

Wiederholung

- Objekt:
 - “Ding” mit Attributen (Eigenschaften) und Methoden (Fähigkeiten)
 - Die Methoden arbeiten “auf dem Objekt”, also mit dessen Attributen
- Klasse:
 - “Bauplan” für einen bestimmten Typ von Objekten

Von einer Klasse kann es beliebig viele Objekte (aka Instanzen der Klasse) geben.

Objektorientierte Programmierung in Java

- Klasse **Kreis**
 - Attribute: **x**, **y** (Mittelpunkt), **r** (Radius)
 - Methoden: **setzeRadius()**, **berechneUmfang()**,
berechneFlaeche()

- Erzeuge weitere Objekte der Klasse **Kreis** und experimentiere mit den vorhandenen Methoden:
 - Weise den Objekten jeweils einen unterschiedlichen Radius zu
 - Lasse Fläche und Umfang berechnen
- Erzeuge eine zweite Klasse **Rechteck**
 - Das Rechteck soll die Position einer Ecke sowie die Breite und Höhe speichern
 - Ergänze Methoden zum Setzen dieser Eigenschaften sowie zur Berechnung von Umfang und Fläche
 - Teste auch diese Methoden mit mehreren Objekten

- Wir konnten den Attributen in BlueJ nicht direkt Werte zuweisen
 - Umweg über eine entsprechende Methode notwendig
- Dies ist ein gewolltes Verhalten, das auch abseits von BlueJ Anwendung findet
 - Man bezeichnet dies als **Kapselung**

Kapselung

- Attribute eines Objekts sollen nur über entsprechende Methoden des Objekts manipuliert werden dürfen

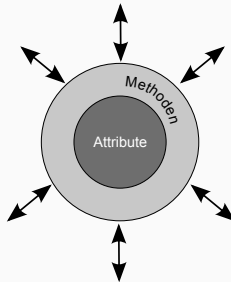


Abbildung 1: Symbolische Darstellung der Kapselung

- Solche Methoden nennt man **Setter** (zum Setzen von Werten) und **Getter** (zum Auslesen von Werten)

- Java erlaubt das Festlegen von Zugriffsrechten zur Umsetzung der Kapselung:

public Auf das Attribut/die Methode darf von überall aus zugegriffen werden

private Zugriff auf das Attribut/die Methode ist nur von innerhalb der Klasse aus möglich

- In der Regel gilt:
 - Attribute sind **private**
 - Methoden sind **public**

```
class Kreis {  
    private double radius;  
  
    public void setRadius(int r) { // Setter  
        radius = r;  
    }  
  
    public double getRadius() { // Getter  
        return(radius);  
    }  
}
```

Hinweis zur Benennung: Üblich ist das Präfix **set** für Setter und **get** für Getter. Auf Deutsch wird manchmal auch **setze** und **gib** verwendet.

- Wozu der Aufwand?
 - Interna der Klasse können dem “Verwender” egal sein (Geheimnisprinzip)
 - Änderung der Interna möglich, ohne dass benutzender Code “kaputt geht” (solange die Setter und Getter gleich bleiben)
 - Konsistenzprüfung (bspw. “Radius muss größer Null sein”)
 - Aktualisierung abhängiger Attribute möglich (bspw. “Änderung des Radius aktualisiert auch den Umfang”)

Aufgabe 2

- Öffne das Projekt **Pong-Template** mit BlueJ
- Vervollständige die Klasse **Ball**
 - Ergänze fehlende Setter und Getter
 - Ermittle Breite und Höhe des Spielfelds anhand der Methode **bewege()**
 - Vervollständige die Methoden **abprallenX()** und **abprallenY()**
- Analysiere den Code in der Klasse **Spielfeld**
 - Finde die Codeteile, die für die Erzeugung des Objekts **ball1** verantwortlich sind
 - Finde heraus, wie eine Methode für ein bestimmtes Objekt aufgerufen wird
 - Finde heraus, wo die Methode **bewege()** des Objekts **ball1** aufgerufen werden muss, damit sich der Ball fortlaufend bewegt
 - Erzeuge mindestens einen weiteren Ball, der sich ebenfalls bewegt