TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Fakultät Informatik

Professur Softwaretechnologie

OOSE_02

VERERBUNG UND POLYMORPHIE MIT BLUEJ

CRC-KARTENMETHODE

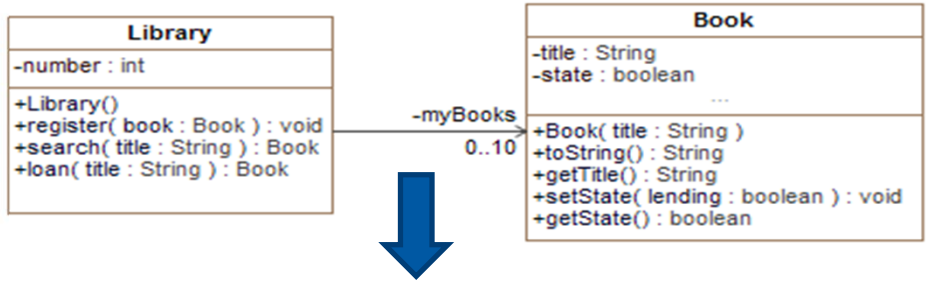
Dr.-Ing. Birgit Demuth

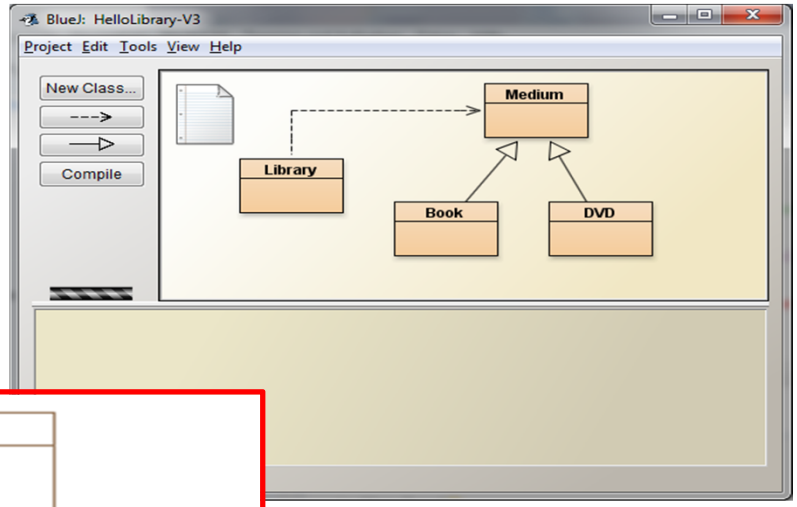
Sommersemester 2017

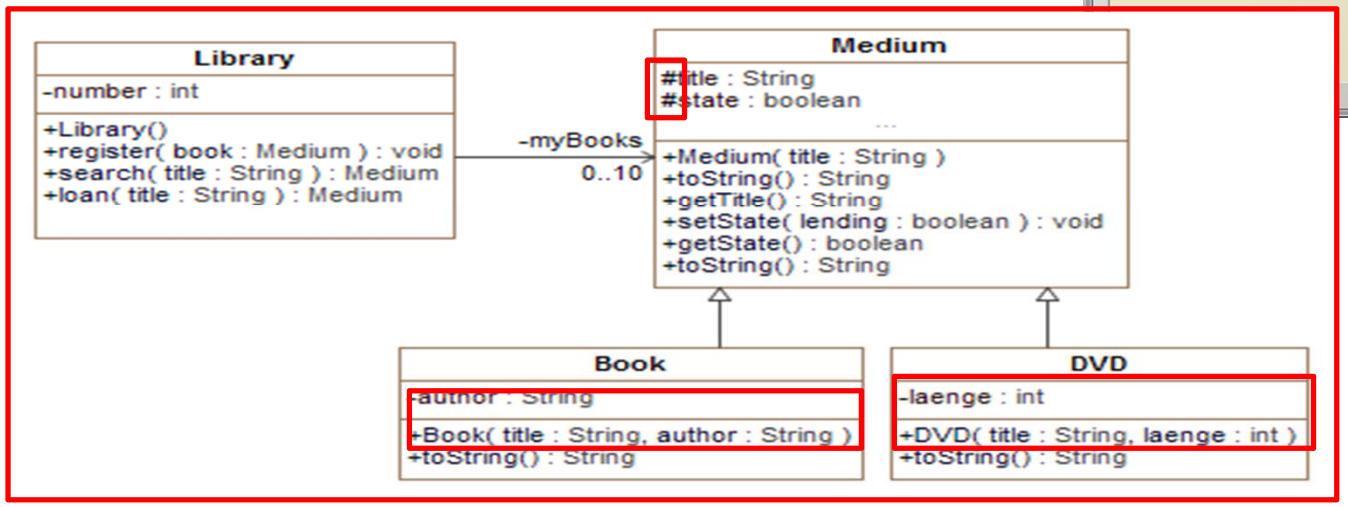
Vererbung und Polymorphie mit BlueJ

[Barnes2012]

BlueJ: Vererbung (1)
Erweiterung von HelloLibrary (U02)







Using inheritance (Wiederholung)

define one **superclass** : **Medium**
define **subclasses** for **Book** and **DVD**
the superclass defines common attributes **title** and **state**
the subclasses **inherit** the superclass attributes **title** and **state**
the subclasses add own attributes **autor** bzw. **laenge**

Wiederholungsfragen (AMCS)

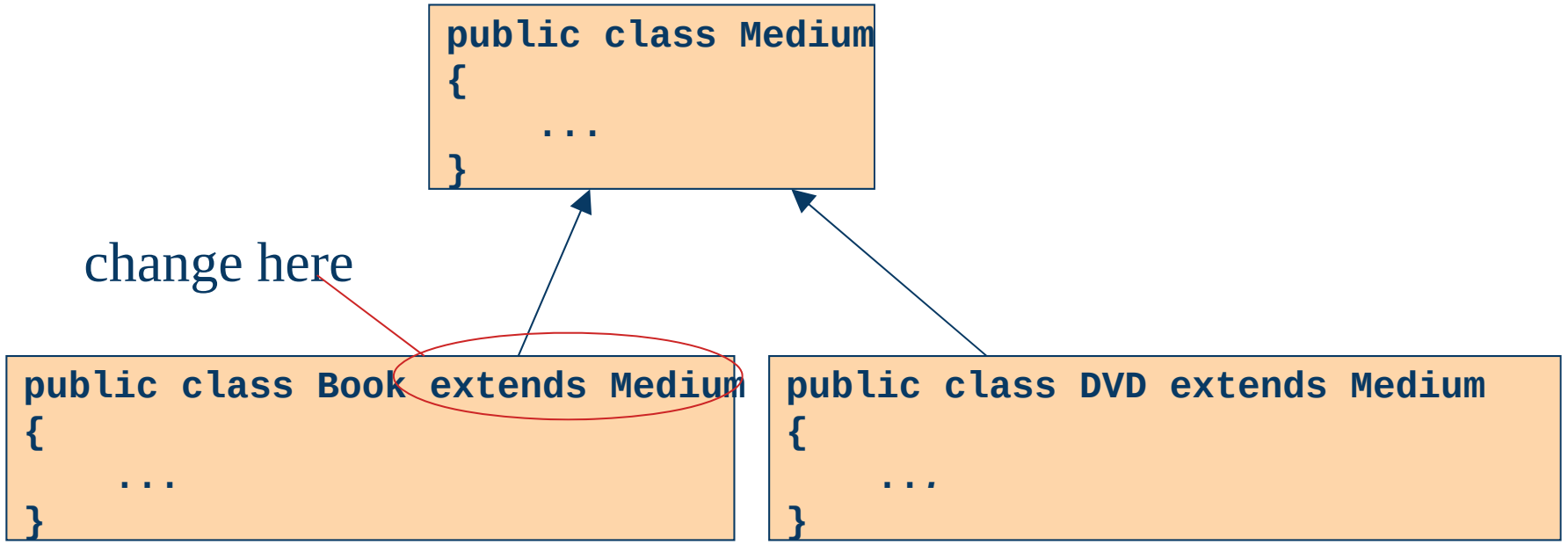
- Ist diese Konstruktion in Java möglich?

```
class A extends C, D, E { ...}
```

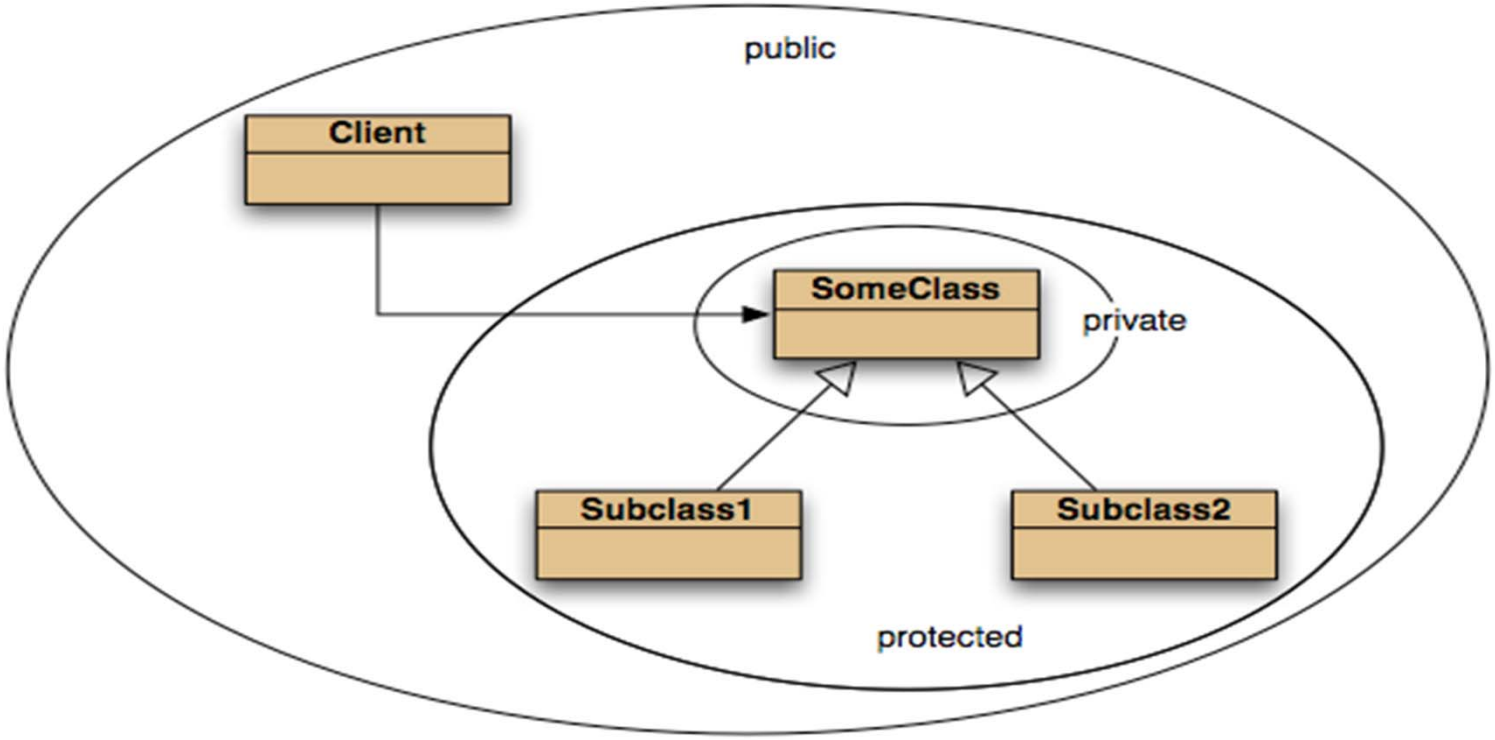
- Ist diese Konstruktion in Java möglich?

```
class A implements C, D, E { ...}
```

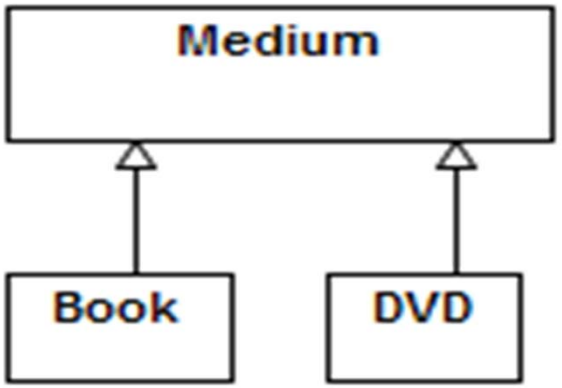
Inheritance in Java



Access levels



Subtyping and assignment



subclass objects may be assigned to superclass variables

```
Medium m1 = new Medium(…);
Medium b1 = new Book(…);
Medium d1 = new DVD(…);
```

Static and dynamic type

What is the type of b1?

```
Book b1 = new Book(…);
```

What is the type of m1?

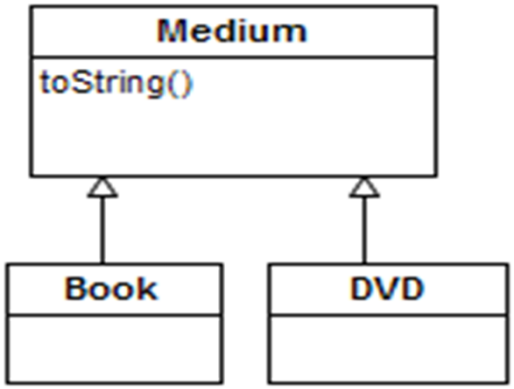
```
Medium m1 = new Book(…);
```

- The declared type of a variable is its **static type**.
- The type of the object a variable refers to is its **dynamic type**.

The Problem

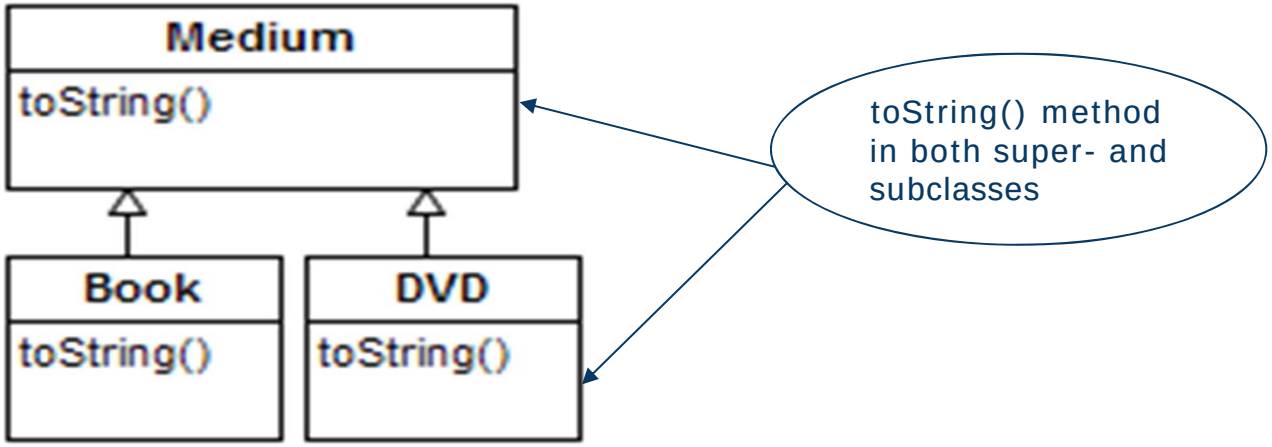
The toString() method in Medium only prints the common fields (title).
Inheritance is a one-way street:

- A subclass inherits the superclass fields.
- The superclass knows nothing about its subclass's fields.

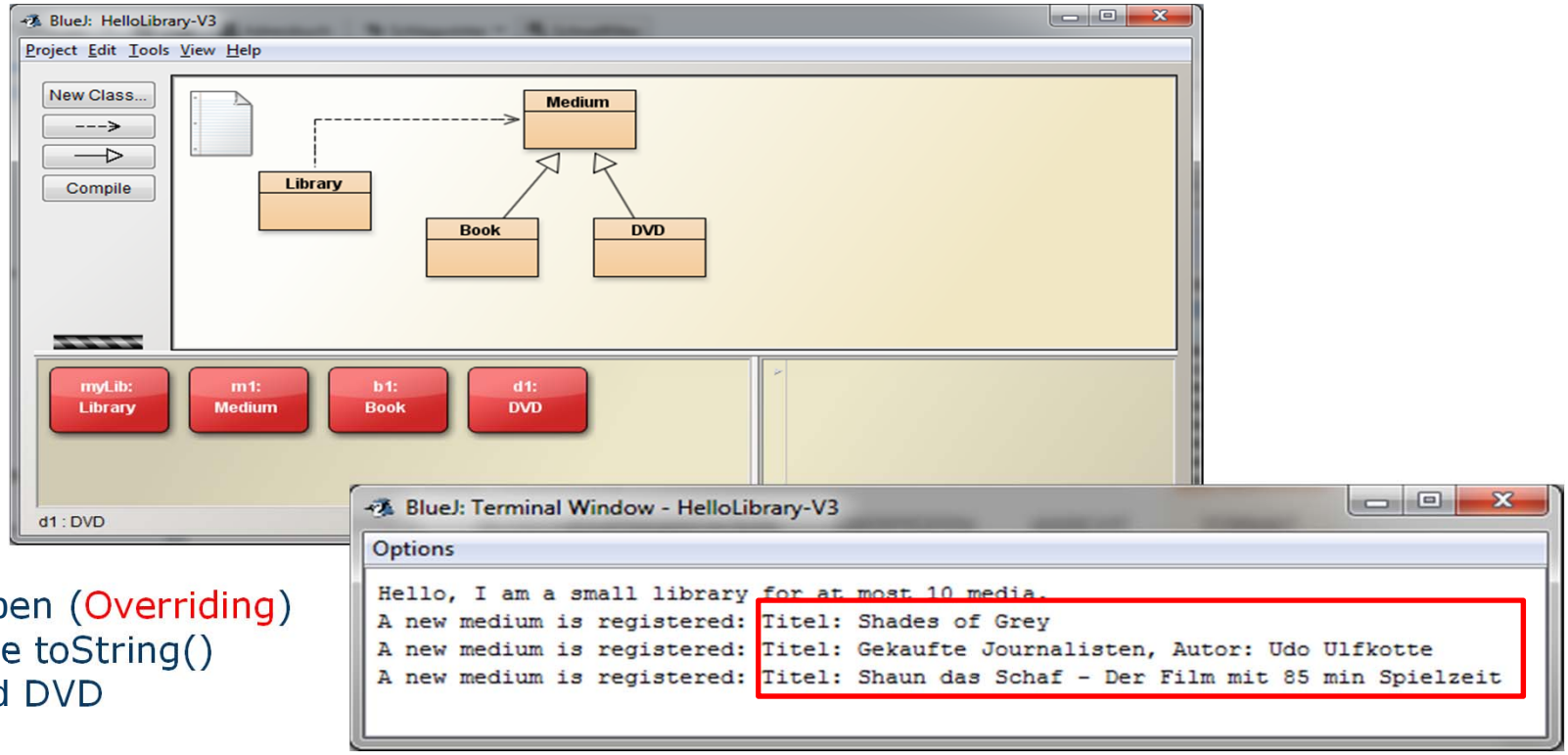


Overriding: the Solution

- Superclass and subclass define methods with the same signature.
- Each has access to the fields of its class.
- Superclass satisfies static type check.
- Subclass method is called at runtime – it *overrides* the superclass version.



BlueJ: Vererbung (2) - Demo
Erweiterung von HelloLibrary (U02)



Überschreiben (Overriding)
der Methode toString()
in Book und DVD

Method lookup / Polymorphic method dispatch

Method calls are polymorphic.
The actual method called depends on the dynamic object type:

- The variable is accessed.
- The object stored in the variable is found.
- The class of the object is found.
- The class is searched for a method match.
- If no match is found, the superclass is searched.
- This is repeated until a match is found.
- Overriding methods take precedence.
- The matched method is dispatched.

Super call in methods

Overridden methods are hidden ...

... but we often still want to be able to call them.

An overridden method *can* be called from the method that overrides it.

- **`super.method(...)`**
- use of **`super()`** in constructors.

Overridden method

```
public class DVD
{
    ...
    public void toString(){
       return  super.toString() +
               " mit " + laenge + " min Spielzeit";

    }
}
```

TECHNISCHE UNIVERSITÄT DRESDEN

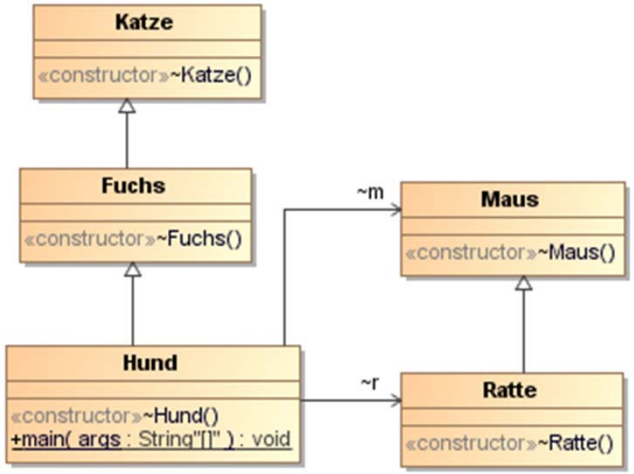
Beispiel Hund-Hierarchie (AMCS)

[Ratz2006]

[http://www.programmierenlernenhq.de/polymorphismus-und-konstruktoren-java/]

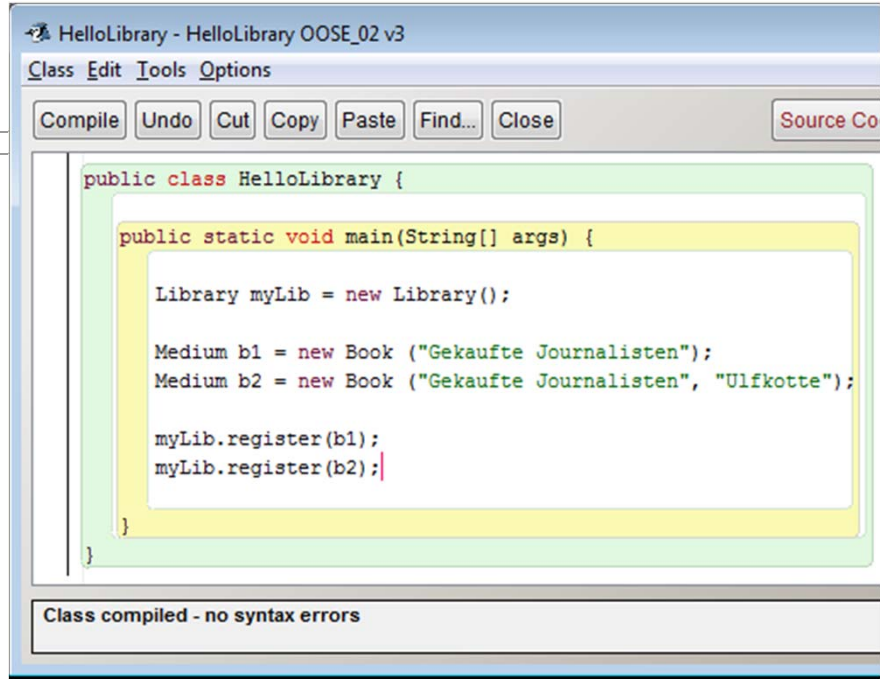
Was gibt das Programm Hund aus?

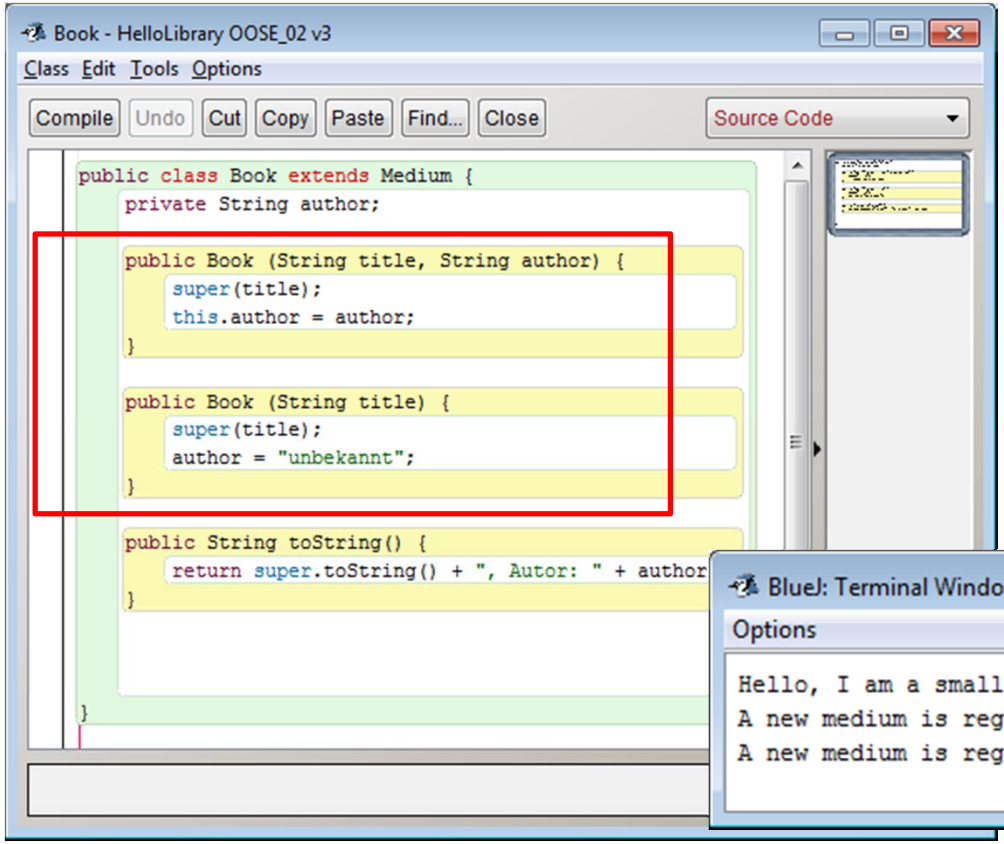
- Hund
- Hund Fuchs Katze
- Katze Fuchs Maus Ratte Hund
- Katze Fuchs Maus Maus Ratte Hund

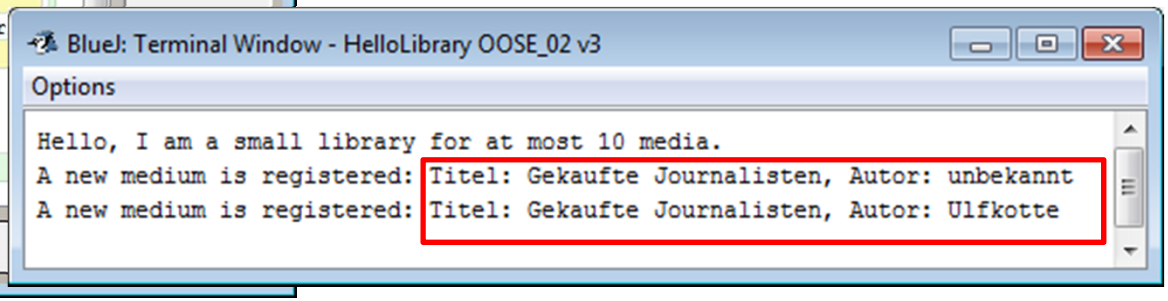


TECHNISCHE UNIVERSITÄT DRESDEN

BlueJ: Overloading of Constructors







Lessons learned (1)

- Inheritance allows the definition of classes as extensions of other classes.

Inheritance
- avoids code duplication
- allows code reuse
- simplifies the code
- simplifies maintenance and extending

- Variables can hold subtype objects.
- Subtypes can be used wherever supertype objects are expected (substitution).

Lessons learned (2)

- The declared type of a variable is its static type.
- Compilers check static types.
- The type of an object is its dynamic type.
- Dynamic types are used at runtime.
- Methods may be overridden in a subclass.
- Method lookup starts with the dynamic type.
- Protected access supports inheritance.

- Overriding means runtime (dynamic) polymorphism.
- Overloading means compile time (static) polymorphism.

Polymorphism (Polymorphie) - Two Types in Java

[http://beginnersbook.com/2013/03/polymorphism-in-java/]

Method Overriding

- Applies only to inherited methods
- Object type (NOT reference variable type) determines which overridden method will be used at runtime
- Overriding method can have different return type
- Overriding method must not have more restrictive access modifier
- Abstract methods must be overridden
- Static and final methods cannot be overridden
- Constructors cannot be overridden
- It is also known as **runtime polymorphism**

Method Overloading

- Overloading can take place in the same class or in its sub-class.
- Constructor in Java can be overloaded
- Overloaded methods must have a different argument list.
- Overloaded method should always be the part of the same class (can also take place in sub class), with same name but different parameters.
- The parameters may differ in their type or number, or in both.
- They may have the same or different return types.
- It is also known as **compile time polymorphism**.

CRC-Kartenmethode am Beispiel einer Klausuraufgabe

siehe Foliensatz von Prof. Aßmann → 12-st-crc-analysis.pdf

Beispiel: Auktionen (1)

Bei der bekannten *Englischen Auktion* werden, von einem festgesetzten Einstandspreis von einem Posten beginnend aufsteigend Gebote abgegeben, bis kein neues Gebot mehr eintrifft. Der letzte Bieter erhält den Zuschlag.

Die Entwickler eines Auktionssystems beginnen mit der CRC-Karten-Analyse. Zunächst haben sie vier Klassen (Karten) durch **Textanalyse** identifiziert.

Farblegende in der textuellen Domänenbeschreibung:

Klasse

Verantwortlichkeit

Attribut bzw. Rolle

Beispiel: Auktionen (2)

Auction für die einzelnen Auktionen. Jede Auktion kennt eine Liste ihrer zu versteigernden Posten (**allItems**) und eine Liste aller Bieter (**bidders**).

Item für die Posten, die für eine Versteigerung vorgesehen sind. Ein Item hat eine Beschreibung (**description**) und bekommt eine Nummer (**number**) sowie einen Einstandspreis (**minPrice**) zugewiesen. Jeder Posten kennt alle auf sich abgegebenen Gebote (**allBids**). Posten existieren nur für eine Auktion und werden wieder gelöscht, wenn die Auktion geschlossen wird.

Bid für die Gebote. Ein Gebot wird beschrieben durch einen Preis (**price**) und wird von einer Person (**bidder**) abgegeben.

Person für alle Personen, die als Bieter eines Postens in der Auktion auftreten. Von jeder Person wird der Name (**name**) gespeichert.

Beispiel: Auktionen (3)

Szenarium

Zunächst eröffnet (erzeugt) ein Auktionator eine **Auktion** (**openAuction**).

Danach registriert er die zu versteigernden Posten, indem er diese im System erzeugt und in einer Liste (**allItems**) registriert (**registerItem**).

Nachdem alle Posten registriert sind, kann die eigentliche Auktion beginnen, indem die Bieter Gebote (einen Preis) für einen bestimmten Posten abgeben (**bidItemBy**).

Das System vergleicht das abgegebene Gebot mit dem derzeit höchsten Gebot und erzeugt ein neues Gebot (**Bid**). Sofern der gebotene Preis (**price**) den Einstandspreis erreicht oder das derzeit höchste Gebot (**highestBid**) für den Posten überschreitet (**bidBy**), wird ein neues Höchstgebot (**highestBid**) vermerkt (**setHighestBid**).

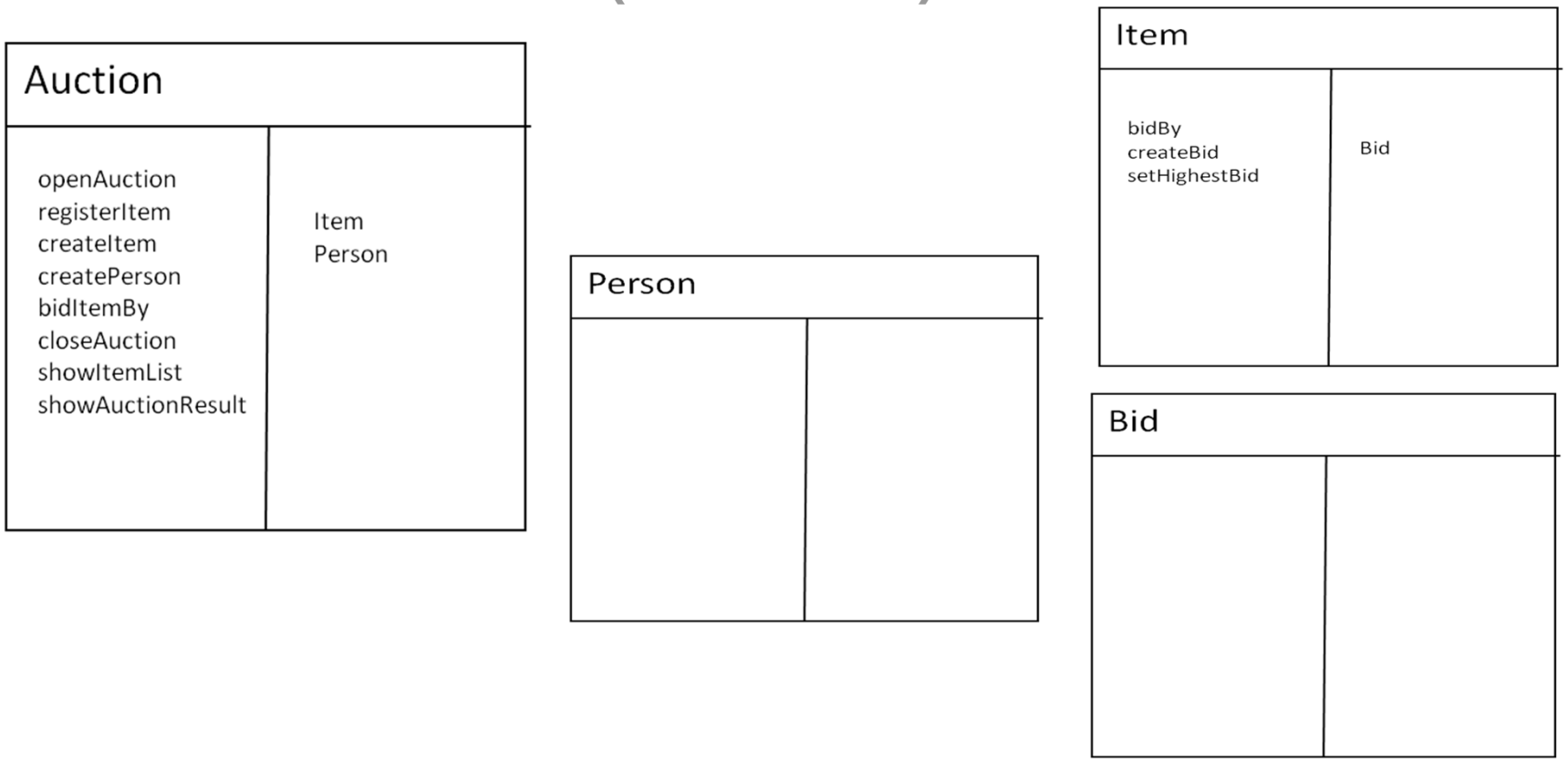
Beispiel: Auktionen (4)

Szenarium (Fortsetzung)

Für jedes Gebot wird der (genau ein) Bieter (**bidder**) vermerkt. Alle Bieter der Auktion werden genau einmal in einer Kollektion (**bidders**) gespeichert.

Wenn keine neuen Gebote eintreffen, schließt der Auktionator die **Auktion** (**closeAuction**), indem das Ergebnis der **Auktion** angezeigt wird (**showAuctionResult**).

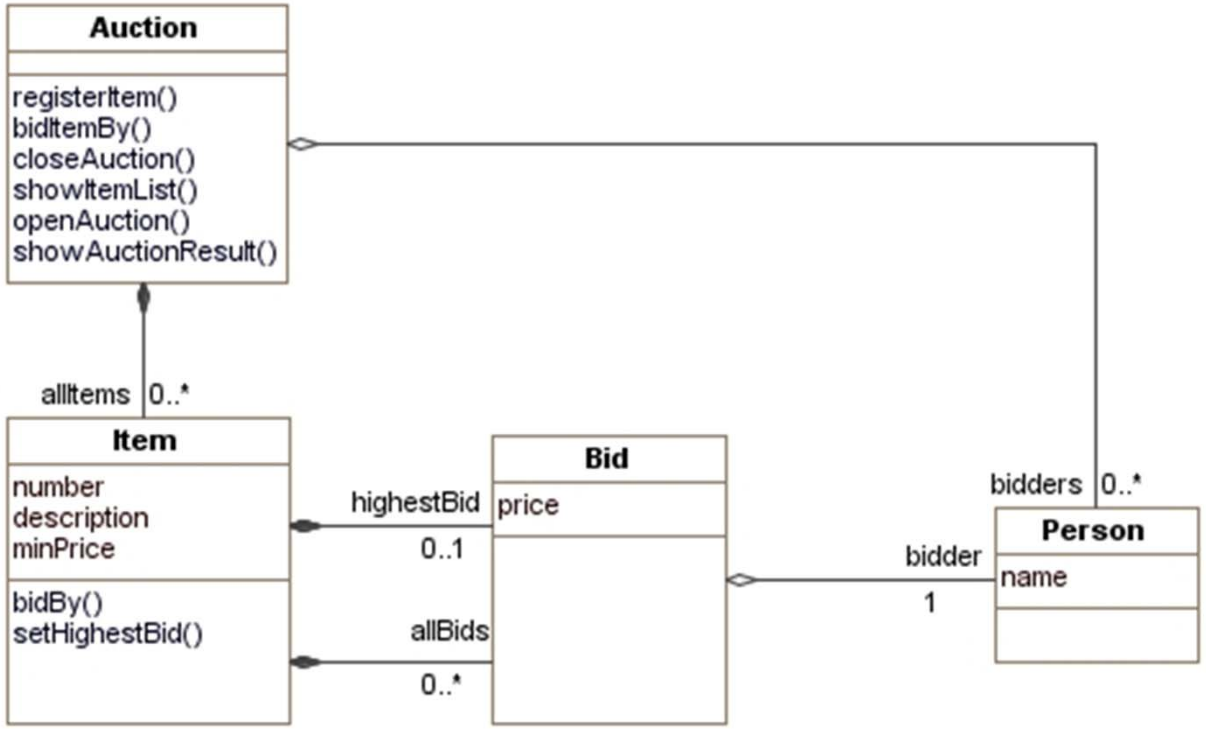
Auktionen: CRC-Karten (Vorderseite)



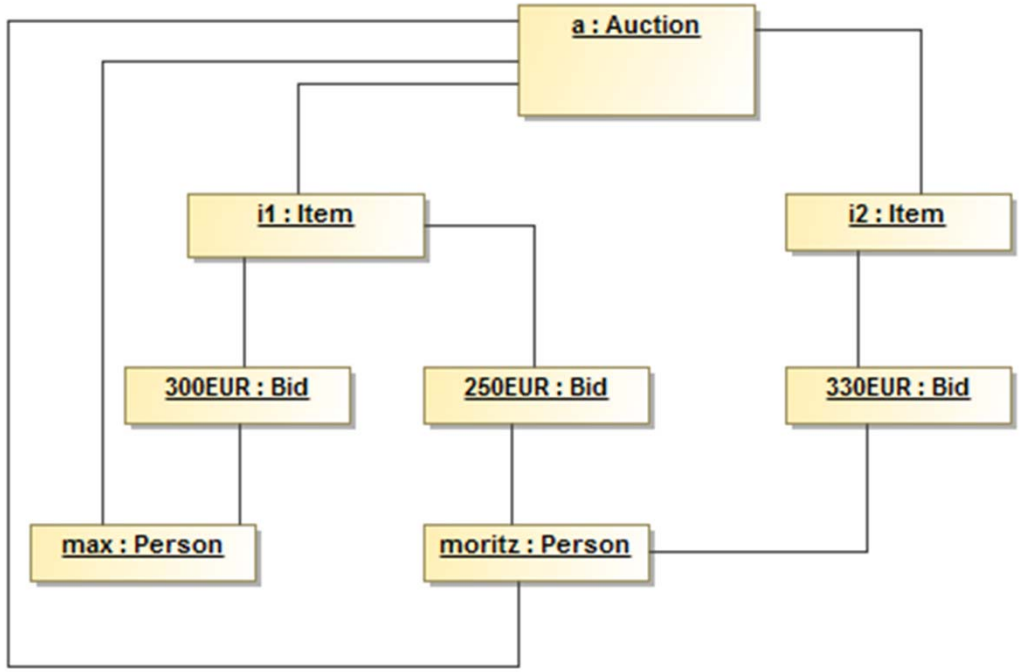
CRC2UML

CRC	UML-Analyseklassendiagramm
Klasse (Class)	Klasse
Verantwortlichkeit (Responsibility)	Methode
Mithelfer (Collaborator)	Beziehung (Assoziation/Aggregation/Komposition) zu anderer Klasse (Mithelfer)
Attribut auf Rückseite der CRC-Karte	Attribut einer Klasse oder Name des Assoziationsendes beim Mithelfer
Oberklasse / Unterklasse	Vererbung

Auktionen: Klassendiagramm

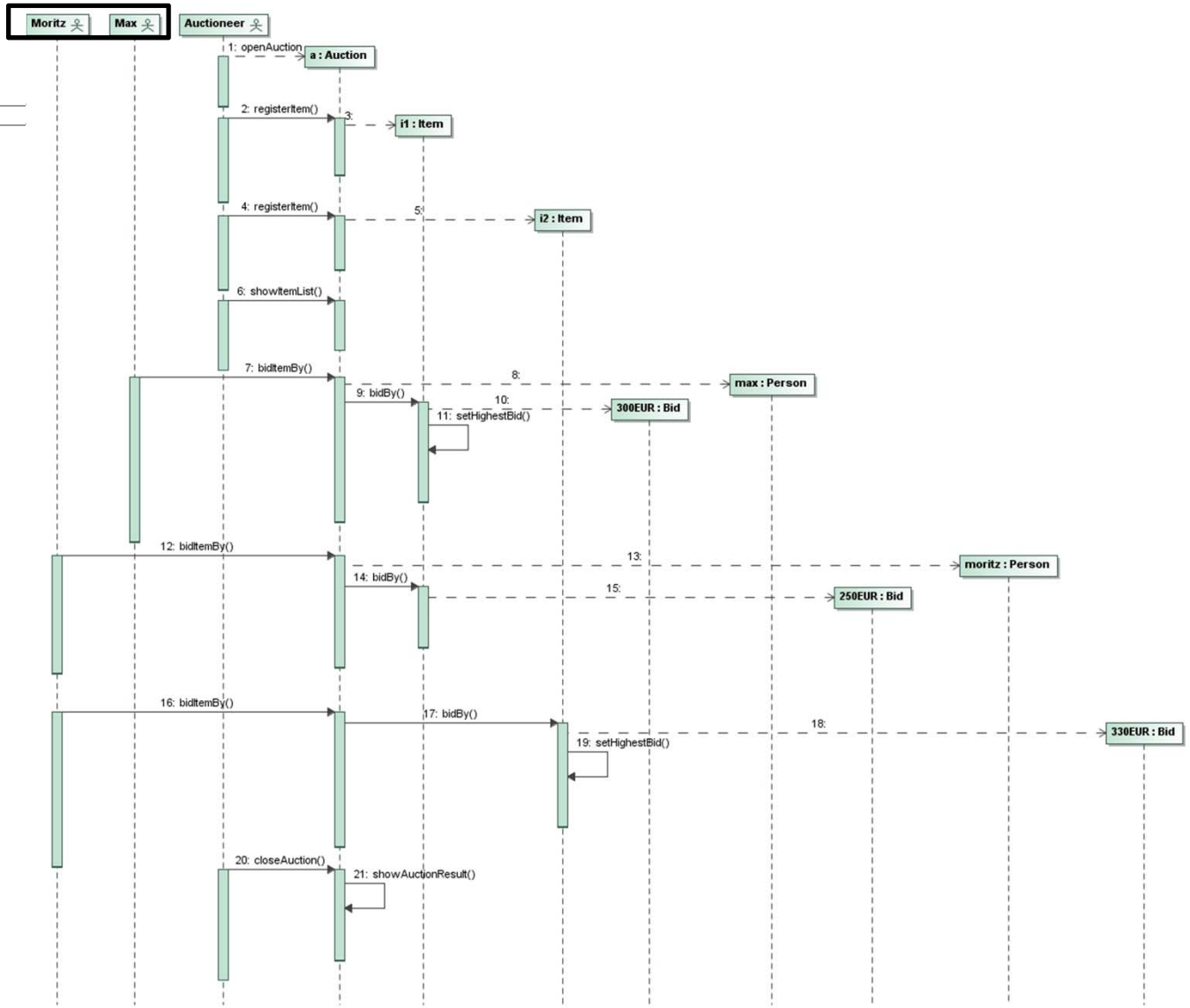


Objektdiagramm



TECHNISCHE UNIVERSITÄT DRESDEN

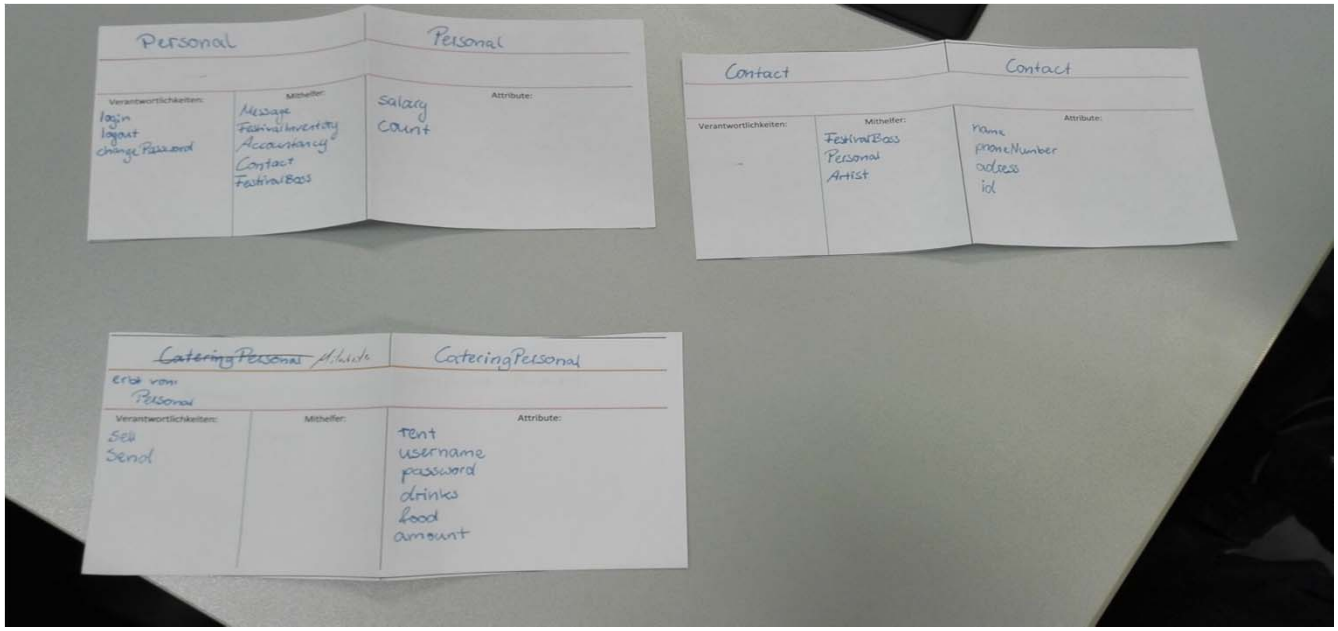
Auktionen: Rollenspiel dokumentiert in einem Sequenzdiagramm (mit **Akteuren**)



Tools für die Arbeit mit CRC Karten?

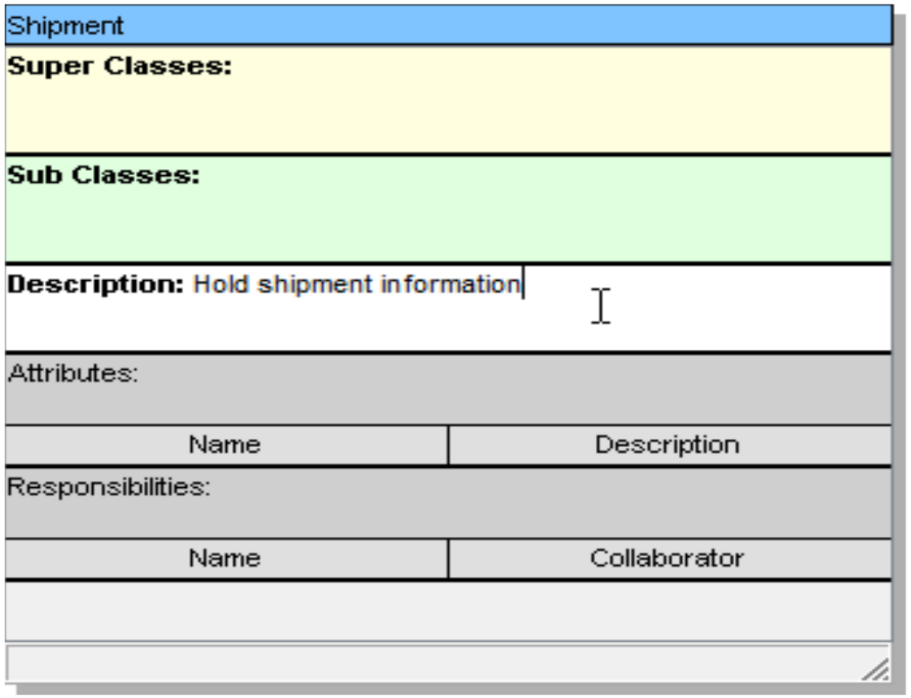
Klassische und manuell ausgefüllte Karteikarten ☺

Beispiel (swp15w14)



Tools für die Arbeit mit CRC Karten?

(UML Tool) Visual Paradigm
http://www.visual-paradigm.com/



Auktionen: CRC-Karten

Auction

Super Classes:

Sub Classes:

Description: repräsentiert eine Auktion

Attributes:

Name	Description
allItems	Liste von Posten
bidders	Liste alle Bieter einer Auktion

Responsibilities:

Name	Collaborator
openAuction	
registerItem	Item
bidItemBy	Item, Person
closeAuction	
showAuctionResult	

Item

Super Classes:

Sub Classes:

Description: Posten zur Versteigerung

Attributes:

Name	Description
number	identifiziert den Posten
description	Beschreibung des Postens
minPrice	Einstandspreis des Postens
allBids	Liste der abgegebenen Gebote für den Posten
highestBid	bislang höchstes Gebot für den Posten

Responsibilities:

Name	Collaborator
bidBy	Bid
setHighestBid	

Bid

Super Classes:

Sub Classes:

Description: ein Gebot für einen Posten

Attributes:

Name	Description
price	gebotener Preis
bidder	zugehöriger Bieter (Person)

Responsibilities:

Name	Collaborator
createBid	

Person

Super Classes:

Sub Classes:

Description: Person, die ein Gebot abgibt

Attributes:

Name	Description
name	

Responsibilities:

Name	Collaborator
createPerson	

Wiederholungsfragen (AMCS)

Textanalyse: Sind die folgenden Aussagen richtig?

- Die Methoden in einem System korrespondieren in etwa mit den Substantiven in der zugehörigen textuellen Systembeschreibung.
- Die Klassen in einem System korrespondieren in etwa den Verben in der zugehörigen textuellen Systembeschreibung.

CRC-Kartenmethode

- Wie funktioniert die CRC-Kartenmethode?
- Wie wird eine Verantwortlichkeit in der CRC-Kartenmethode auf UML abgebildet?
- Was steht auf der Vorderseite einer CRC-Karte?

Literatur

[Barnes2012] David J. Barnes and Michael Kölling: Objects First with Java - A Practical Introduction using BlueJ. Fifth edition, Prentice Hall / Pearson Education, 2012

[Ratz2006] Dietmar Ratz, Jens Scheffler, Detlef Seese, Jan Wiesenberger: Grundkurs Programmieren in Java. Carl Hanser Verlag, 3. Auflage, 2006

Ende