



JADE HOCHSCHULE

Wilhelmshaven Oldenburg Elsfleth

Informatik 2



JADE HOCHSCHULE

Wilhelmshaven Oldenburg Elsfleth

Einführung in die objektorientierte Programmierung

Ziel:

Fachausdrücke und Konzepte der objektorientierten Programmierung verstehen

- Attribute in Java deklarieren können
- Methoden in Java deklarieren können
- Einfache Programmieraufgaben durch Aufrufen von Methoden lösen können

Gliederung

- Objekte und Klassen als Modelle der realen Welt
- Programm-Objekte als vereinfachte Modelle realer Objekte
- Lösen von Programmieraufgaben mit Hilfe von Programm-Objekten

Programme sind Modelle eines Ausschnitts der realen Welt

- Programme modellieren Teile der realen Welt
- Die reale Welt lässt sich näherungsweise durch Objekte beschreiben
- Programme enthalten vereinfachte Modelle der realen Objekte
- Programm-Objekte sind vereinfachte Modelle der realen Objekte

Beispiel: Straßenverkehr

Realität



Mögliche Ausschnitte der Realität:

- Verkehrsteilnehmer
- Geschwindigkeitsüberschreitungen
- Grünanlagen im Straßenverkehr

Ausschnitt der Realität:



Objekte können klassifiziert werden

- Verkehrsteilnehmer:
 - Autos
 - Fahrräder
 - Motorräder
 - Fußgänger
 - Lastwagen
 - Busse
 - Straßenbahnen
 - ...
- Ähnliche Objekte gehören der gleichen Klasse an

Klassen können in Unterklassen eingeteilt werden

- **Klasse Auto:**

- Klasse Smart
- Klasse Audi
- Klasse BMW
- Klasse Ford
- ...

- **Klasse Fahrrad**

- Klasse Mountain- Bike
- Klasse Trekking-Rad
- Klasse Kinderfahrrad
- ...

- **Klasse Fußgänger:**

- Klasse Kinder
- Klasse Senioren
- Klasse Berufstätige
- ...

- **Busse**

- Klasse Gelenkbusse
- Klasse Niederflurbusse
- Klasse Citybusse
- ...

Objekte der Klasse "Smart"



Eigenschaften der Smart-Objekte

- Alle Smarts sind sich ähnlich
- Ähnlich = gleiche Klasse
- Die einzelnen Smarts unterscheiden sich durch
 - unterschiedliche Werte in den Attributen
 - Farbe: rot, blau, schwarz,...
 - Dachart: geschlossen, Cabrio
 - Motorleistung: 33 kW, 40 kW, 45 kW
 - Baujahr: ganze Zahl > 1980?
 - Fahrgestellnummer: Zeichenkette
 - ...
- Werte sind von verschiedenem Typ (Datentyp)

Notation für die Attribute einer Klasse

- Baujahr : ganze Zahl
- Fahrgestellnummer : Zeichenkette
- Dachart : geschlossen/Cabrio
- Motorleistung : ganze Zahl
- Farbe: Aufzählung

Fähigkeiten der Smart-Objekte

- Man kann die Fahrertür öffnen/schließen
- Man kann die Beifahrertür öffnen/schließen
- Man kann den Motor starten/stoppen
- Man kann die Lenkung links/rechts einschlagen
- Man kann beschleunigen/bremsen
- ...

Alle Objekte der Klasse Smart verfügen über die gleichen Fähigkeiten.

Nutzbare Fähigkeiten von Objekten heißen Methoden

Die Klasse Smart

- Attribute (speichern Werte)
 - Farbe
 - Motorleistung
 - Dachart
 - Fahrgestellnummer
 - ...
- Methoden (bewirken etwas)
 - Tür auf/zu
 - Motor an/aus
 - Gas geben
 - Bremsen
 - ...

Zwei Objekte der Klasse Smart

- Karls Kiste:
 - Farbe: rot
 - Motorleistung: 45 kW
 - Dachart: geschlossen
 - Fahrgestellnummer: SM1234
- Friedas Flitzer:
 - Farbe: grün
 - Motorleistung: 33 kW
 - Dachart: Cabrio
 - Fahrgestellnummer: SM5678



Objekte werden durch Namen identifiziert!

Fazit

- Interessierende Ausschnitte der Realität können durch geeignete Auswahl von Objekten modelliert werden
- Objekte können auf Basis ihrer Eigenschaften (Attribute) klassifiziert werden
- Eine Klasse fasst Objekte mit ähnlichen Eigenschaften (Attributen) zusammen
- Durch feinere Differenzierung von Klassen können Unterklassen gebildet werden

Orientierung

- Objekte und Klassen als Modelle der realen Welt
- Programm-Objekte als vereinfachte Modelle realer Objekte
- Lösen von Programmieraufgaben mit Hilfe von Programm-Objekten

Modellierung von Klassen und Objekten mit Java

- Die relevanten Klassen und Objekte sollen mit den Mitteln der Programmiersprache Java modelliert werden:
 - Namen für Klassen, Objekte, Attribute und Methoden müssen den Regeln von Java entsprechen
 - Typen der realen Attribute durch Datentypen der Programmiersprache Java abbilden
 - Fähigkeiten der realen Objekte durch Java-Methoden abbilden

Deklaration einer Methode (1)

- Real:

- Man kann die Fahrertür öffnen
- zusätzliche Informationen (Parameter) sind nicht erforderlich, um die Methode nutzen zu können

- Java:

- Name der Methode: **oeffneFahrertuer**
- zusätzliche Parameter nicht erforderlich: **()**
- Signatur der Methode: **oeffneFahrertuer ()**

Deklaration einer Methode (2)

- Real:

- Man kann den Kilometerstand ablesen
- zusätzliche Informationen sind nicht erforderlich, um die Methode nutzen zu können
- die Methode liefert ein verwertbares Ergebnis, nämlich den Kilometerstand als ganze Zahl

- Java:

- Name der Methode: **liesKmStand**
- zusätzliche Parameter nicht erforderlich: **()**
- Datentyp des Ergebnisses: **int**
- Signatur der Methode: **int liesKmStand ()**

Deklaration einer Methode (3)

- Real:

- Man kann die Geschwindigkeit erhöhen
- zusätzlich erforderlicher Parameter:

Um welchen Betrag soll beschleunigt werden?
(km/h-Wert als ganze Zahl)

- die Methode liefert kein verwertbares Ergebnis

- Java:

- Name der Methode: **beschleunigeUm**
- Name des Parameters: **kmh**
- Datentyp des Parameters: **int**
- Signatur der Methode: **beschleunigeUm (int kmh)**

Methoden der Klasse Smart

- Neue Regel:
 - Methoden, die kein Ergebnis liefern, erhalten den Datentyp **void** als Ergebnistyp
- Signaturen der bisher deklarierten Methoden:

Orientierung

- Objekte und Klassen als Modelle der realen Welt
- Programm-Objekte als vereinfachte Modelle realer Objekte
- Lösen von Programmieraufgaben mit Hilfe von Programm-Objekten

Programmieraufgaben mit Hilfe von Objekten lösen

- Objekte mit den geeigneten Fähigkeiten (Methoden) müssen verfügbar sein:
 - Objekt "ist da" und kann über seinen Namen angesprochen werden
 - Objekt "ist noch nicht da" und muss neu erzeugt werden
- Methoden eines oder mehrerer Objekte werden nacheinander in der richtigen Reihenfolge "ausgeführt"

Objekte der Klasse Smart

- Karls Kiste

Java: **karlsKiste**



- Friedas Flitzer

Java: **friedasFlitzer**



Objekte werden durch Namen identifiziert!

Sprachgebrauch

- **karlsKiste** ist
 - ein Objekt der Klasse **Smart** oder
 - ein Exemplar der Klasse **Smart** oder
 - eine Instanz der Klasse **Smart**
- Die Klasse **Smart** definiert die Attribute und Methoden ihrer Objekte
- Ein Objekt der Klasse **Smart** wird
 - neu erzeugt oder
 - instanziiert oder
 - konstruiert

Erteilen einer Anweisung durch
Aufrufen einer Methode (1)

- Voraussetzungen:
 - Objekt **karlsKiste** der Klasse **Smart** ist bereits erzeugt
 - Objekte der Klasse **Smart** verfügen über eine Methode mit folgender Signatur: **void oeffneFahrertuer ()**
- Aufrufen der Methode für ein bestimmtes Objekt:

karlsKiste . oeffneFahrertuer () ;

Welches Objekt
ist gemeint?

Punkt trennt
Objektnamen
vom
Methodennamen

welche Methode
ist gemeint?

Zusätzliche Parameter
in runden Klammern
(hier: keine Parameter)

Erteilen einer Anweisung durch
Aufrufen einer Methode (2)

- Voraussetzungen:
 - Objekt **karlsKiste** der Klasse **Smart** ist bereits erzeugt
 - Objekte der Klasse **Smart** verfügen über eine Methode mit folgender Signatur: **void beschleunigeUm (int kmh)**
- Aufrufen der Methode für ein bestimmtes Objekt:

karlsKiste . beschleunigeUm (20) ;

Welches Objekt
ist gemeint?

Punkt trennt
Objektnamen
vom

Methodennamen

welche Methode
ist gemeint?


Zusätzliche Parameter
in runden Klammern
(hier: Parameter)

Objekt einer Klasse erzeugen


- Ist noch kein Objekt vorhanden, muss es erzeugt werden.
- Von jeder Klasse können beliebig viele Objekte erzeugt werden.

Beispiel:


```
karlsKiste = new Smart();
```




Name des
neuen
Objekts



New erzeugt ein
neues Objekt



Zu welcher
Klasse gehört
das Objekt



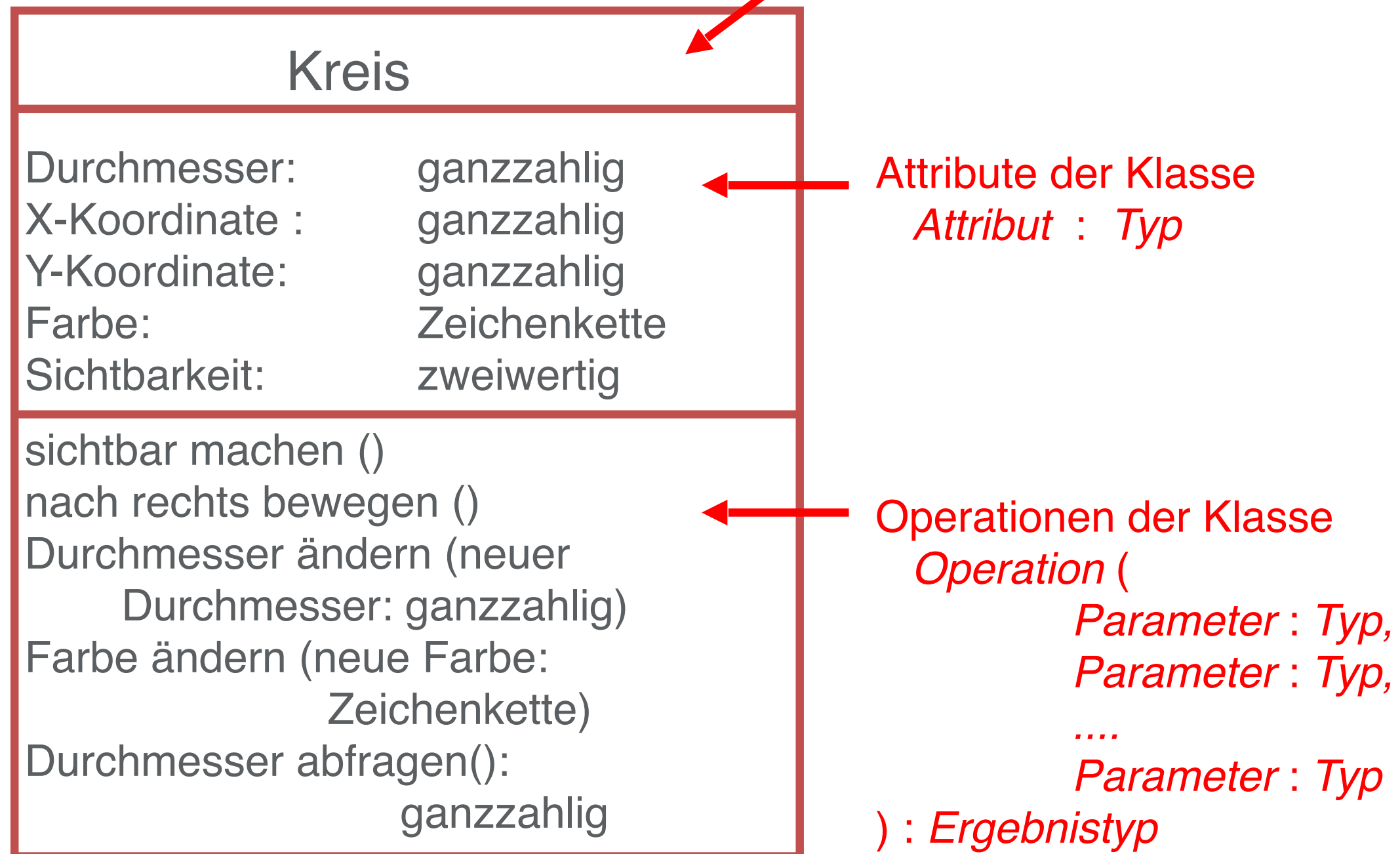
Keine zusätzlichen
Parameter beim Erzeugen.

Rückblick Lernziele

- Fachausdrücke und Konzepte der objektorientierten Programmierung verstehen
- Einfache Programmieraufgaben durch Aufrufen von Methoden lösen können

- **Umgangssprache:** umständlich, viel Text
- **UML-Klassendiagramm** (*Unified Modeling Language*): übersichtliche grafische Darstellung einer Klasse mit den wesentlichen Informationen
- **Java-Quellcode:** Geschriebener Text in der Programmiersprache Java; angelehnt an englische Sprache und mathematische Formelschreibweise

wenig formal, teilweise
umgangssprachlich



Name der Klasse,
nach Java-Konventionen;
fett bzw. nicht unterstrichen

sehr formal, angelehnt
an Java-Sprachregeln

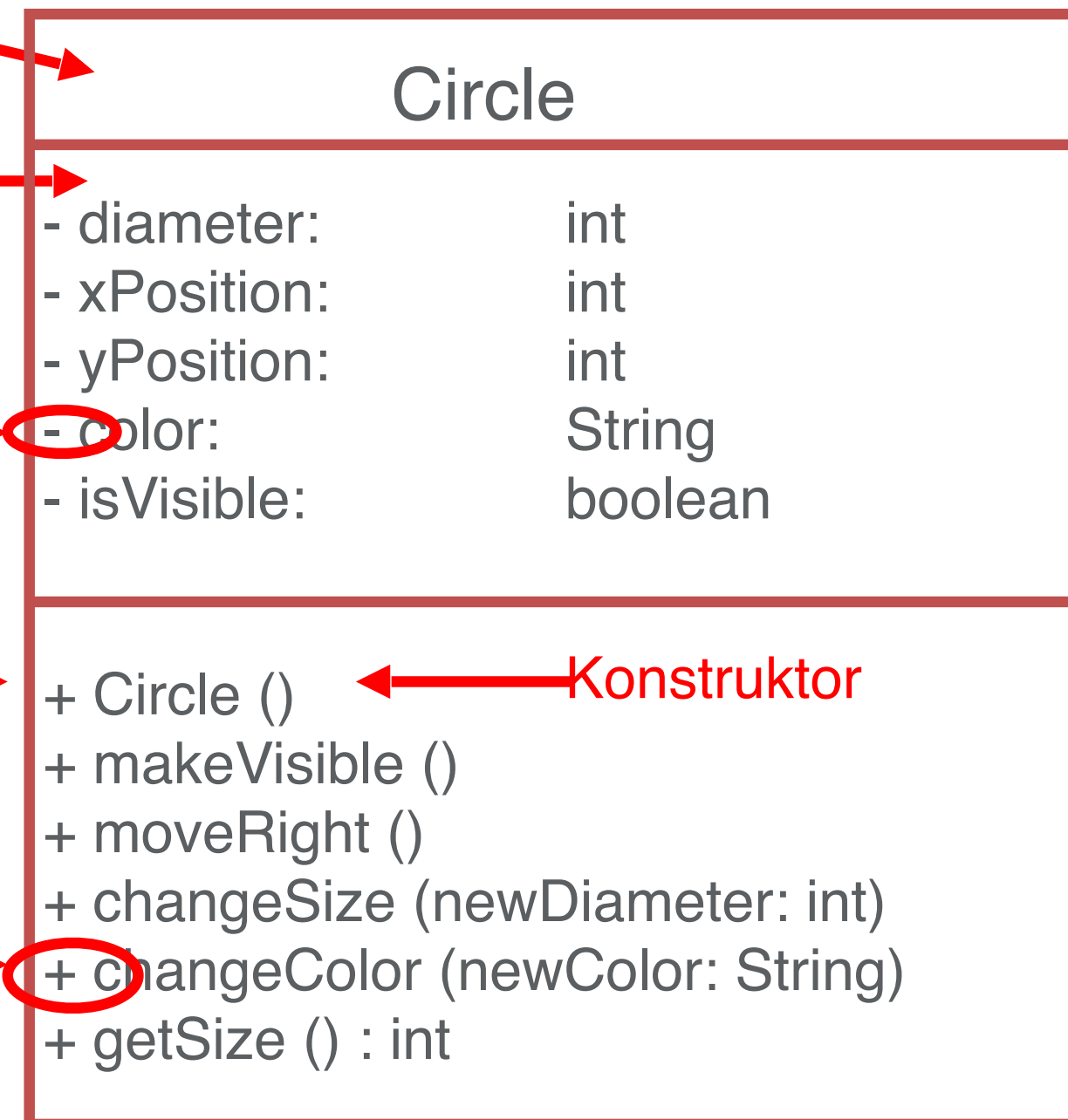
Attribute der Klasse entsprechen
Instanzvariablen;
Namen und Typen nach Java-
Konventionen

Instanzvariable : Java-Datentyp

- kennzeichnet private
Instanzvariable

Operationen der Klasse entsprechen
Methoden;
Namen, Parameter und Typen nach Java-
Konventionen

+ kennzeichnet öffentliche
Operation



wenig formal, teilweise
umgangssprachlich

Kreis	
Durchmesser:	ganzzahlig
X-Koordinate :	ganzzahlig
Y-Koordinate:	ganzzahlig
Farbe:	Zeichenkette
Sichtbarkeit:	zweiwertig
sichtbar machen ()	
nach rechts bewegen ()	
Durchmesser ändern (neuer Durchmesser: ganzzahlig)	
Farbe ändern (neue Farbe: Zeichenkette)	
Durchmesser abfragen(): ganzzahlig	

sehr formal, angelehnt
an Java-Sprachregeln

Circle	
- diameter:	int
- xPosition:	int
- yPosition:	int
- color:	String
- isVisible:	boolean
+ Circle ()	
+ makeVisible ()	
+ moveRight ()	
+ changeSize (newDiameter: int)	
+ changeColor (newColor: String)	
+ getSize (): int	

- In einer Methode werden Anweisungen zusammengefasst.
- Wird eine Methode aufgerufen, so werden die darin enthaltenen Anweisungen nacheinander ausgeführt.
- Anweisungen sind z.B.:
 - Aufrufen der Methoden anderer Objekte
 - Verknüpfen von Variablen durch Operatoren, z.B. Addieren
 - Speichern von Werten in Variablen (Zuweisungen)