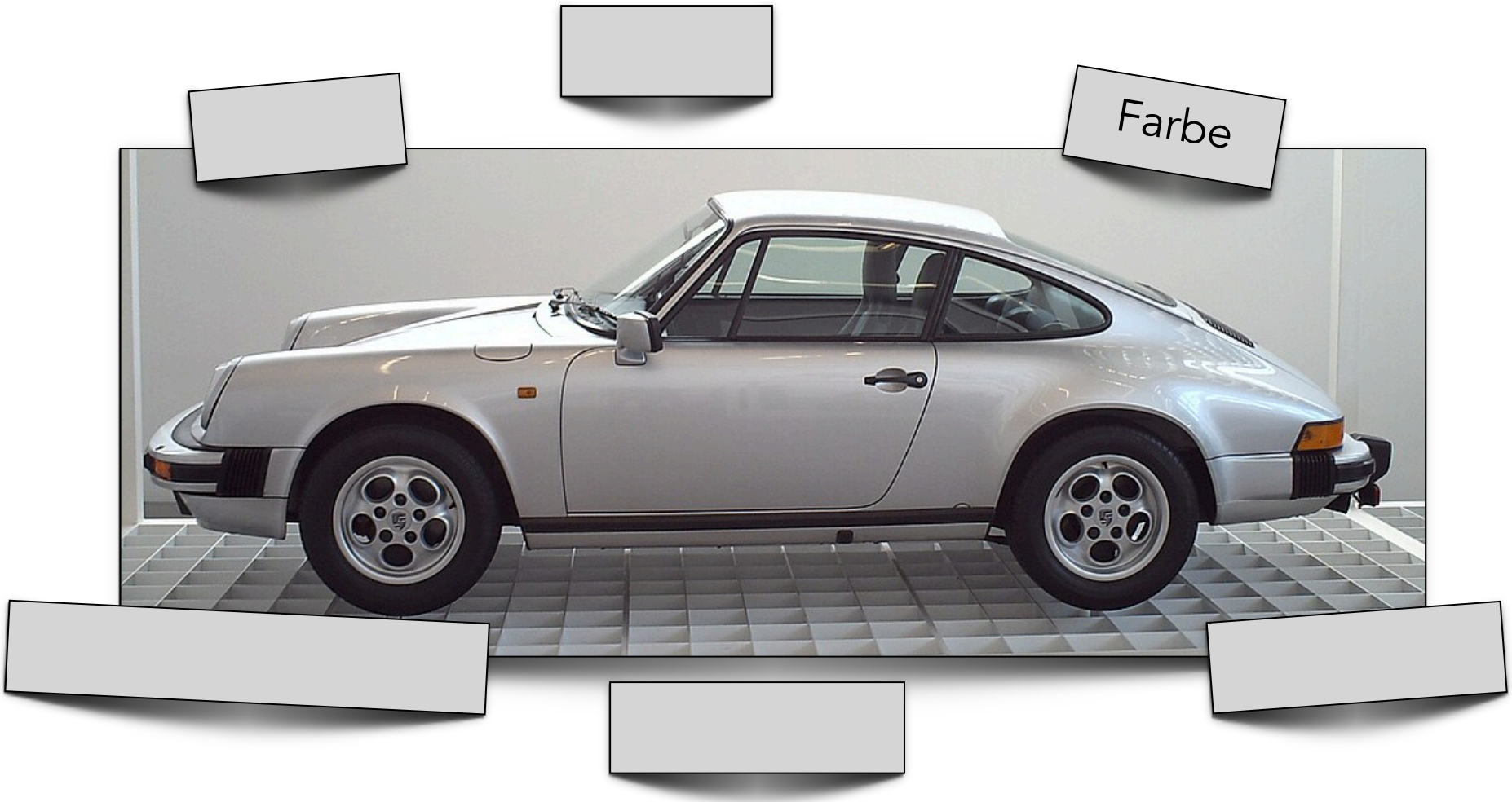
Folge 1.4
Klassen, Objekte und Methoden

1.4.1 Objekte des Alltags



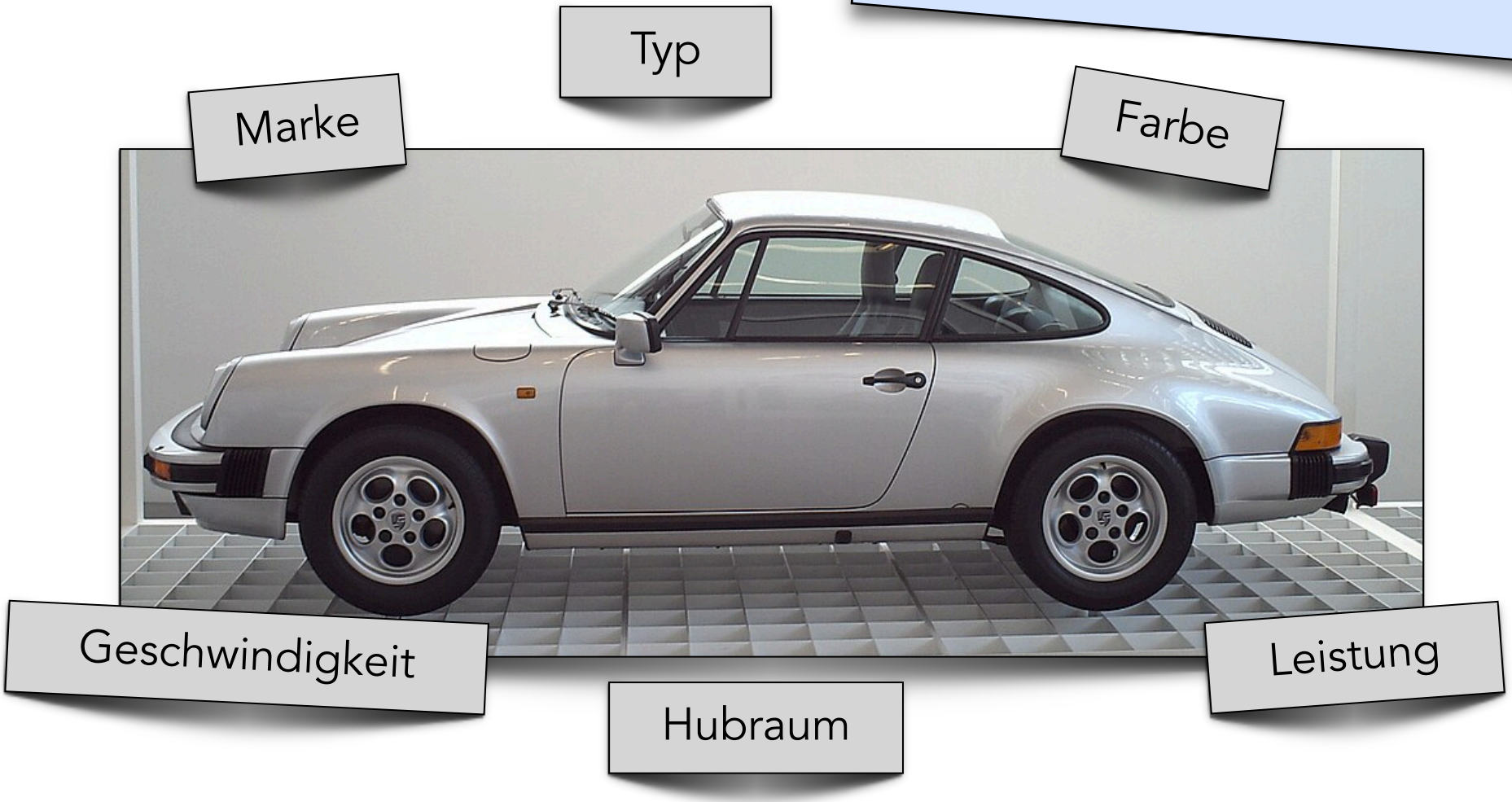
Objekte im Alltag

1.4.1 Objekte des Alltags



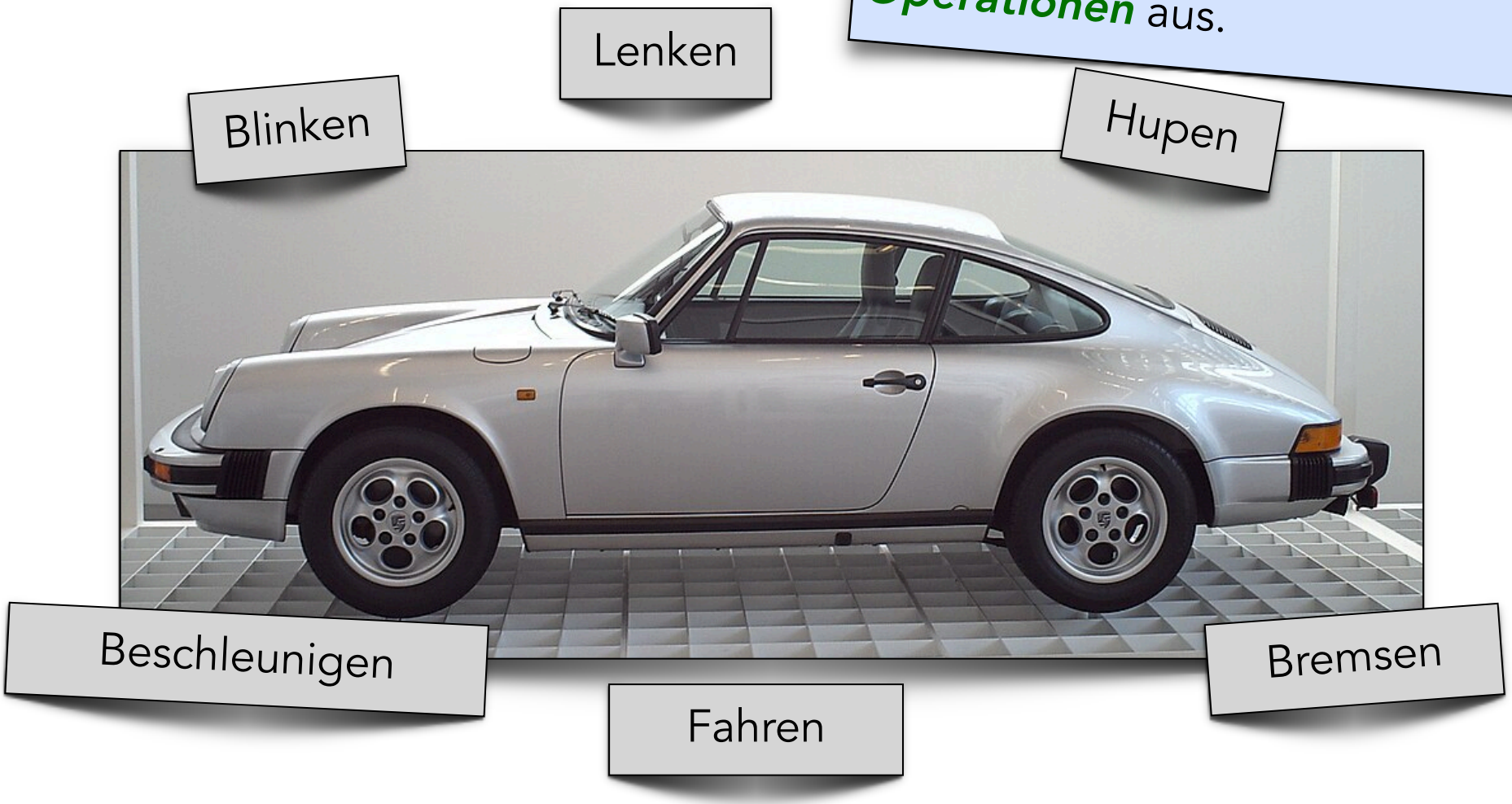
1.4.1 Objekte des Alltags

Jedes Objekt hat Eigenschaften bzw. ***Attribute***

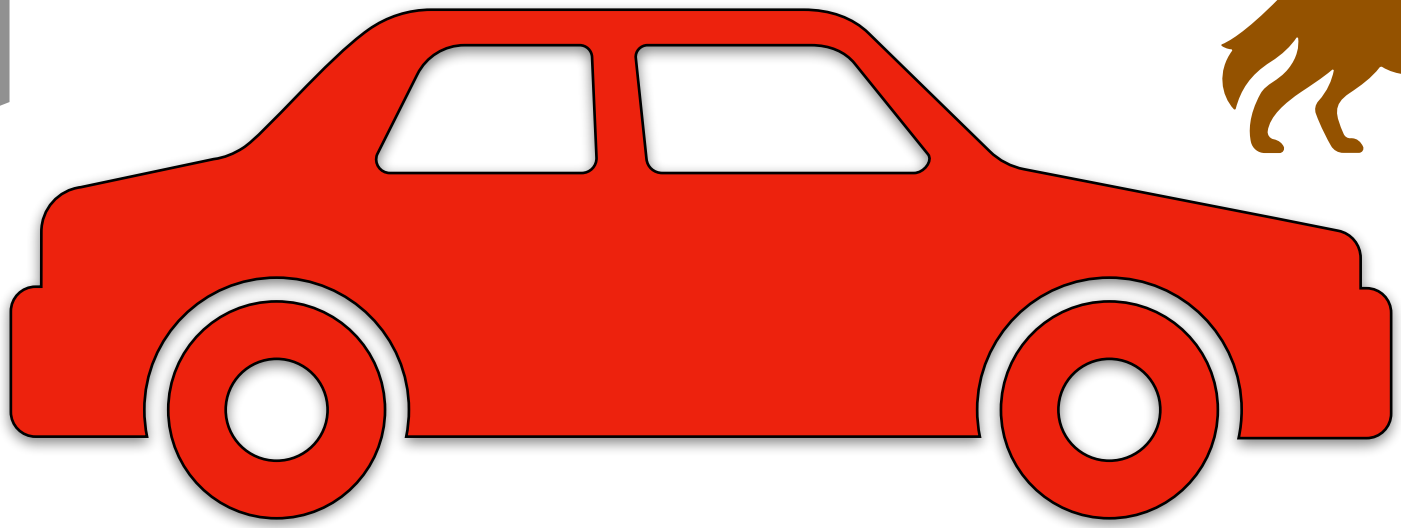


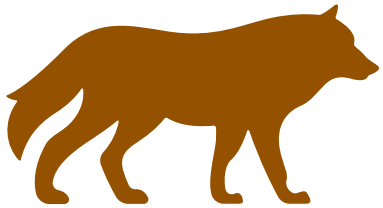
1.4.1 Objekte des Alltags

Jedes Objekt zeichnet sich durch Fähigkeiten bzw. ***Aktionen / Operationen*** aus.



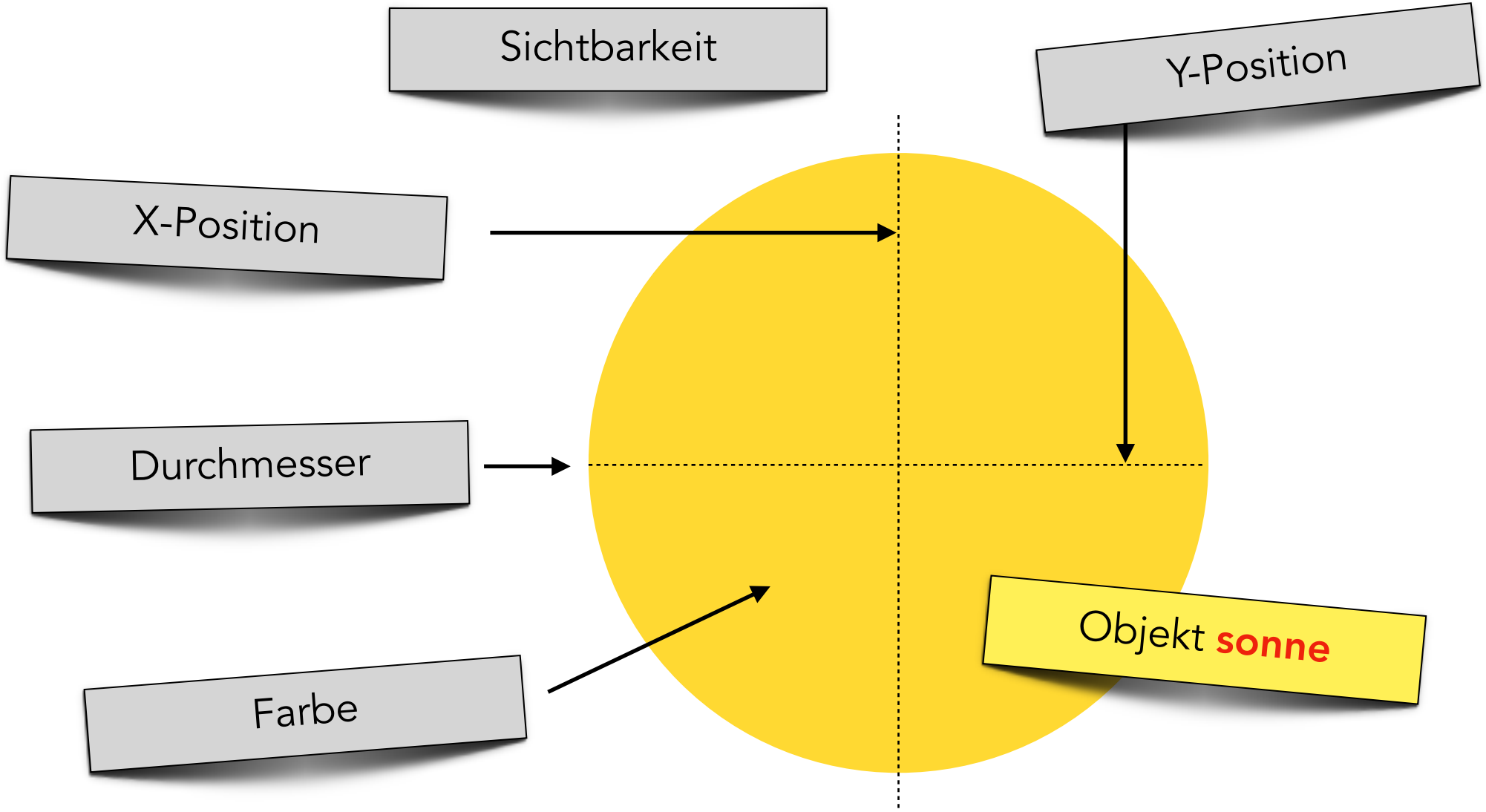
1.4.1 Objekte des Alltags



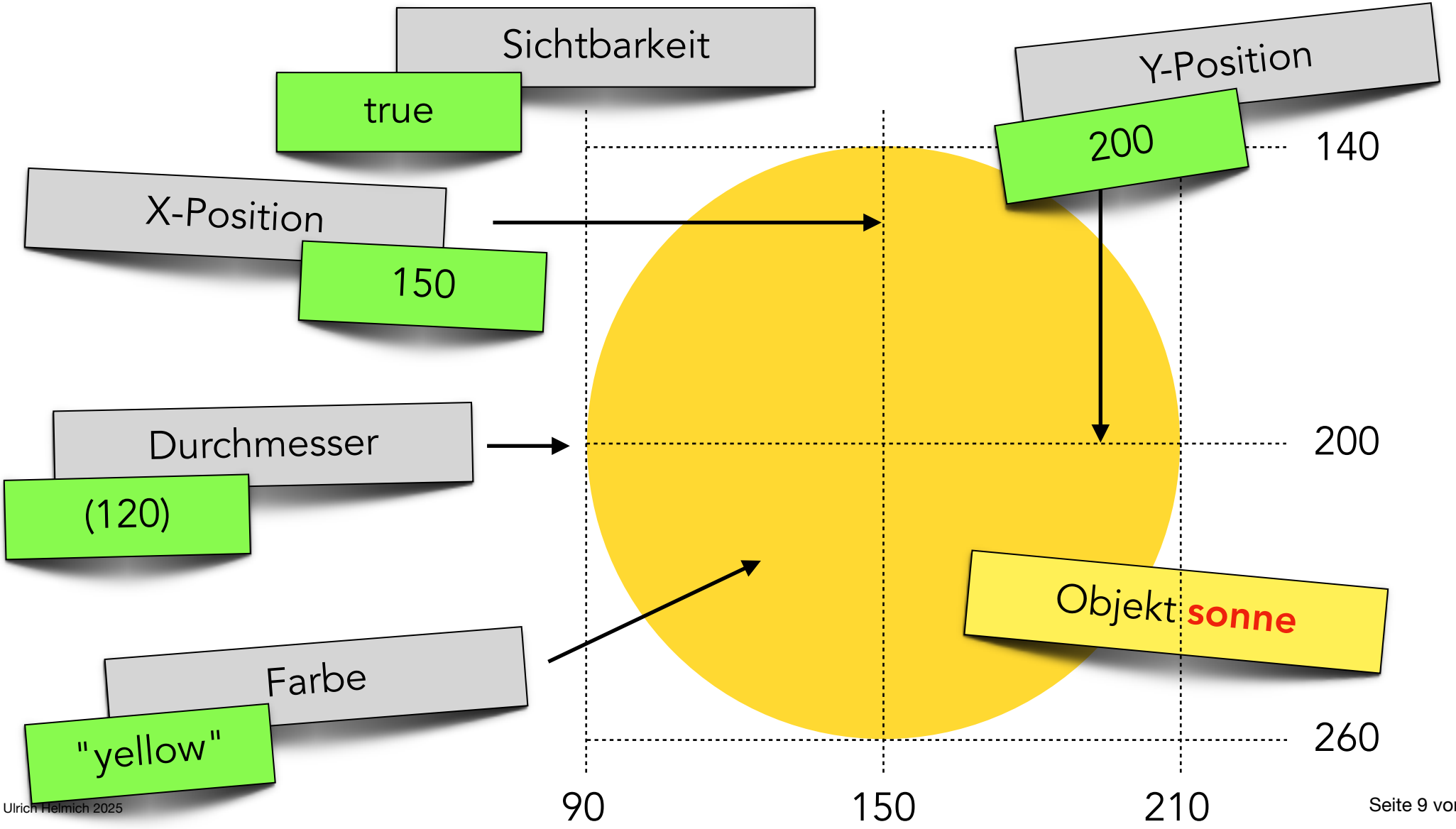


Konkrete und abstrakte Objekte des Alltags zeichnen sich durch bestimmte ***Eigenschaften*** oder ***Attribute*** aus und sie haben bestimmte ***Fähigkeiten*** bzw. man kann ***Operationen*** mit ihnen durchführen.

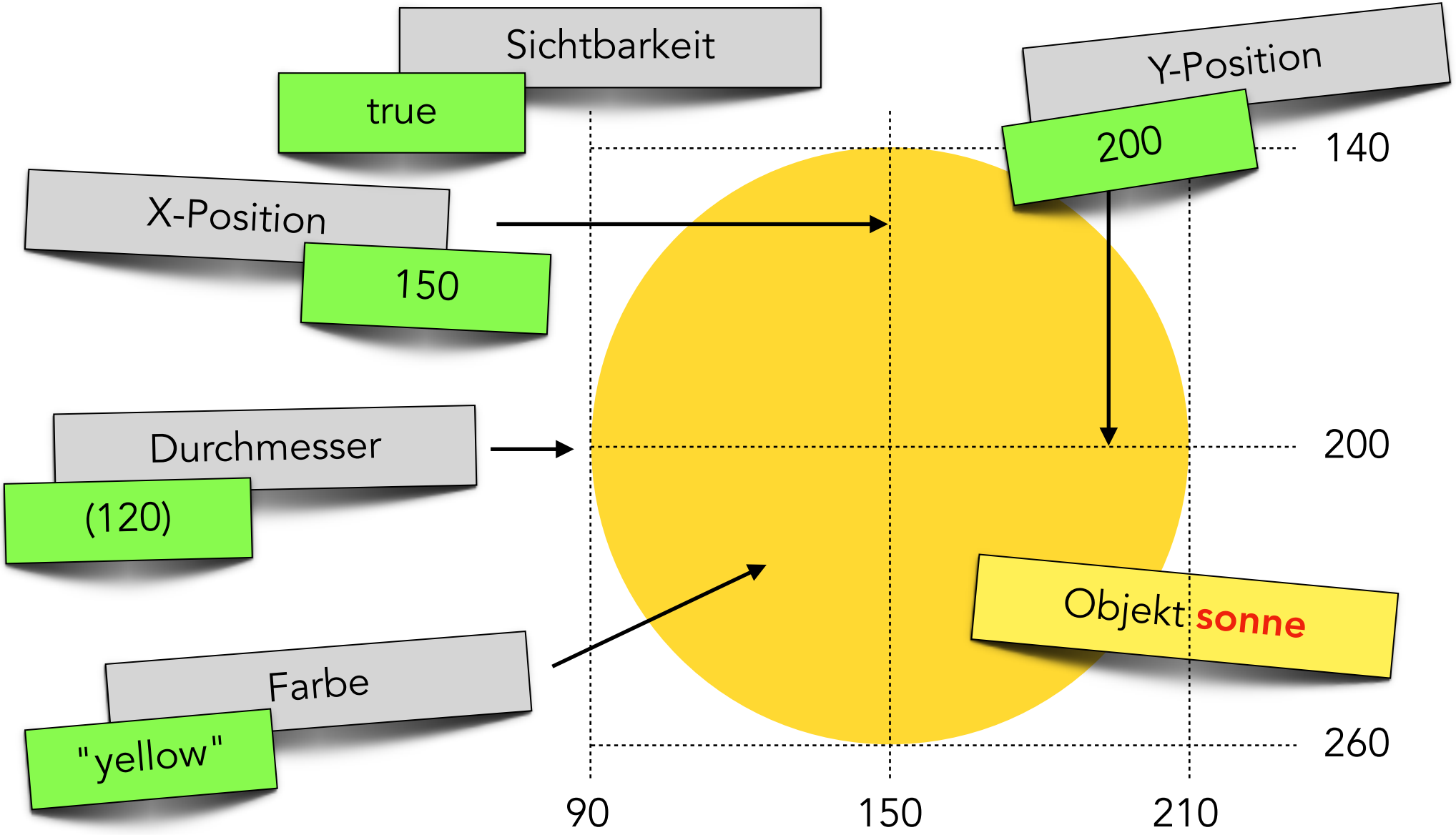
1.4.2 Objekte in Java: Attribute und Instanzvariablen



1.4.2 Objekte in Java: Attribute und Instanzvariablen



1.4.2 Objekte in Java: Attribute und Instanzvariablen



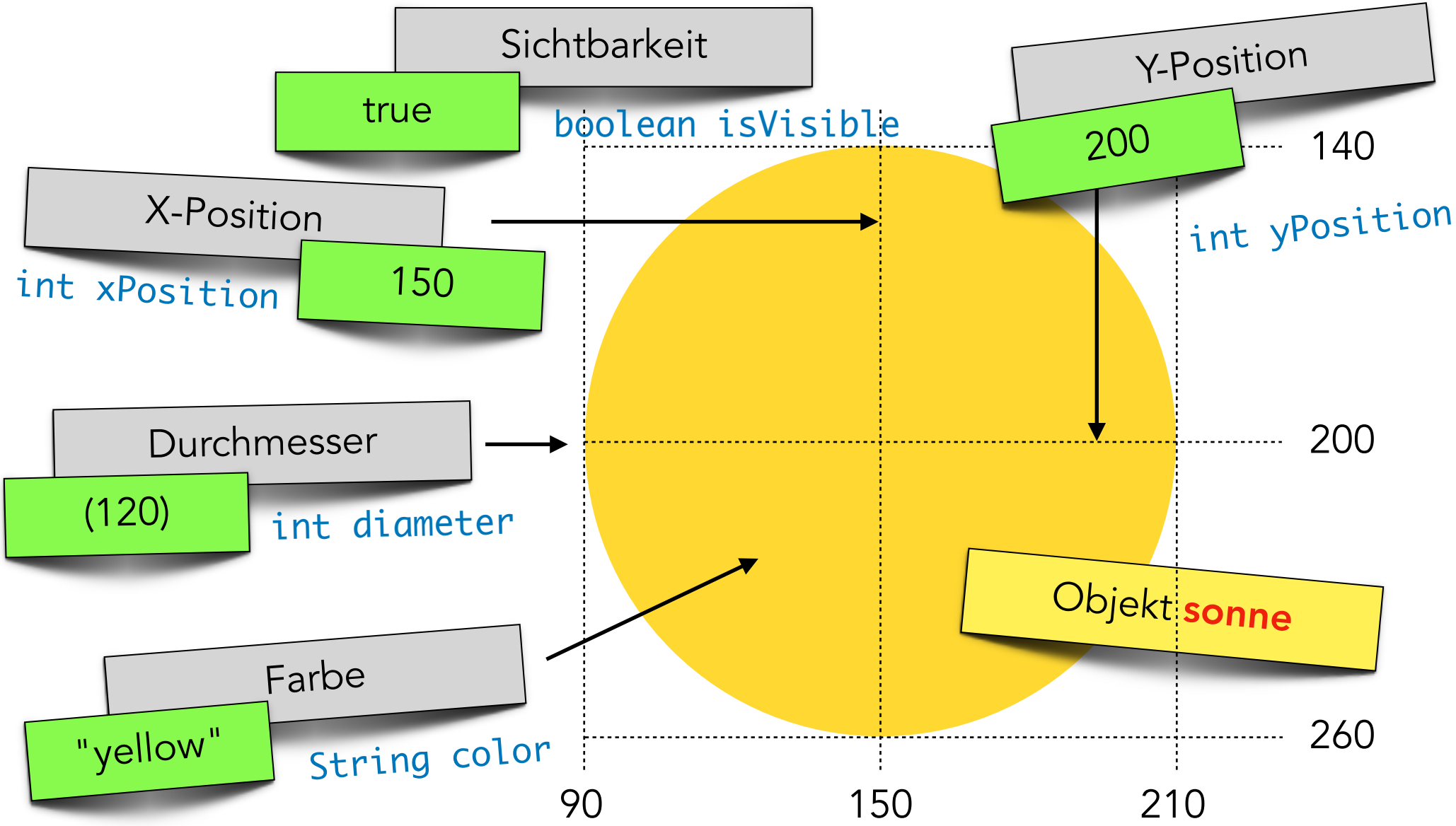
1.4.2 Objekte in Java: Attribute und Instanzvariablen

Attribut	**Instanzvariable**
X-Position	`int xPosition`
Y-Position	`int yPosition`
Durchmesser	`int diameter`
Farbe	`String color`
Sichtbarkeit	`boolean isVisible`

*Attribute und Instanzvariablen der Klasse **Circle***

Die Attribute von Java-Objekten werden durch ***Instanzvariablen*** realisiert.

1.4.2 Objekte in Java: Attribute und Instanzvariablen



1.4.2 Objekte in Java: Attribute und Instanzvariablen

```
11 public class Circle
12 {
13     private int diameter;
14     private int xPosition;
15     private int yPosition;
16     private String color;
17     private boolean isVisible;
18
19     /**
20      * Create a new circle at default position with default color.
21      */
22     public Circle()
23     {
24         diameter = 30;
25         xPosition = 20;
26         yPosition = 60;
27         color = "blue";
28         isVisible = false;
29     }
30
```

Die ***Instanzvariablen*** der Klasse **Circle**.

Die Werte der Instanzvariablen.

1.4.2 Objekte in Java: Attribute und Instanzvariablen

```
11 public class Circle
12 {
13     private int diameter;
14     private int xPosition;
15     private int yPosition;
16     private String color;
17     private boolean isVisible;
18
19     /**
20      * Create a new circle at default position with default color.
21      */
22     public Circle()
23     {
24         diameter = 30;
25         xPosition = 20;
26         yPosition = 60;
27         color = "blue";
28         isVisible = false;
29     }
30
```

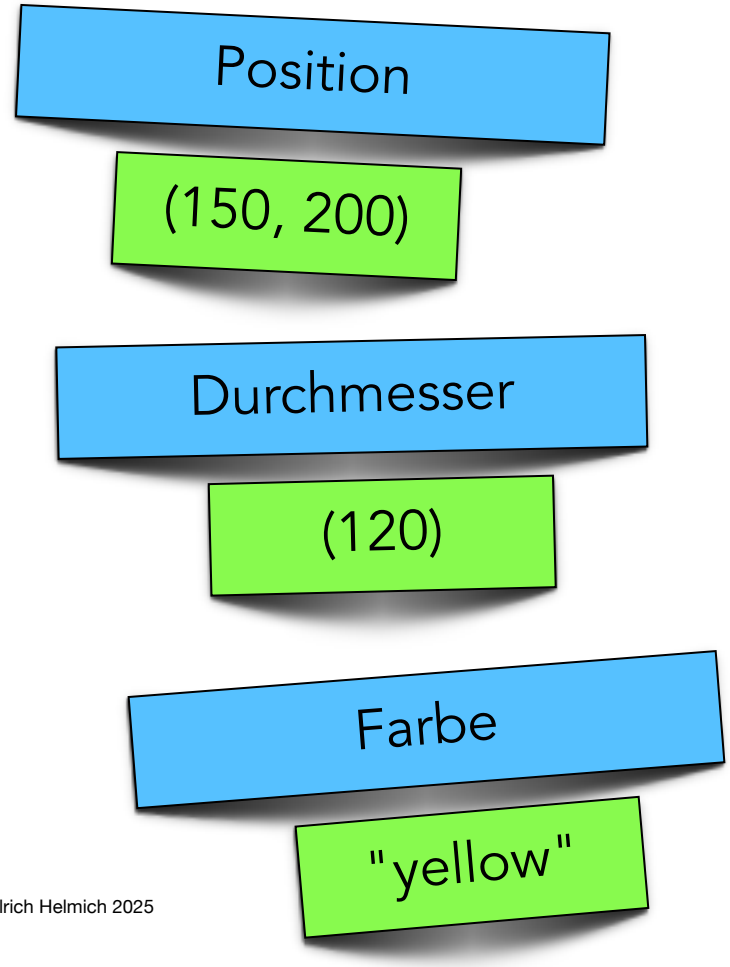
Die ***Instanzvariablen*** der Klasse **Circle**.

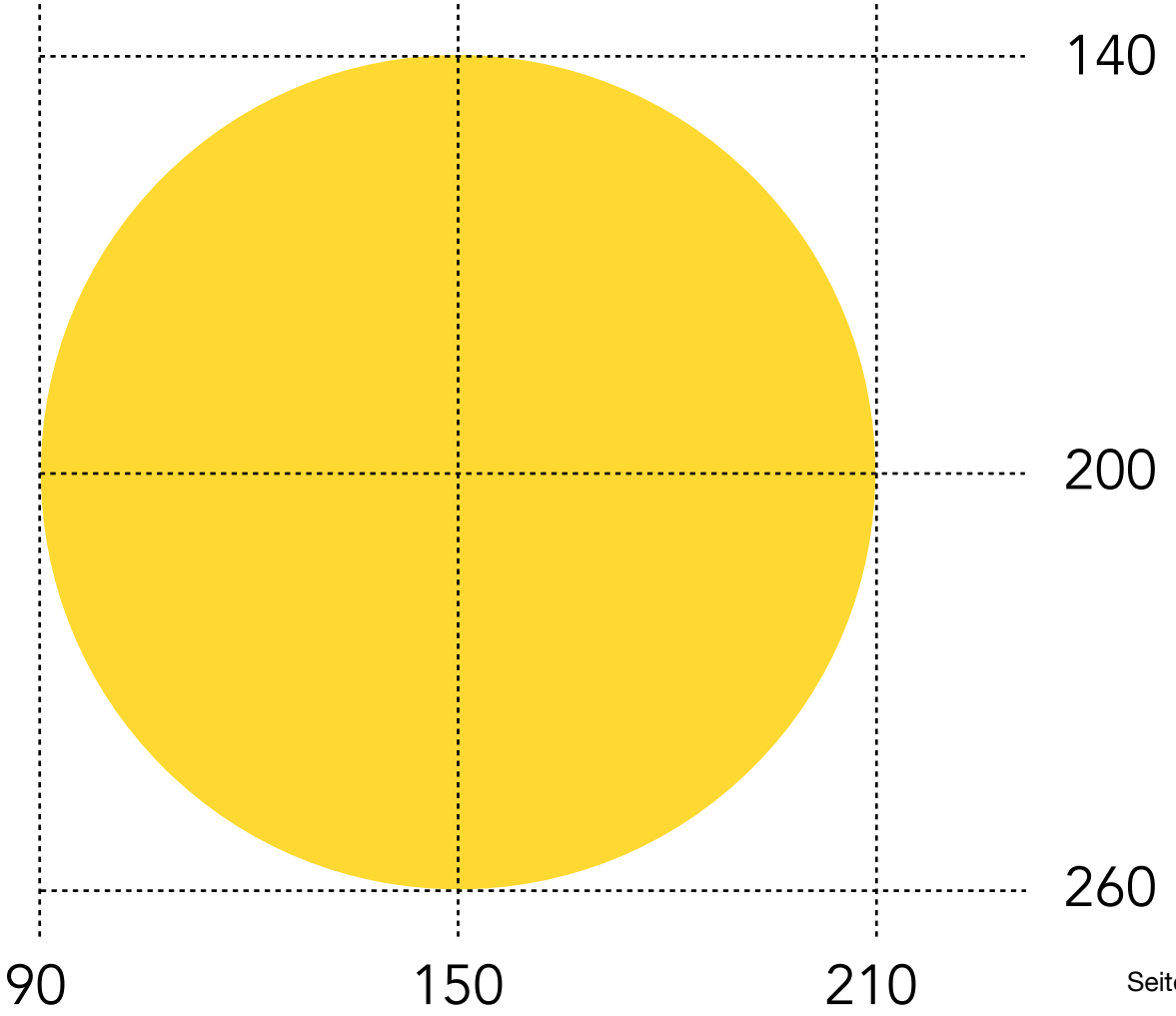
Die Werte der Instanzvariablen.

Deklaration der Instanzvariablen im Kopf der Klasse.

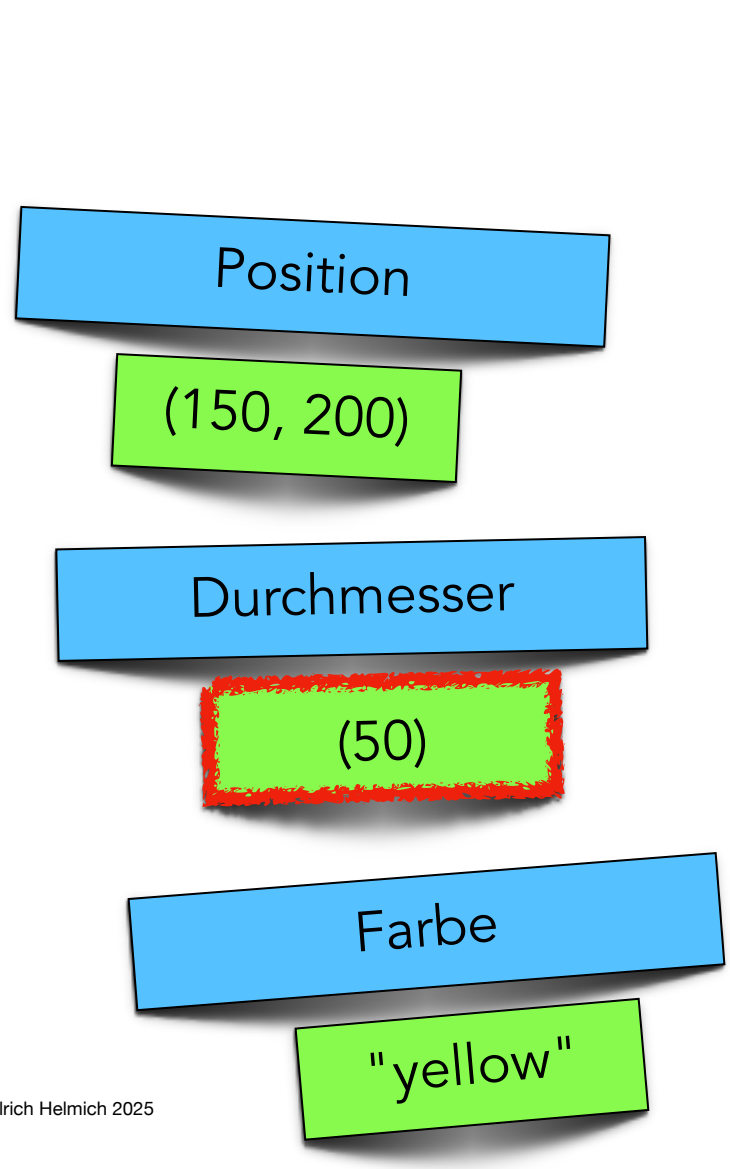
Initialisierung der Instanzvariablen im Konstruktor der Klasse.

1.4.3 Objekte in Java: Methoden



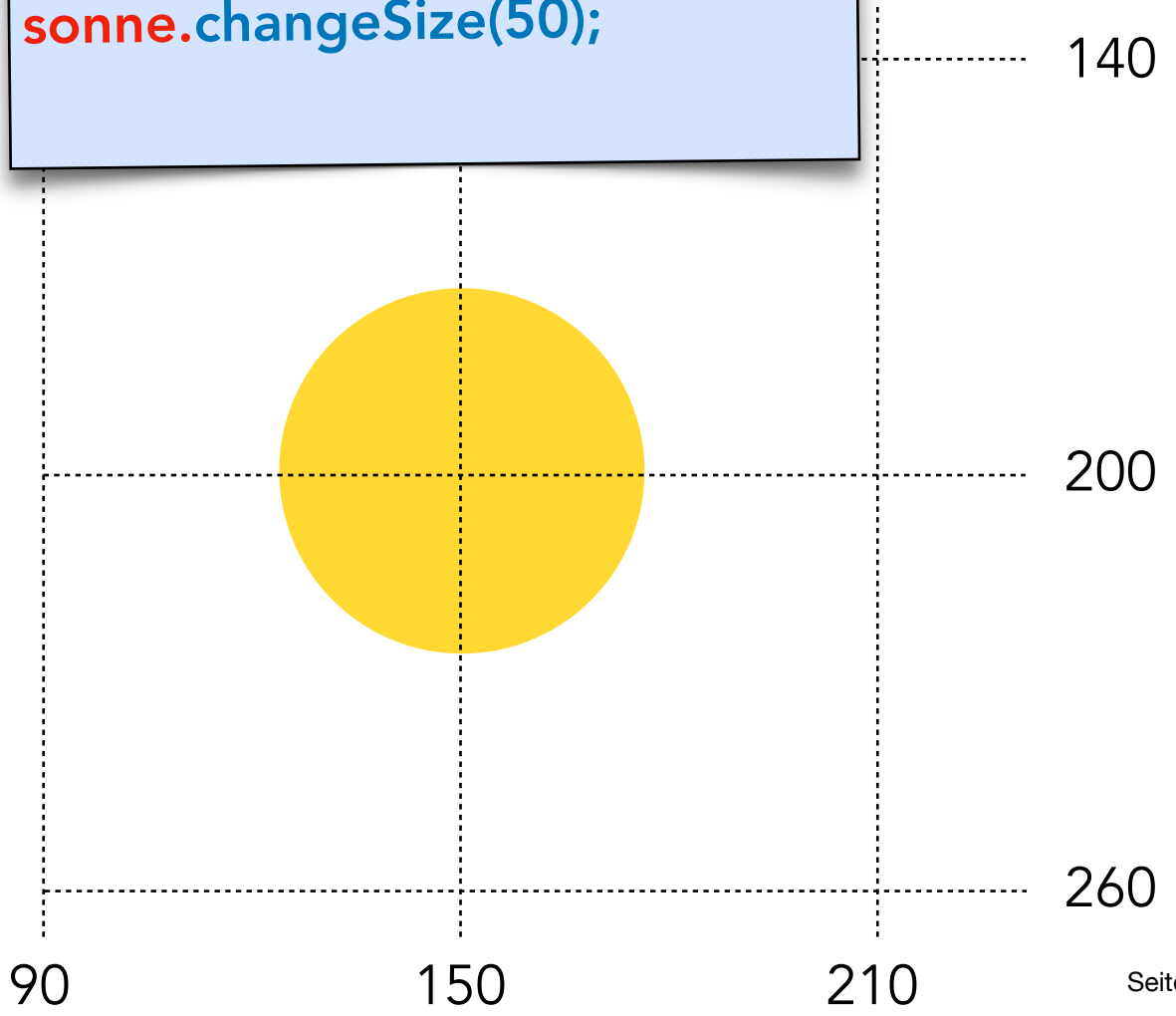


1.4.3 Objekte in Java: Methoden

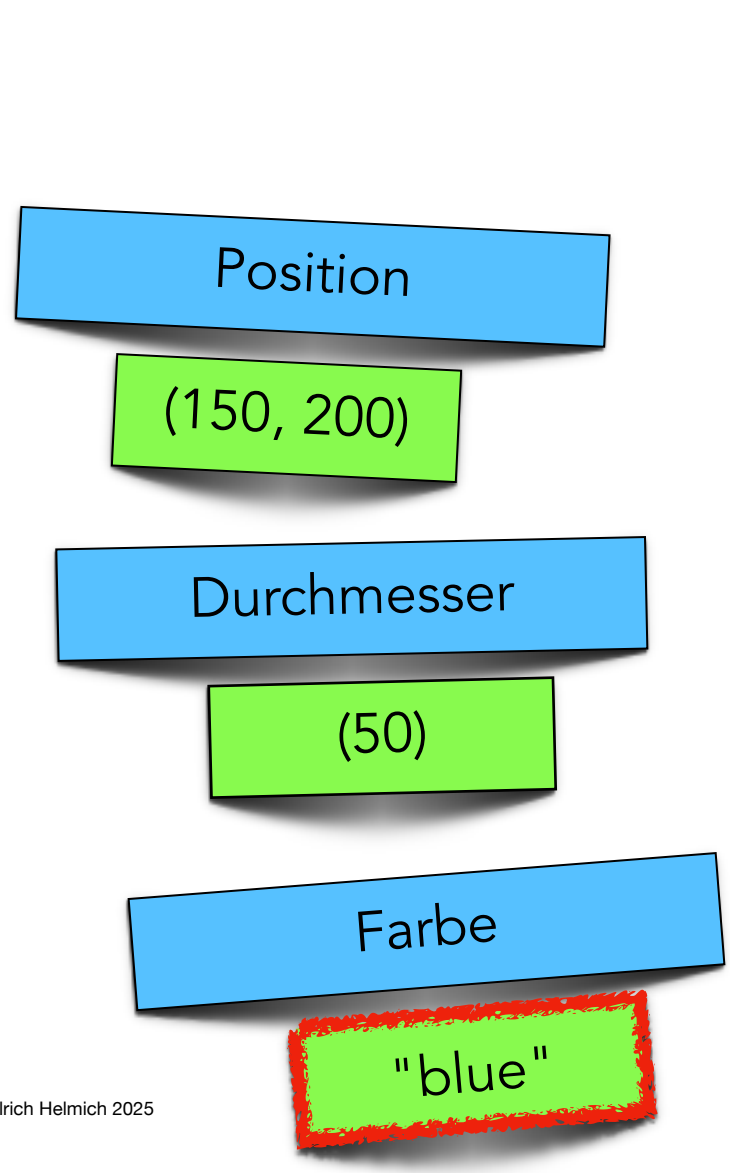


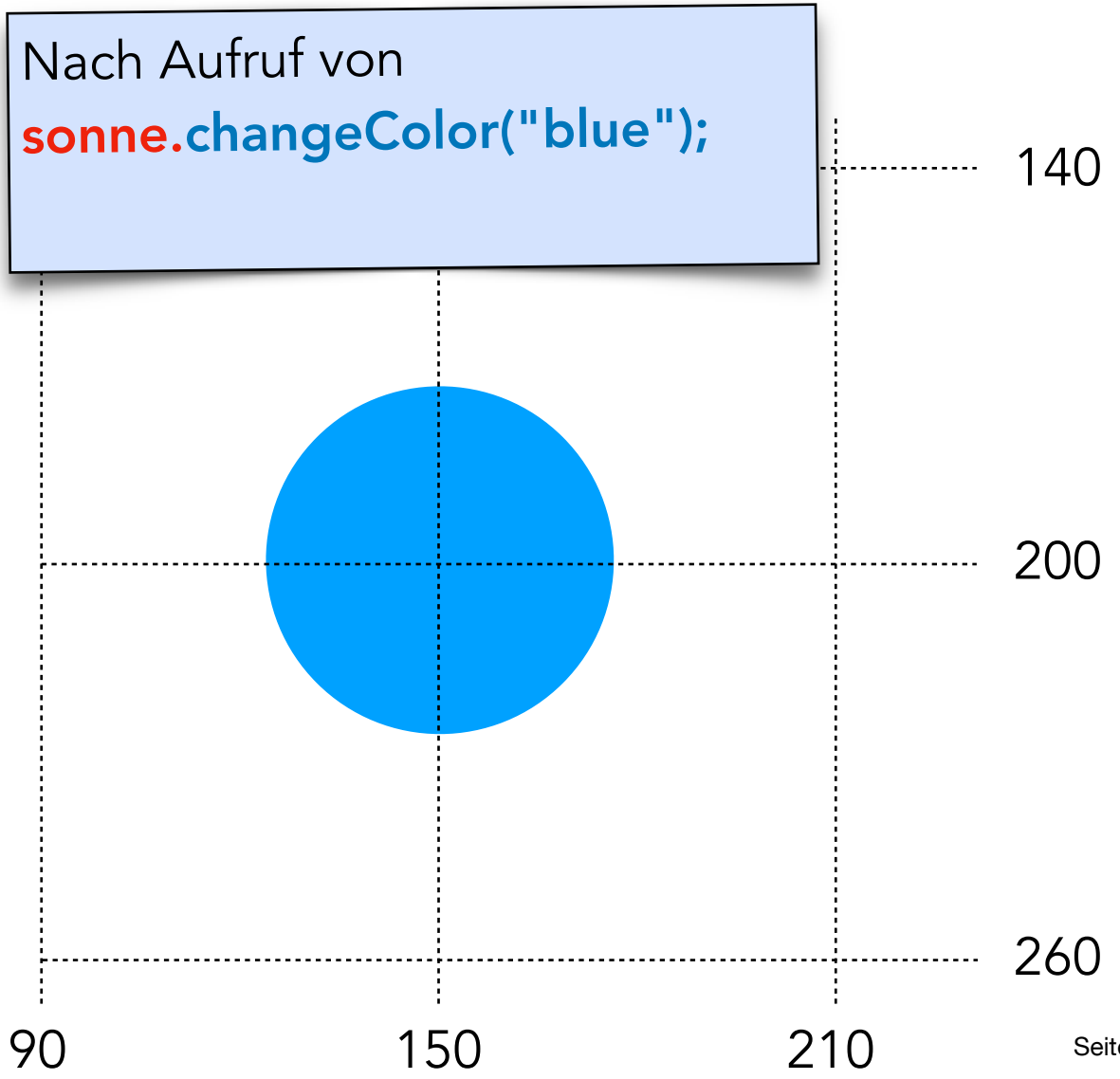
Nach Aufruf von

```
sonne.changeSize(50);
```

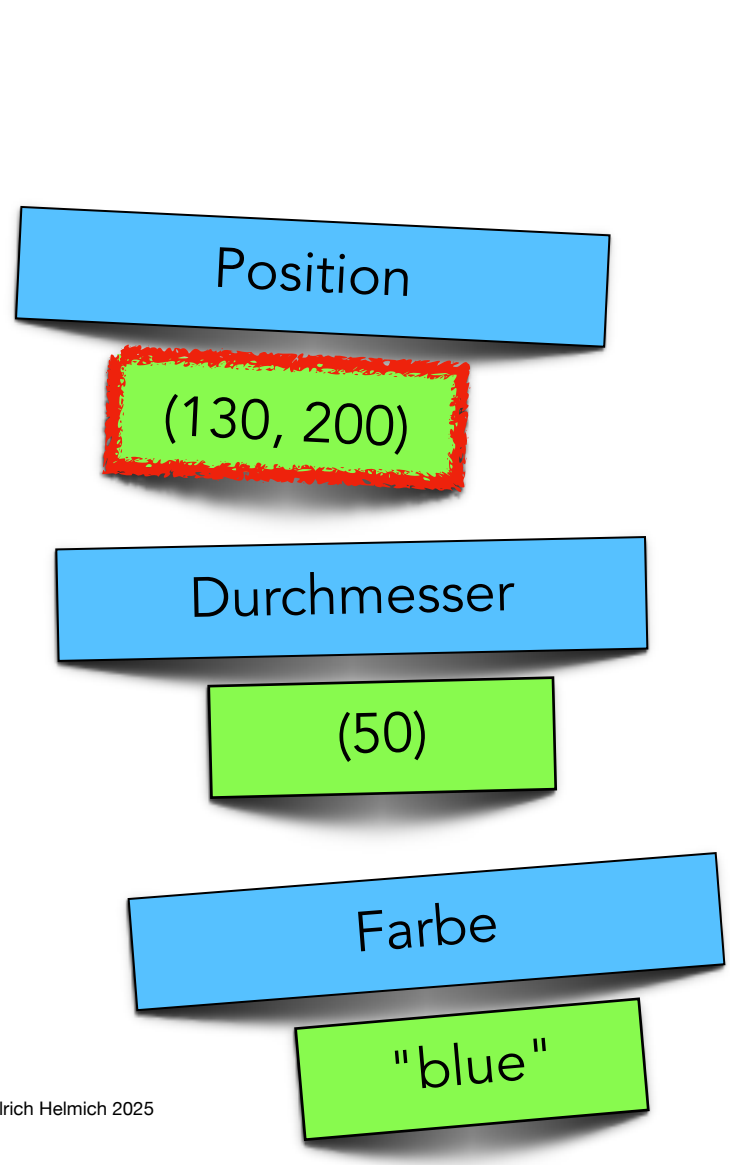


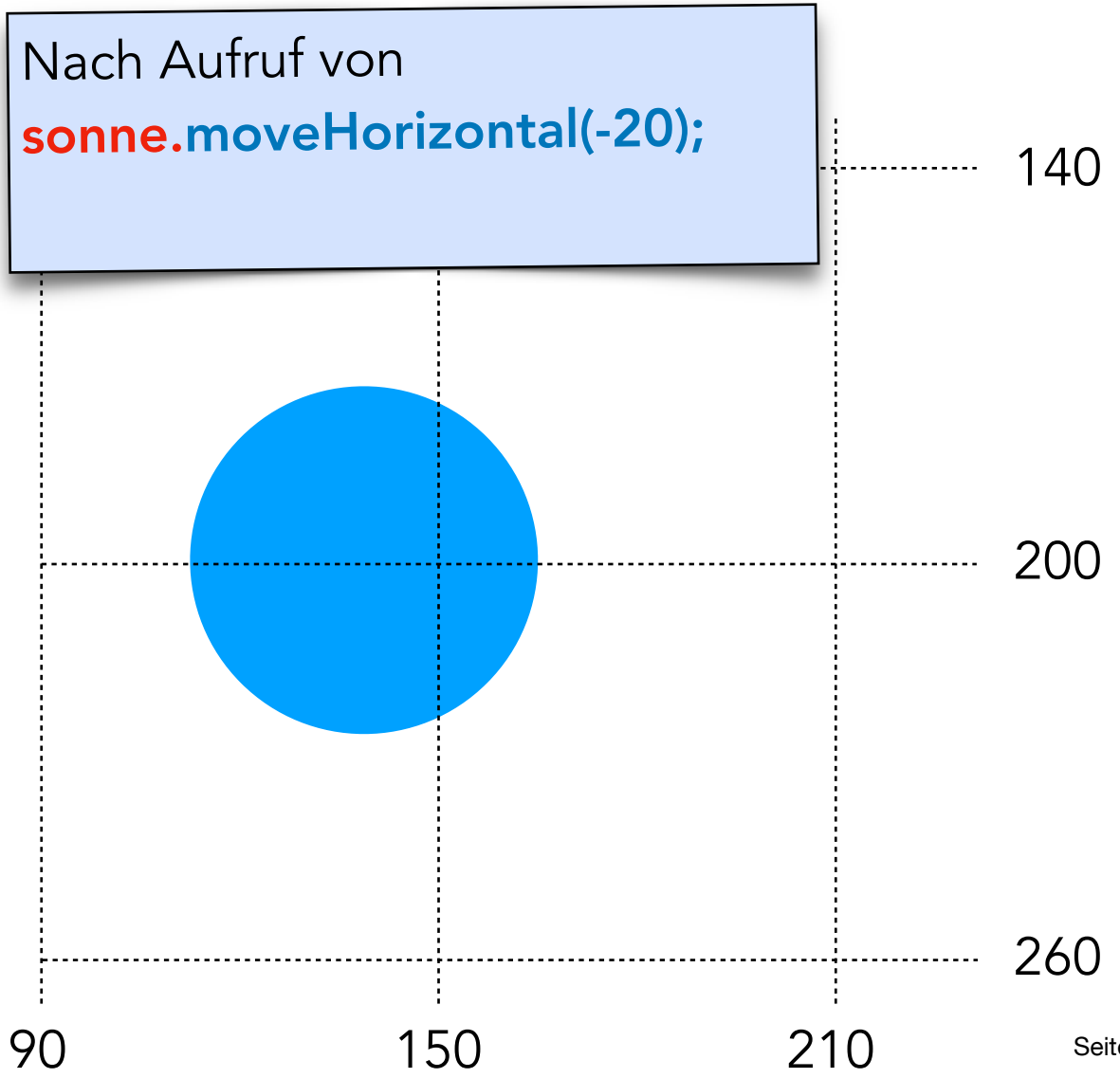
1.4.3 Objekte in Java: Methoden





1.4.3 Objekte in Java: Methoden





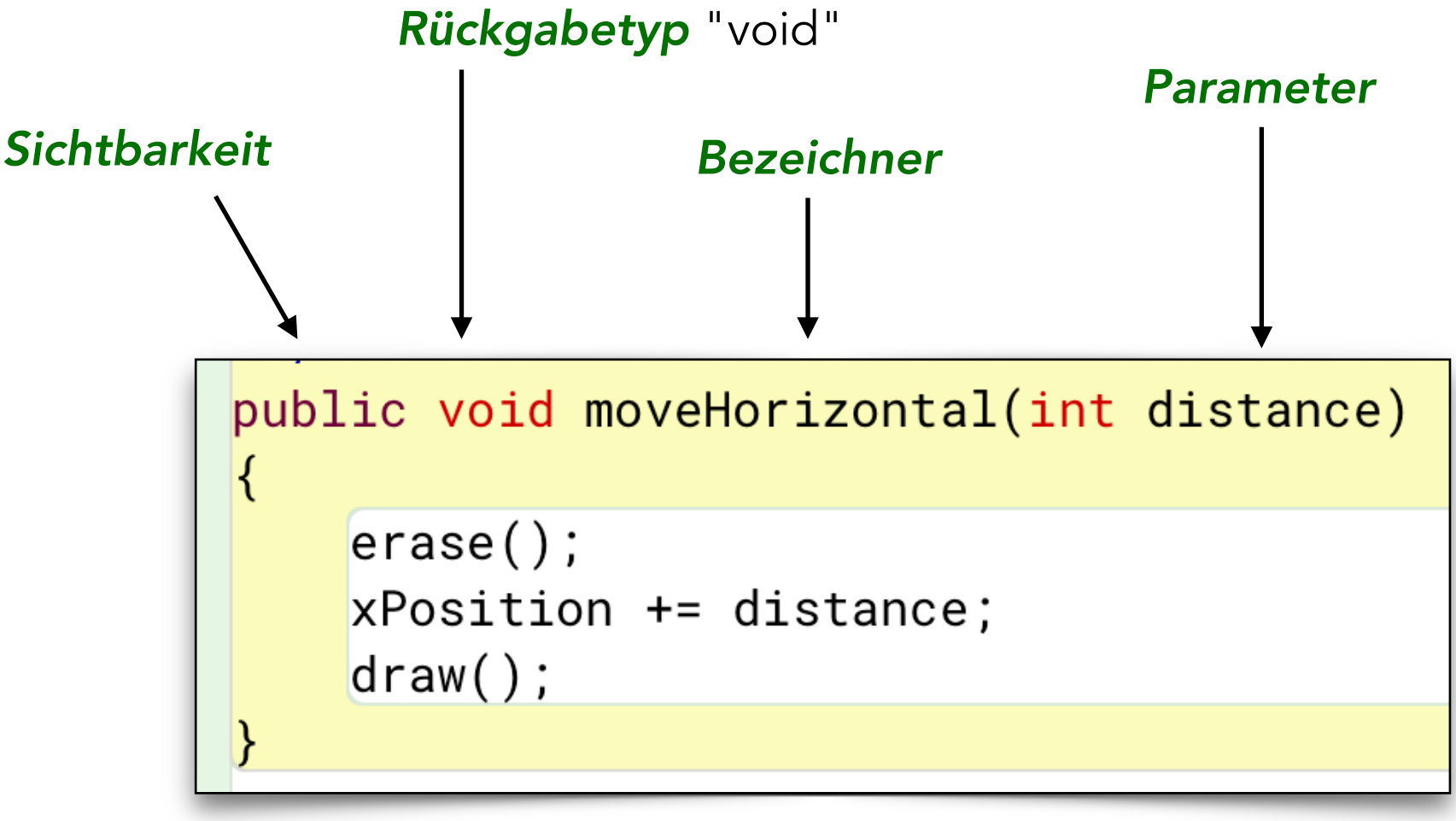
1.4.3 Objekte in Java: Methoden

```
/**
 * Move the circle horizontally by 'distance' pixels.
 */
public void moveHorizontal(int distance)
{
    erase();
    xPosition += distance;
    draw();
}

/**
 * Move the circle vertically by 'distance' pixels.
 */
public void moveVertical(int distance)
{
    erase();
    yPosition += distance;
    draw();
}
```

Zwei Methoden der Klasse **Circle**.

1.4.3 Objekte in Java: Methoden



1.4.3 Objekte in Java: Methoden

Löschen des Kreises

Wert der Instanzvariable `xPosition` wird um den Wert des Parameters `distance` erhöht

```
public void moveHorizontal(int distance)
{
    erase();
    xPosition += distance;
    draw();
}
```

Zeichnen des Kreises an neuer Position

1.4.3 Objekte in Java: Methoden

```
public void moveHorizontal(int distance)
{
    erase();
    xPosition += distance;
    draw();
}
```

Manipulierende Methoden

verändern die Werte von Instanzvariablen

1.4.3 Objekte in Java: Methoden

Parameter
informieren Methoden darüber, wie die Instanzvariablen verändert werden sollen.

```
public void moveHorizontal(int distance)
{
    erase();
    xPosition += distance;
    draw();
}
```

Manipulierende Methoden
verändern die Werte von Instanzvariablen

1.4.3 Objekte in Java: Methoden

```
public void moveHorizontal(int distance)
{
    erase();
    xPosition += distance;
    draw();
}
```

Aufgabe 1.4.3 #1

Ergänzen Sie die Klasse Circle um eine Methode

public void moveDiagonal(int xDistance, int yDistance)

welche den Kreis gleichzeitig in horizontaler und vertikaler Richtung bewegen kann. Testen Sie die Methode dann.

1.4.3 Objekte in Java: Methoden

```
public int getYPosition()
{
    return yPosition;
}
```

Sondierende Methoden

liefern die Werte von Instanzvariablen zurück

1.4.3 Objekte in Java: Methoden

Aufgabe 1.4.3 #2

Ergänzen Sie die Klasse **Circle** um eine Getter-Methode, die den Umfang des Kreises zurückliefert.
Die Formel zur Berechnung des Umfangs kennen Sie sicherlich noch aus dem Schulunterricht.

Und wenn Sie schon mal dabei sind: Schreiben Sie auch noch eine Getter-Methode zur Berechnung des Flächeninhaltes.

1.4.3 Objekte in Java: Methoden

Aufgabe 1.4.3 #3 (Experten)

Umfang und Fläche sind Attribute eines jeden Kreises. Allerdings werden in der Klasse **Circle** diese Attribute nicht durch Instanzvariablen repräsentiert, sondern müssen aus dem Durchmesser (diameter) berechnet werden.

Schreiben Sie jetzt eine Setter-Methode, die den neuen Flächeninhalt des Kreises als Parameter übernimmt und dann den Durchmesser des Kreises entsprechend anpasst.

1.4.3 Objekte in Java: Methoden

Getter-Methoden

Sondierende Methoden, die den Wert von Attributen zurückliefern.

Namens-Konvention

Präfix "get" + Name des Attributs mit Großbuchstaben

```
public int getYPosition()
{
    return yPosition;
}
```

1.4.3 Objekte in Java: Methoden

Setter-Methoden

Manipulierende Methoden, die den Wert von Instanzvariablen ändern.

```
public void setYPosition(int y)
{
    yPosition = y;
}
```

```
public void setPosition(int x, int y)
{
    xPosition = x;
    yPosition = y;
}
```

1.4.4 Aufgaben

```
1  public class Zeichnung1
2  {
3      public Zeichnung1()
4      {
5          zeichneKreisMitRand(70,"yellow");
6      }
7
8      public void zeichneKreis(int durchmesser, String farbe)
9      {
10         Circle kreis = new Circle();
11         kreis.makeVisible();
12         kreis.changeSize(durchmesser);
13         kreis.changeColor(farbe);
14     }
15
16     public void zeichneKreisMitRand(int durchmesser, String farbe)
17     {
18         zeichneKreis(durchmesser+4,"black");
19         zeichneKreis(durchmesser,farbe);
20     }
21 }
```

Aufgabe 1.4.4#1

1. Was wird hier versucht?
2. Was funktioniert hier, und was nicht?
3. Was könnte man machen, damit es besser funktioniert?

1.4.4 Aufgaben

Aufgabe 1.4.4#2

1. Ergänzen Sie die Klasse **Circle** um eine Methode, mit der die Position des Kreises neu bestimmt werden kann:

   ```
   public void setPosition(int x, int y)
   ```

2. Wenden Sie diese neue Methode auf **zeichneKreis()** und **zeichneKreisMitRand()** an, um diese zu erweitern und gleichzeitig zu vereinfachen.

1.4.4 Aufgaben

Aufgabe 1.4.4#3 (Experten)

1. Ergänzen Sie auch die Klassen **Square**, **Rect** und **Triangle** um eine entsprechende Methode

   ```
   public void setPosition(int x, int y)
   ```

2. Erweitern Sie die Klasse **Zeichnung** um Quadrate, Rechtecke und Dreiecke mit Rand.

1.4.4 Aufgaben

Aufgabe 1.4.4#4 (Experten)

Machen Sie die Methoden zum Zeichnen der Graphiken mit Rand flexibler:

1. Die **Randstärke** soll durch einen Parameter festgelegt werden können.
2. Die **Randfarbe** soll durch einen weiteren Parameter bestimmt werden können.