02 Objekte und Klassen

Rouven Dreimann, B.Eng

October 16, 2016

Organisatorisches

Anmeldung zur Klausur

Erster Termin

Aushang beachten !!!

Prüfungszeitraum: 23.01.17 - 03.02.2016

Zweiter Termin

Aushang beachten !!!

Abmeldung

bis 8 Tage vor der Klausur möglich! => !Prüfungsordnung!

Optimierte Praktikumszeiten

Gilt ab KW 43:

Tragen Sie 2 Wunschzeiten ein

Notenschlüssel Praktikum (120 Punkte maximal)

- $>= 110 := 1.0$
- $>= 105 := 1.3$
- $>= 100 := 1.7$
- $>= 095 := 2.0$
- $>= 090 := 2.3$
- $>= 085 := 2.7$
- $>= 080 := 3.0$
- $>= 075 := 3.3$
- $>= 070 := 3.7$
- $>= 065 := 4.0$
- $< 060 := 5.0$

Inhalt

Inhaltsverzeichnis

Lernziele

- Erklären können, was ein Objekt von einer Klasse unterscheidet?
- Verstehen, wie Objekte kommunizieren?
- Zusammenfassen, was Attribute, Methoden und Konstruktoren sind?
- Strukturieren, von Klassen in Paketen?
- Schreiben, von einfachsten Javaprogrammen in Eclipse
- Ausführen dieser Programme

Modell

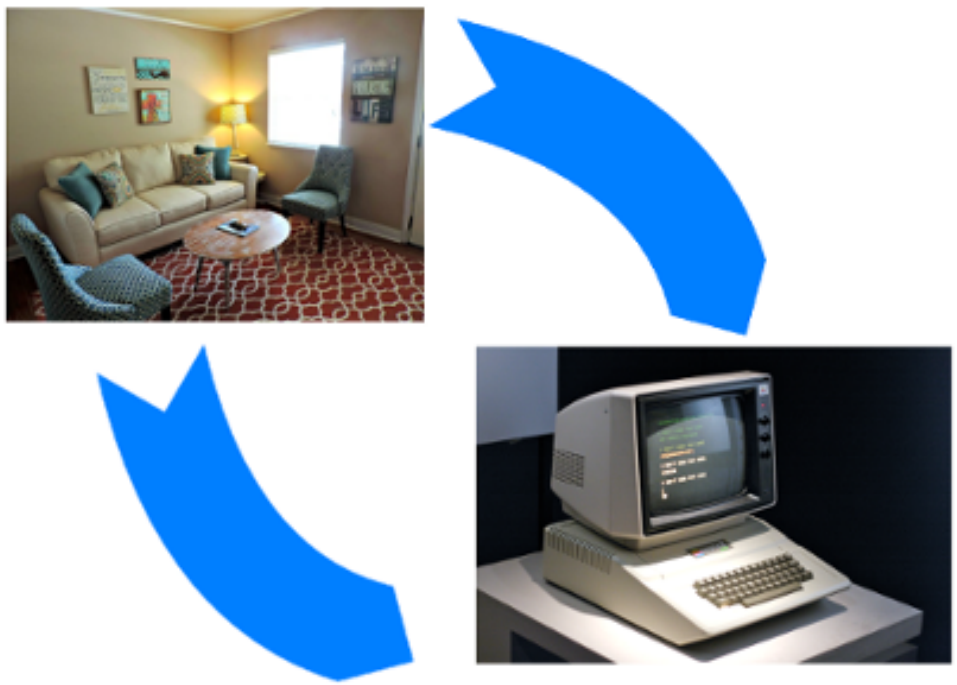
Modell und Wirklichkeit

Ein Modell ist ein beschränktes Abbild der Wirklichkeit.

1. Abbildung – Ein Modell ist stets ein Modell von etwas, nämlich Abbildung, Repräsentation eines natürlichen oder eines künstlichen Originals, das selbst wieder Modell sein kann.
2. Verkürzung – Ein Modell erfasst im Allgemeinen nicht alle Attribute des Originals, sondern nur diejenigen, die dem Modellschaffer bzw. Modellnutzer relevant erscheinen.
3. Pragmatismus – Modelle sind ihren Originalen nicht eindeutig zugeordnet. Sie erfüllen ihre Ersetzungsfunktion a) für bestimmte Subjekte (Für Wen?), b) innerhalb bestimmter Zeitintervalle (Wann?) und c) unter Einschränkung auf bestimmte gedankliche oder tätliche Operationen (Wozu?).

Modell und Wirklichkeit

Ziel:
Ein Modell der Wirklichkeit mittels objektorientierten Konzepten im Rechner abbilden



Objekte

Was ist ein Objekt

- Ein Objekt hat eine eindeutige Identität
- Identität

Zustand

- Der Zustand eines Objekts ist über die Menge der `Attribute` definiert
- Attribute sind Eigenschaften eines Objekts
- Zustände sind veränderbar

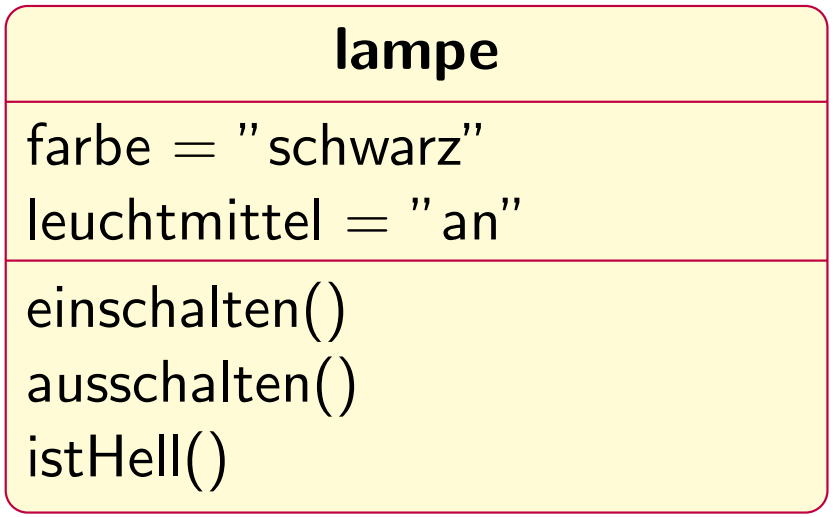
- Identität
- Zustand

lampe
farbe = "schwarz"
leuchtmittel = "an"

Verhalten

- Ein Objekt ist eine Einheit aus Daten (Attributen) und Operationen (Methoden)
- Objekte ändern ihre Attribute über Methoden
- Die Methoden bilden dabei die Schnittstelle nach außen

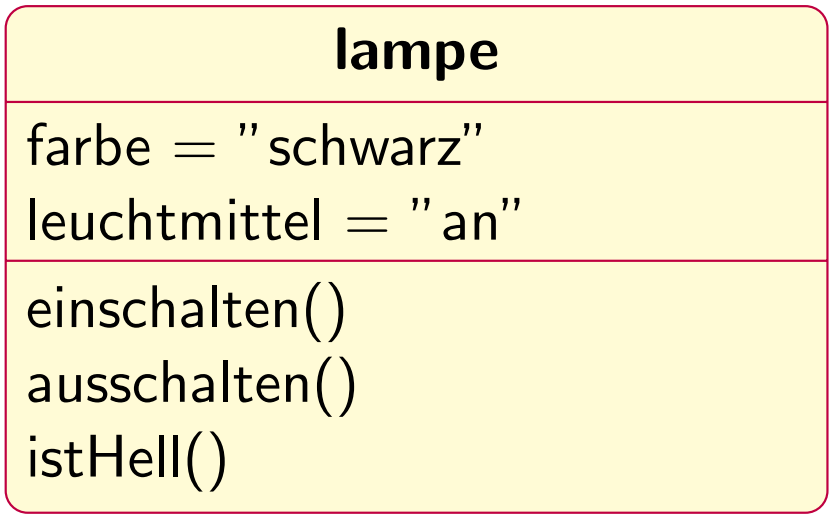
- Identität
- Zustand
- Verhalten



Datenkapselung

- Objekte kapseln ihre Attribute nach außen
- Ein Objekt hält die Hoheit über seine Daten
- Diese Aufteilung ist bei großer Software enorm hilfreich

- Identität
- Zustand
- Verhalten
- Kapselung von Daten



Identität 2

- Objekte sind paarweise verschieden, auch bei gleichem `Zustand` und gleichem `Verhalten`!
- Die Identität ist vom Zustand und vom Verhalten `UNABHÄNGIG`

lampe01
farbe = "schwarz" leuchtmittel = "an"
einschalten() ausschalten() istHell()

lampe02
farbe = "weiß" leuchtmittel = "aus"
einschalten() ausschalten() istHell()

lampe03
farbe = "schwarz" leuchtmittel = "an"
einschalten() ausschalten() istHell()

Lebensdauer von Objekten

- werden nach Programmstart erzeugt
- sterben spätestens bei Programmende (zumindest vorerst für uns)
- Objekte sind im flüchtigen Speicher (RAM) vorhanden

Klassen

Vom Objekt zur Klasse

lampe01
farbe = "schwarz" leuchtmittel = "an"
einschalten() ausschalten() istHell()

lampe02
farbe = "weiß" leuchtmittel = "aus"
einschalten() ausschalten() istHell()

lampe03
farbe = "schwarz" leuchtmittel = "an"
einschalten() ausschalten() istHell()

lampe04
farbe = "schwarz" leuchtmittel = "aus"
einschalten() ausschalten() istHell()

Klasse

Eine Klasse:

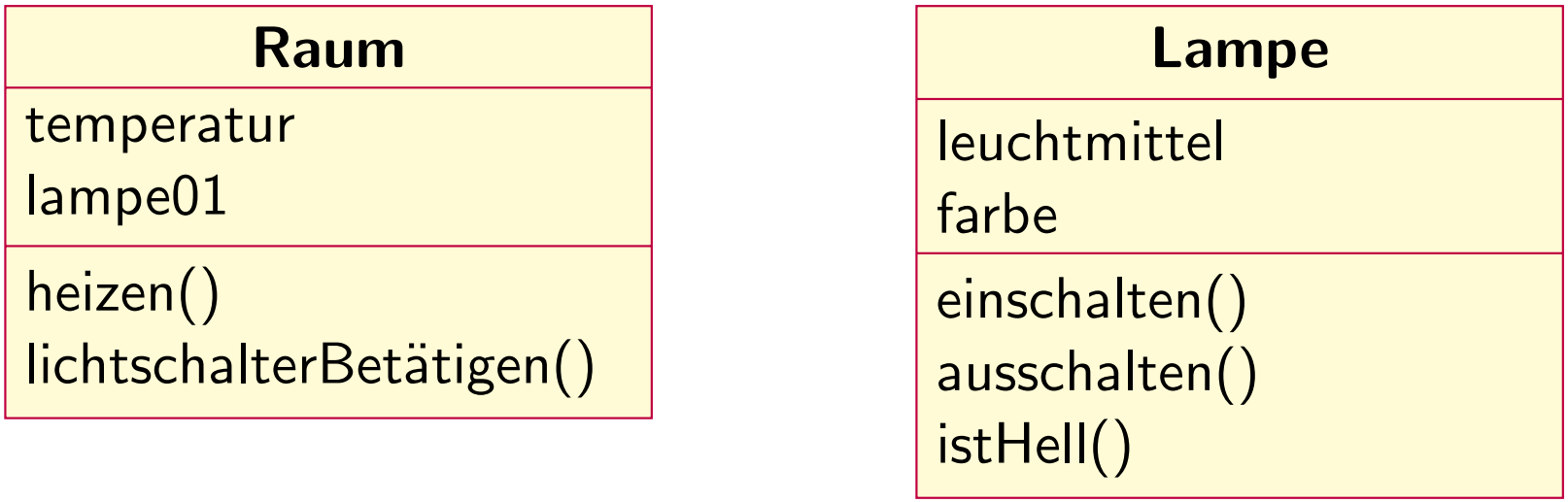
- ist eine abstrakte Beschreibung, quasi der Bauplan, eines oder mehrerer Objekte (`Datentyp`)
- definiert die möglichen Zustände der Objekte
- legt das Verhalten der Objekte durch Methoden fest

Klassendefinition

- Klassennamen werden groß geschrieben
- Attribute definieren die möglichen Objektzustände
- Methoden definieren wie reagiert wird

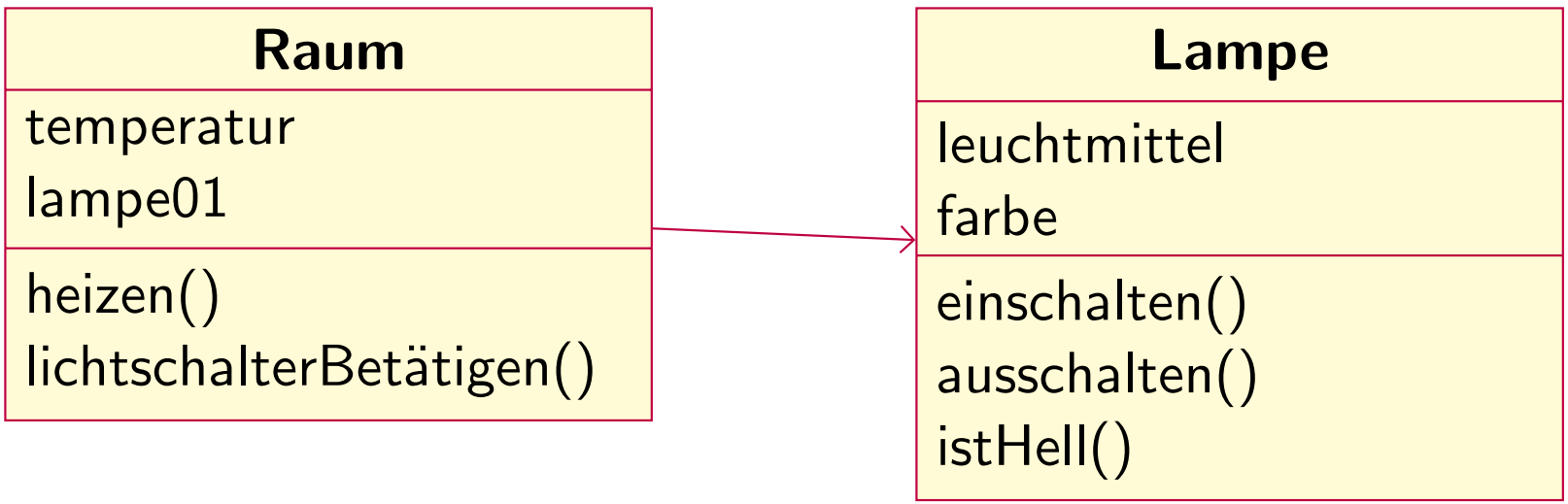
Klassendiagramme

- Objekte des Typs *Raum* wollen Methoden vom Typ *Lampe* aufrufen



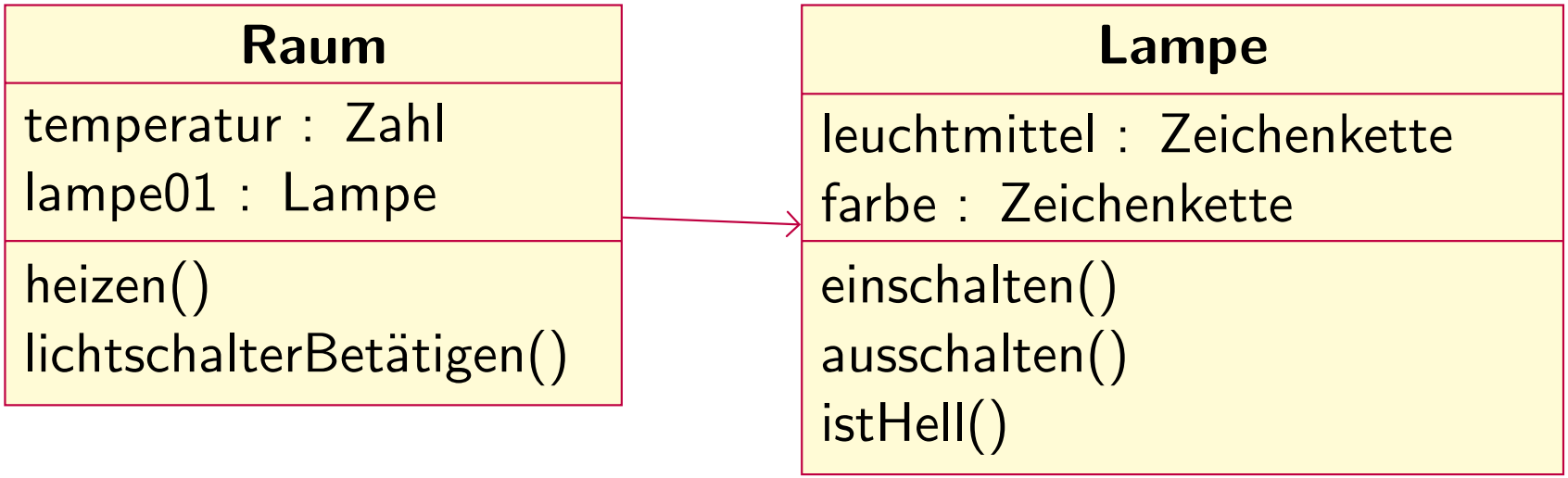
Klassendiagramme

- Klasse *Raum* kennt die Klasse *Lampe*
- Objekte vom Typ `Raum` können Methoden von Lampen-Objekten aufrufen
- Klasse *Lampe* kennt die Klasse *Raum* nicht!
- *Lampen*-Objekte können keine Methoden von *Raum*-Objekten aufrufen



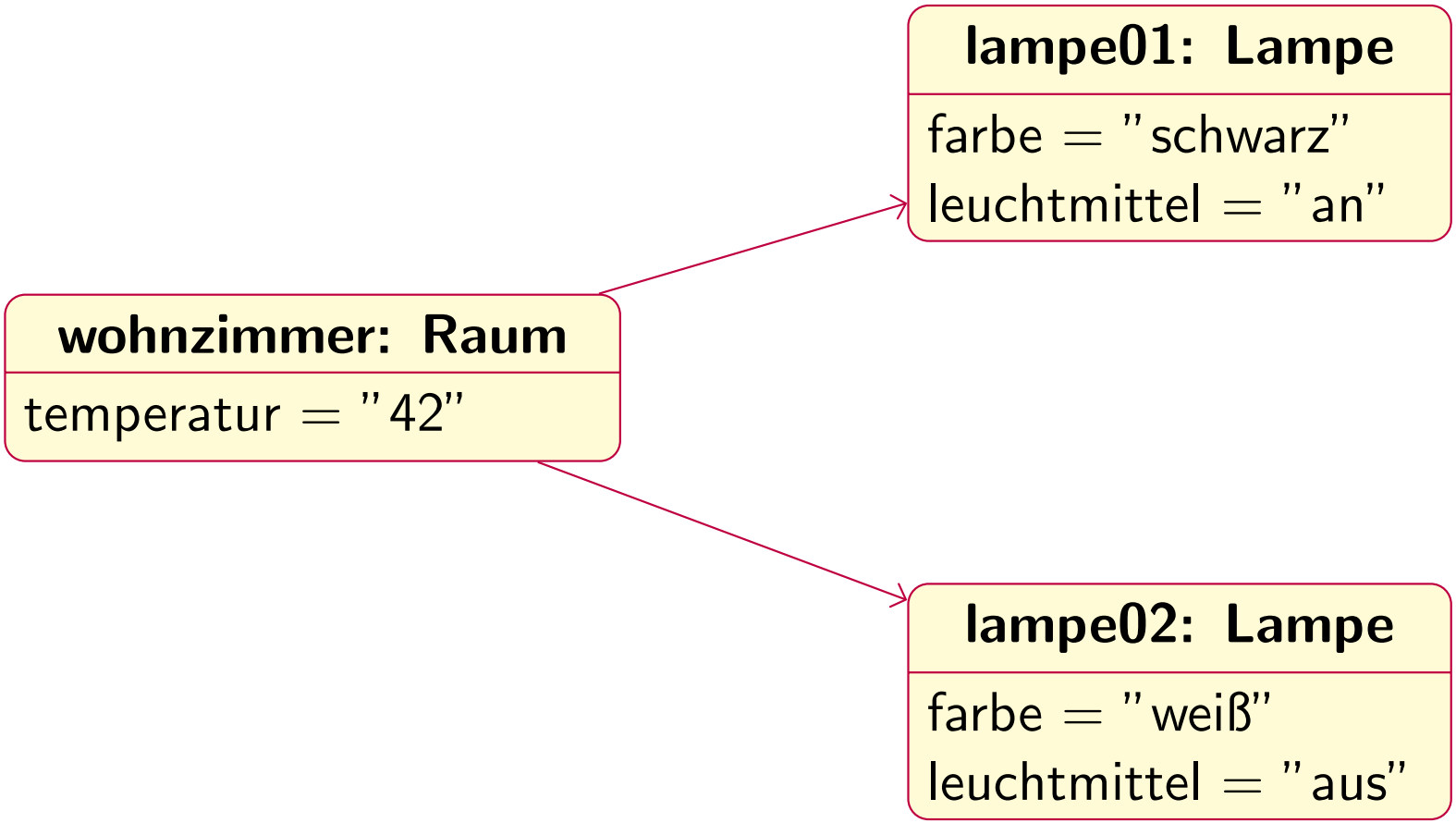
Klassendiagramme

- Attribute sind von einem speziellen Datentyp
- Attribute sind wiederrum Objekte



Objektdiagramm

- instanziierte Objekte aus dem Klassendiagramm

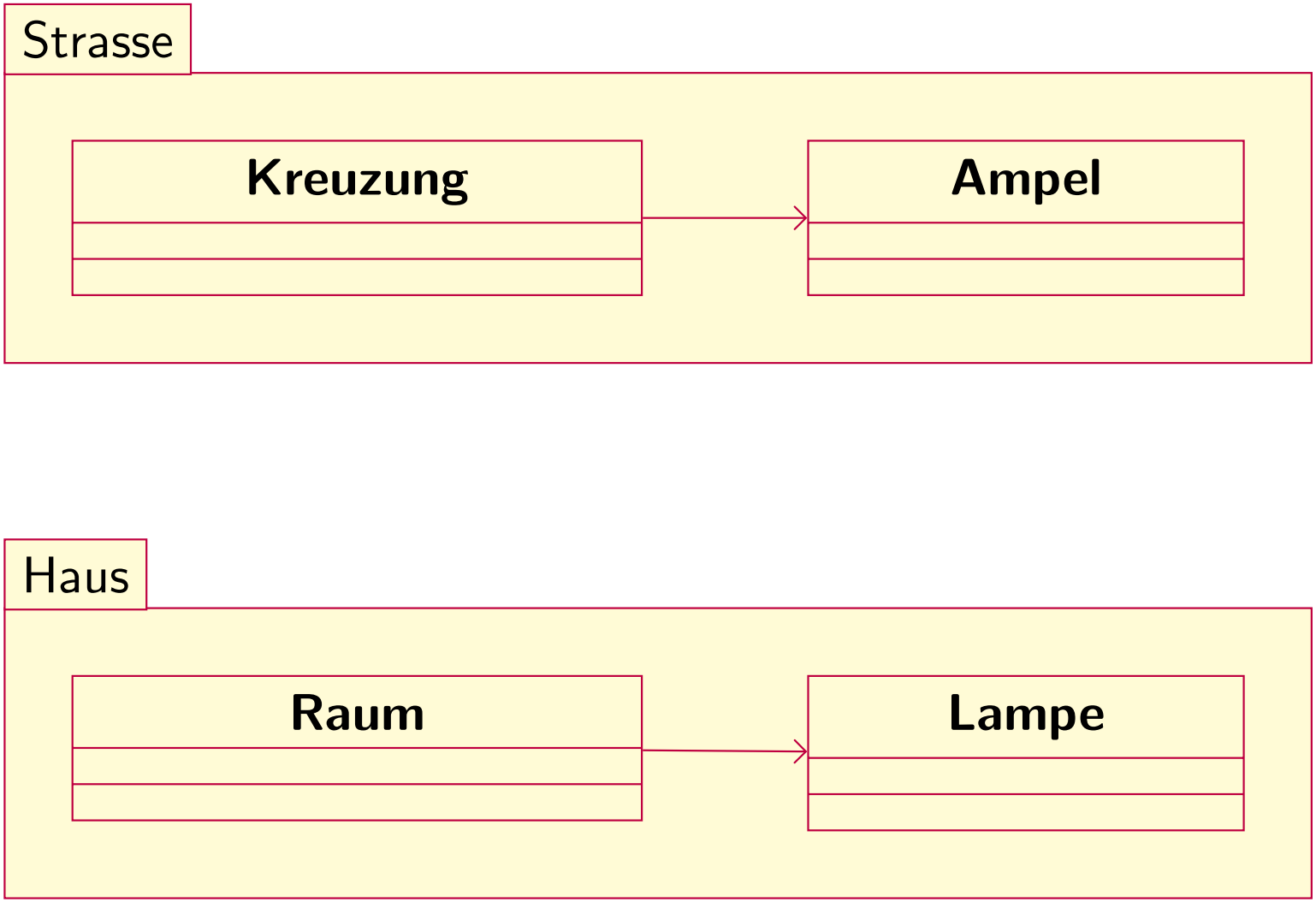


Pakete

Package

- Große Software erfordert weitere Unterteilungsstufen
- Mehrere tausend Klassen sind denkbar
- Paket - Klasse <=> Ordner - Datei
- Pakete können weitere Pakete enthalten

Packages in Klassendiagrammen



Zugriffsmodifikatoren

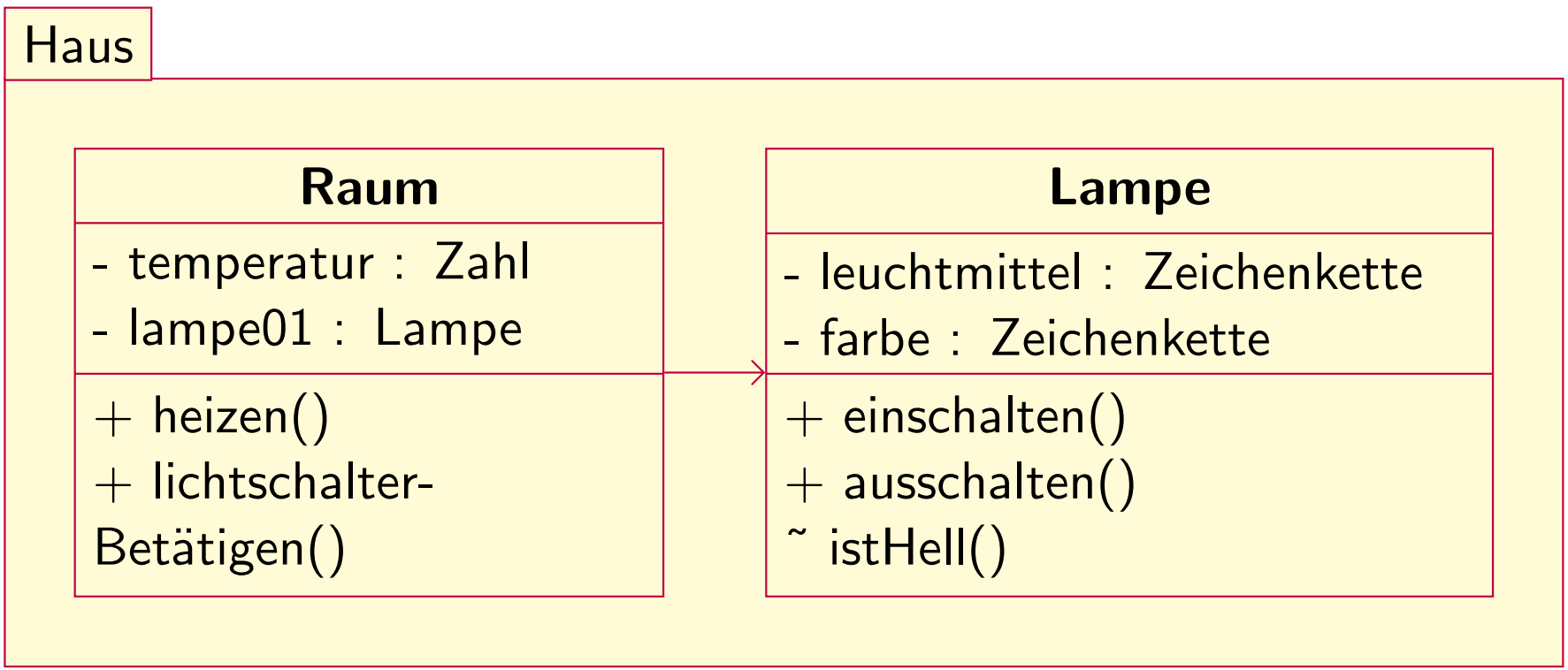
Sichtbarkeiten

Die Sichtbarkeit von Klassen, Attributen und Methoden kann eingeschränkt werden

- public (keine Einschränkung, überall sichtbar)
- default (package genannt, nur sichtbar im gleichen Paket, Standard in Java)
- private (nur sichtbar innerhalb der Klasse)
- protected (kommt später)

Sichtbarkeiten im Klassendiagramm

- + public (Methoden und Klassen sind in der Regel public)
- - private (Attribute sind in der Regel private)
- ~ package (wird selten benötigt)
- # protected (kommt später)



Implementierung in Java

Programmieren in Java

- In Java werden Klassen beschrieben (`deklariert`)
- Aus Klassen werden nach Programmstart Objekte gebaut (`instanziiert`)
- Java liefert Klassen, die häufig gebraucht werden
- Java-Befehle sind englisch

```
package haus;

class Lampe {

}
```

Vordefinierte Datentypen

Java liefert Klassen für häufig gebrauchte Datentypen:

- Object (dazu später mehr)
- Integer (ganze Zahlen [-123, 645231])
- Double (Fließkommazahlen [1.234*10^12, 3.141592])
- String (Zeichenketten [“ABC..abc..1234..!§”])
- Boolean (Wahrheitswerte [true,false])
- ...

Attribute

- Attribute haben einen Typ, Namen und Wert
- Attribute können auf Objekte verweisen

```
package haus;

class Lampe {
    String farbe = "schwarz";
    String leuchtmittel = "aus";
}
```

Methoden

- Methoden definieren das Verhalten eines Objekts
- Methoden können einen Wert, eines bestimmten Datentyps zurückliefern

```
package haus;
class Lampe {
    String farbe = "schwarz";
    String leuchtmittel = "aus";

    void einschalten() {
    }
    void ausschalten() {
    }
    void istHell() {
    }
}
```

Konstruktoren

- Konstruktoren sind Methoden, die ein Objekt erstellen

```
package haus;
class Lampe {
    String farbe = "schwarz";
    String leuchtmittel = "aus";

    Lampe() {
    }

    void einschalten() {
        leuchtmittel = "an";
}
```

Klassen in Java schreiben

Was ist was?

- Klasse
- Attribut
- Methode
- Konstruktor

```
package haus;
class Lampe {
    String farbe = "schwarz";
    String leuchtmittel = "aus";
    Lampe () {
    }
    void einschalten() {
        leuchtmittel = "an";
    }
    void ausschalten() {
        leuchtmittel = "aus";
    }
    void istHell() {
        System.out.println(
            "die Lampe ist " + leuchtmittel);
    }
}
```

Klassen mit new instanziieren

- Objekte werden durch spezielle Methoden (Konstruktoren) gebaut (instanziiert)
- Konstruktoren haben den gleichen Namen, wie die Klasse
- ein Objekt wird mit dem `new`-Operator `instanziiert`

```
Lampe lampe01 = new Lampe();
```

Programmstarter

- `public static void main(String[] args)` wird später erklärt
- main() -> Startpunkt vom Programm

```
package paket;
class Start {

    public static void main(String[] args) {
        Lampe lampe01 = new Lampe();
    }
}
```

Klassen in Java schreiben

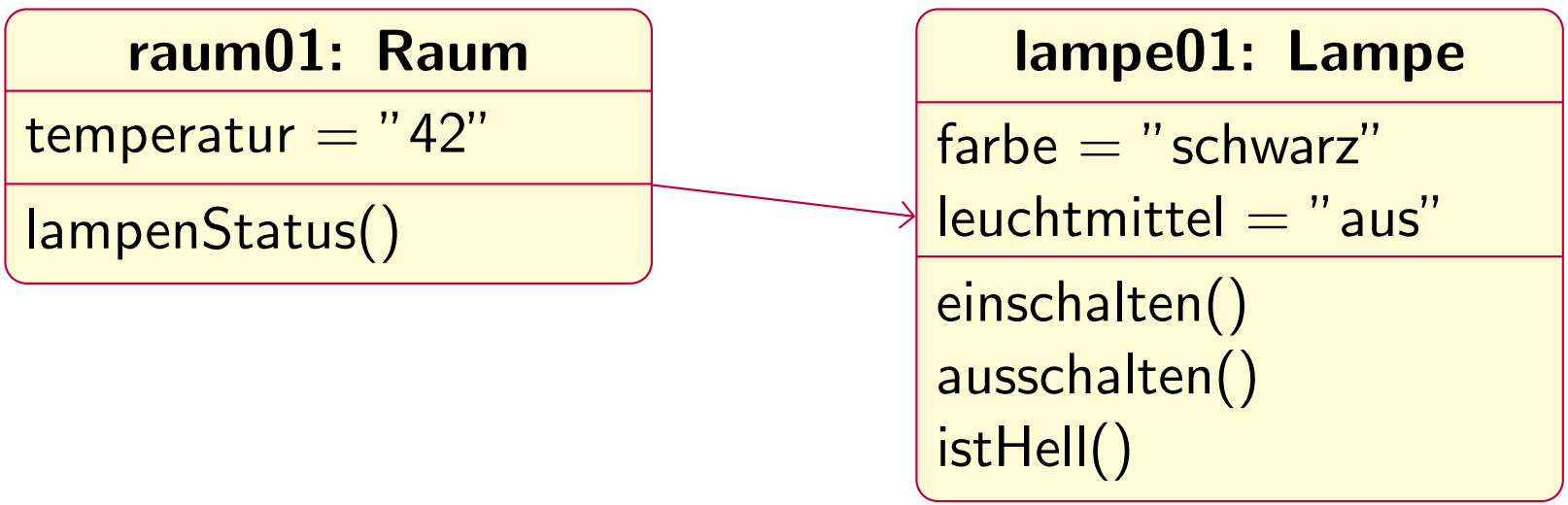
```
package haus;
class Raum {
    private Integer temperature;
    private Lampe lampe01;

    public Raum() {
        temperature = 42;
        lampe01 = new Lampe();
    }
    public void lampenStatus() {
        lampe01.istHell();
    }
    public static void main(String[] args) {
        Raum raum01 = new Raum();
        raum01.lampenStatus();
    }
}
```

Methodenaufruf - Nachrichten

```
void lampenStatus() {
    lampe01.istHell();
}
```

- raum01 sendet eine Nachricht an lampe01
- lampe01 reagiert, indem die Methode istHell() ausgeführt wird



Let's get our hands dirty...

Lessons Learned

Lernziele erreicht?

- Was ist ein Objekt / eine Klasse ?
- Wie kommunizieren Objekte?
- Was sind Attribute / Methoden / Konstruktoren?
- Wie schreibe ich ein Java-Programm mit Eclipse?
- Wie führe ich ein Java-Programm aus?

Außerdem...

- Was ist ein Klassendiagramm?
- Was ist ein Objektdiagramm?
- Wie können Klassen strukturiert werden?
- Wie können Sichtbarkeiten beeinflusst werden?