Überblick

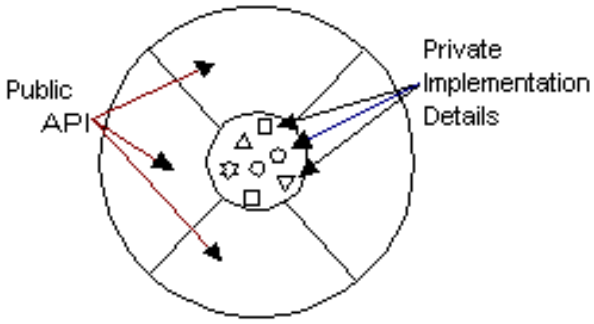
- Von C nach Java
- Background
- Konzepte des objektorientierten Programmierens
- Ein ausgearbeitetes Java-Programm als Beispiel
- Ein weiteres Java-Programm als Beispiel
- Typen und Subtypen
- Vom Design zum Programmieren (OOD zu OOP)
- Patterns in OOP
- Testen (von objektorientierten Programmen)

Konzepte des objektorientierten Programmierens

- Was ist eine Klasse (in OOP)?
- Datenkapselung
- Was ist eine Message?
- Subklassen / Superklassen
- Was ist Vererbung (in OOP)?
- Was ist Polymorphismus (in OOP)?
- Was ist ein Interface (in Java)?
- Abstrakte Klasse vs. Interface (in Java)

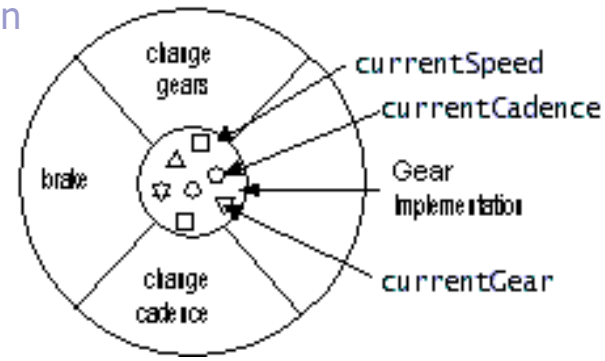
Was ist eine Klasse (in OOP)?

Definition: Eine Klasse ist ein „Prägestempel" oder Prototyp, der die Variablen und die Methoden gemeinsam für alle Objekte einer bestimmten Art definiert.



Was ist eine Klasse (in OOP)? – Fahrrad-Beispiel

- Instanz-Variablen, z.B. currentGear definieren Zustand
- Instanz-Methoden, z.B. changeGears definieren Verhalten



Was ist eine Klasse (in OOP)? – Beispiel in Java

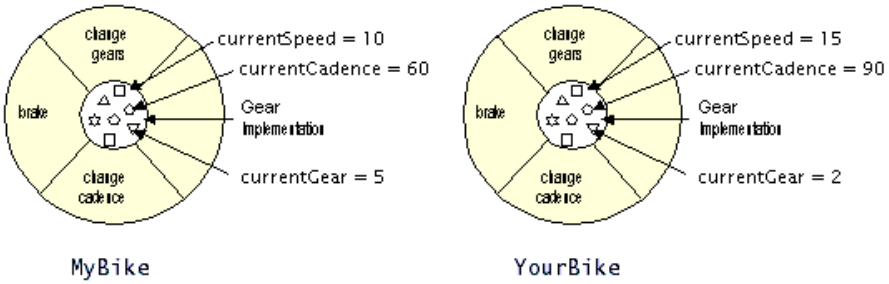
```
public class Bicycle {
    //instance variables
    private int currentGear;
    …
    //instance methods
    public void changeGears (int gear) {
        this.currentGear = gear;
    }
    …
}
```


Was ist eine Klasse (in OOP)? – Kreieren von Instanzen

Kreiere Instanzen (Objekte) der Fahrrad-Klasse, z.B. MyBike und YourBike:

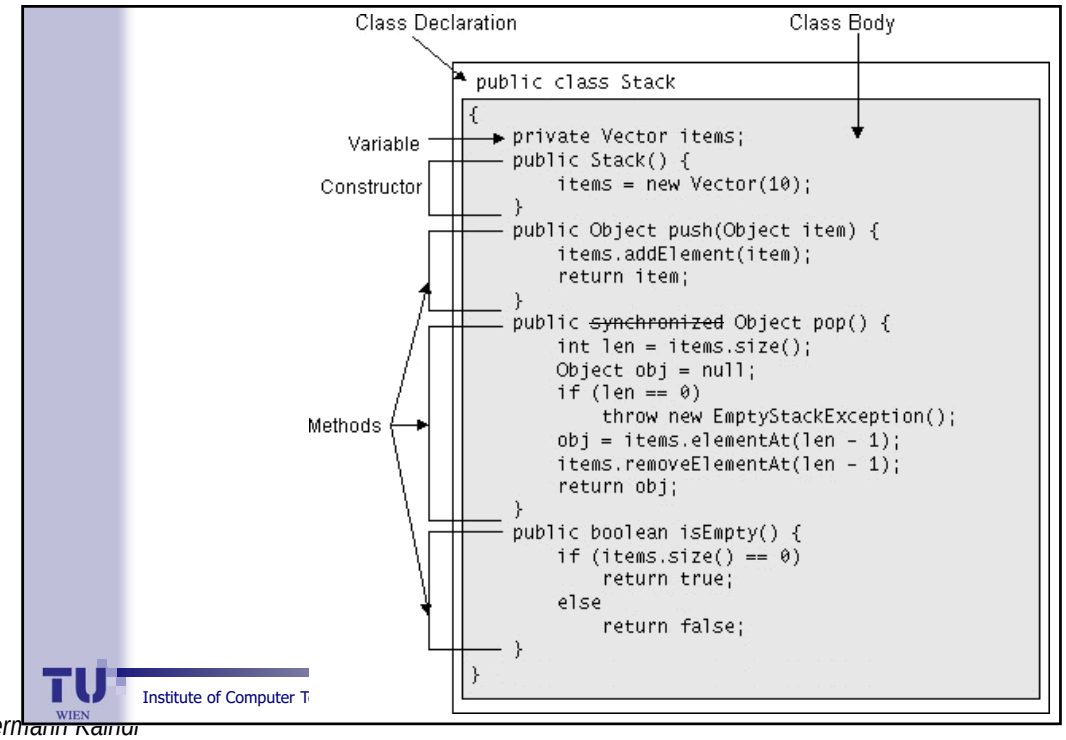
MyBike = **new** Bicycle (…);

YourBike = **new** Bicycle (…);



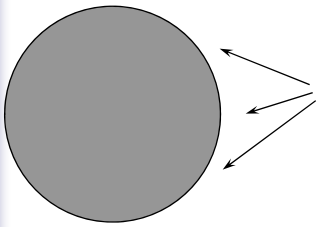
Was ist eine Klasse (in OOP)? – Constructor

- Unterprogramm zum Initialisieren neuer Objekte, die von einer Klasse kreiert wurden
- Hat denselben Namen wie die Klasse
- Initiale Werte, vom Aufrufer dem Constructor als Argumente übergeben



Datenkapselung

Schutz von Daten innerhalb eines Objektes gegen direkten Zugriff von außen



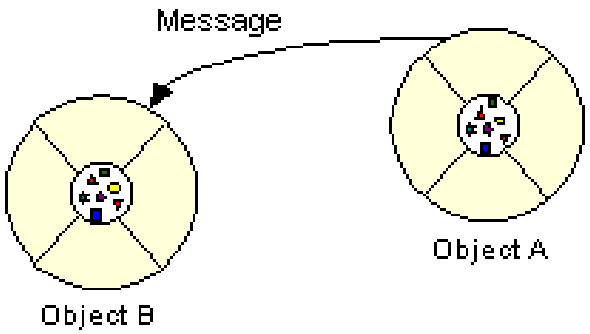
Zugriff nur durch Methoden

Abstrakter Datentyp

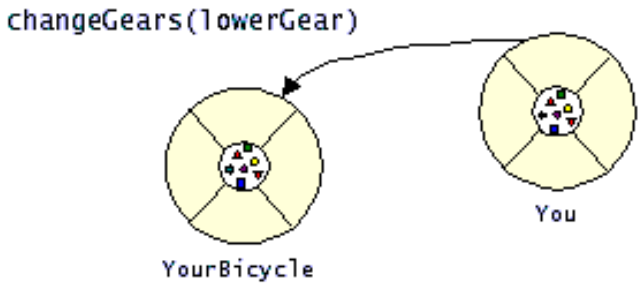
Information hiding

Was ist eine Message?

Softwareobjekte interagieren und kommunizieren miteinander, indem sie einander Messages senden.



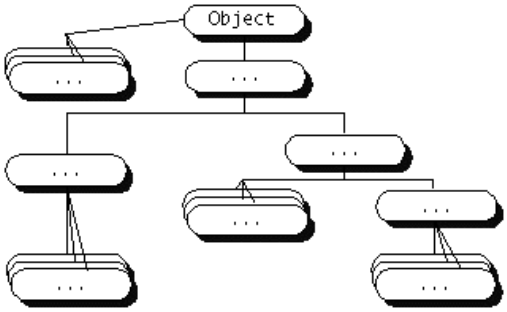
Was ist eine Message ? – Fahrrad-Beispiel



YourBicycle.changeGears(lowerGear);

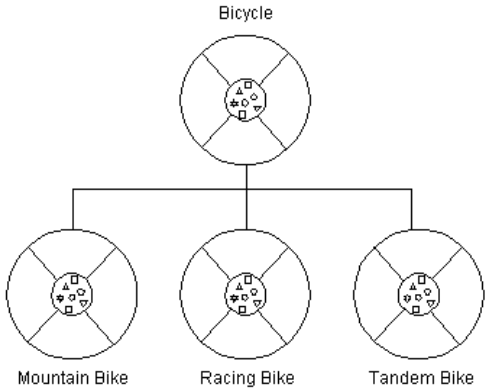
Empfangendes Objekt — Name der Methode — Parameter

Subklassen / Superklassen



- Eine Subklasse ist eine Klasse, die eine andere Klasse „erweitert".
- Eine Superklasse ist eine Klasse, von der eine bestimmte Klasse abgeleitet wird, auch indirekt.

Subklassen / Superklassen – Fahrrad-Beispiel



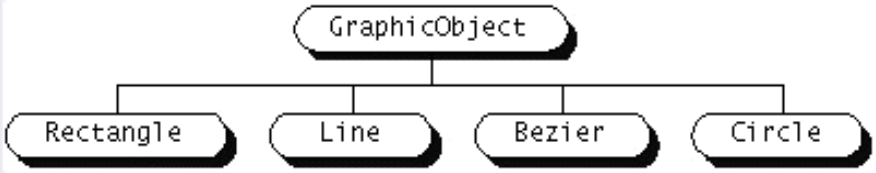
public class MountainBike **extends Bicycle** { … };

Was ist Vererbung (in OOP)?

- Mechanismus für Halten gemeinsamer Information
- Eine Klasse erbt Zustand und Verhalten von ihrer Superklasse. Z.B. übernimmt MountainBike currentSpeed und brake von Bicycle.
- Darüber hinaus können Subklassen Variablen und Methoden hinzufügen. Z.B. hat TandemBike einen zweiten Sitz.
- Verhalten kann auch spezialisiert werden. Z.B. überschreibt MountainBike mit zusätzlichen Gängen changeGears.

Was ist Polymorphismus (in OOP)?

- Vielgestaltigkeit
- Zeichnen kann eine polymorphe Operation in graphischen Objekten sein.
- Z.B. ist das Zeichnen eines Kreises ziemlich unterschiedlich vom Zeichnen eines Rechtecks.



Was ist Polymorphismus (in OOP)? – Java

```
abstract class GraphicObject {
    //cannot have instances
    abstract void draw();
}

public class Circle extends GraphicObject {
    void draw() {
            …
    }
}

public class Rectangle extends GraphicObject {
    void draw() {
            …
    }
}
```

Polymorphismus und Messages – Java

```
GraphicObject go0, go1, go2, go3;
boolean query;

go0 = new GraphicObject();
go1 = new Circle();
go2 = new Rectangle();
...
if ( query ) {
   go3 = go1;
} else {
   go3 = go2;
}
...
go3.draw();
```

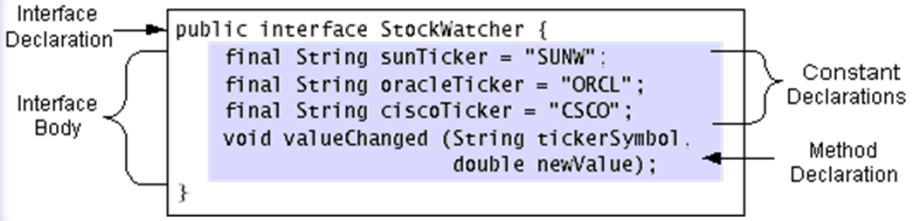

Was ist ein Interface (in Java)?

- **Definition:** Ein Interface ist eine benannte Sammlung von Methodendefinitionen (ohne Implementierungen).
- Analog zu einem Protokoll (einem vereinbarten Verhalten)

`public`	Makes this interface public.
`interface InterfaceName`	This is the name of the interface.
`Extends SuperInterfaces`	This interface's superinterfaces.

`{`
`    InterfaceBody`
`}`	

Was ist ein Interface (in Java)? – Beispiel




```
public class StockWatcherObj implements StockWatcher {
    ...
    public void valueChanged(String tickerSymbol,
                             double newValue) {
        ...
    }
}
```

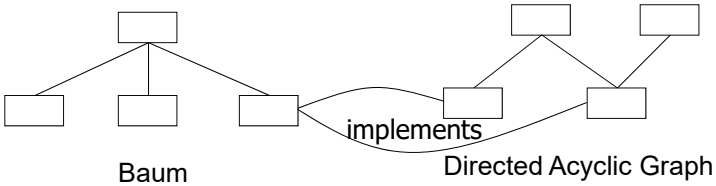

Abstrakte Klasse vs. Interface (in Java)

Klasse

- kann eine Methode implementieren
- ist Teil einer Klassenhierarchie
- kann nur eine Klasse erweitern

Interface

- kann das nicht
- ist nicht Teil einer Klassenhierarchie
- kann eine beliebige Anzahl von Schnittstellen erweitern



Baum

Directed Acyclic Graph (DAG)