

Hochschule für Technik und Wirtschaft Berlin

- wir entscheiden uns daher, das Auszahlungsdatum nur in Textform zur Verfügung zu stellen
 - und zwar in diesem Format: tt.mm.jjjj
 - das können wir erzeugen, indem wir uns mit der Methode get der Klasse GregorianCalendar den Tag, den Monat und das Jahr besorgen und dann die zu liefernde Zeichenkette so zusammenstellen:

Hochschule für Technik und Wirtschaft Berlin

Klassen

Klassen

- was ist eine statische Methode?
 - eine Methode, die mit dem Methodenmodifizierer static versehen wurde
- statische Methoden sind der Ersatz für die in Java nicht vorhandenen globalen Funktionen
- sie k\u00f6nnen benutzt werden, ohne zuvor ein Objekt der Klasse erzeugen zu m\u00fcssen
 - > und zwar nach folgendem Schema:

KlassenName.machWas()

konkret:

String.format("%1\$td.%1\$tm.%1\$tY", startdatum)

© H. Brandenburg Programmierung 2

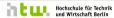


- weil statische Methoden nicht Fähigkeiten von Objekten realisieren, werden sie gelegentlich auch als Klassenmethoden bezeichnet
 - > im Gegensatz zu den gewöhnlichen Objektmethoden
- Achtung: da statische Methoden nicht der Grundidee der Objektorientierung entsprechen, sollten Methoden nur in begründeten Ausnahmefällen als static deklariert werden
 - dabei ist zu beachten:
 - statische Methoden können nur auf Attribute zugreifen, die static sind
 - benutzen sie andere Methoden der Klasse, müssen diese ebenfalls static sein

© H. Brandenburg

Programmierung 2

97



Klassen

 wir lösen das Problem, indem wir zuerst eine lokale Kopie des Startdatums erzeugen und dann diese benutzen:

 Hinweis: das Erzeugen einer Kopie geht auch einfacher, worauf wir aber erst später eingehen

© H. Brandenburg Programmierung 2 99



Klassen

- die Methode liefereTilgungsEnde unserer Klasse AnnuitaetenDarlehen stellt uns vor ein neues Problem:
 - wir müssen die Laufzeit zum Startdatum addieren
 - die Methode add der Klasse GregorianCalendar ist zwar in der Lage, zu einem Datum Tage, Monate oder Jahre zu addieren, sie verändert dabei aber den Zustand des Objektes
 - würden wir sie für unser Attribut startdatum benutzen, um das Tilgungsende zu ermitteln, würde sich das Startdatum verändern

© H. Brandenburg

Programmierung 2

30



Klassen

 die Methode liefereJaehrlicheRate deklarieren wir so:

```
public double liefereJaehrlicheRate()
{
   return liefereAnnuitaet();
}
```

 dabei ist liefereAnnuitaet eine private Hilfsmethode, die wir mehrfach benutzen

© H. Brandenburg

Programmierung 2

100



 die nächsten beiden Methoden benutzen jeweils andere Methoden unserer Klasse:

```
public double liefereZinsenBeiJaehrlicherRate()
{
   return liefereGesamtAufwandBeiJaehrlicherRate() - betrag;
}

public double liefereGesamtAufwandBeiJaehrlicherRate()
{
   return laufzeit * liefereAnnuitaet();
}
```

- die Berechnung der monatlichen Rate ist etwas komplizierter
 - > sie verlangt finanzmathematisches Know How

© H. Brandenburg

Programmierung 2

101



Klassen

- dabei kann es anfänglich vorkommen, dass die Restschuld rechnerisch wächst
- die Einzelheiten k\u00f6nnen folgender Methode entnommen werden:

```
public double liefereZinsenBeiMonatlicherRate()
{
   double zinsenGesamt = 0;
   double zinsenImMonat = 0;
   double zinsenImMonat = 0;
   double zinsenImJahr = 0;
   double monatlicheRate = liefereMonatlicheRate();
   double restschuld = betrag;
   for (int jahr = 1; jahr <= laufzeit; jahr++)
   {
      for (int monat = 1; monat <= 12; monat++)
      {
        zinsenImMonat = (restschuld / 100 * zinssatz) / 12;
        zinsenImJahr += zinsenImMonat;
</pre>
```

```
Hochschule für Technik und Wirtschaft Berlin Klassen

public double liefereMonatlicheRate()
{
  int ratenProJahr = 12;
  double zinsfaktor = zinssatz / 100;
  double nenner =
        ratenProJahr + ((zinsfaktor / 2.0) * (ratenProJahr - 1));
  return liefereAnnuitaet() / nenner;
}
```

- das gilt auch für die Berechnung der insgesamt zu zahlenden Zinsen bei monatlicher Ratenzahlung
 - es ist üblich, die ersten 11 Raten eines Jahres als reine Tilgung zu betrachten
 - die zu zahlenden Zinsen werden kumuliert und am Ende des Jahres mit der letzten Rate verrechnet

© H. Brandenburg

Programmierung 2

102

```
if (monat != 12)
    restschuld -= monatlicheRate;
else
{
    if (zinsenImJahr >= monatlicheRate)
        restschuld += (zinsenImJahr - monatlicheRate);
else
        restschuld -= (monatlicheRate - zinsenImJahr);
}
zinsenGesamt += zinsenImJahr;
zinsenImJahr = 0;
}
return zinsenGesamt;
}
```

 die Berechnung des Gesamtaufwandes bei monatlicher Ratenzahlung ist vergleichsweise einfach:

© H. Brandenburg

Programmierung 2

104

```
Hochschule für Technik
und Wirtschaft Berlin

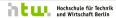
Public double liefereGesamtAufwandBeiMonatlicherRate()
{
    return laufzeit * 12 * liefereMonatlicheRate();
}
```

- damit ist die Deklaration der Klasse
 AnnuitaetenDarlehen abgeschlossen
 - gegenüber dem UML-Entwurf ist die private Hilfsmethode liefereAnnuitaet hinzugekommen
 - dass private Hilfsmethoden erst beim Implementieren auftauchen, ist typisch
- wie empfehlen, in Klassendeklarationen immer zuerst alle öffentlichen Methoden anzugeben (d.h. das API) und erst danach die privaten Methoden

© H. Brandenburg

Programmierung 2

105



Klassen

- um die Klasse AnnuitaetenDarlehen nutzen zu können, schreiben wir jetzt eine Klasse AnnuitaetenDarlehenMain
- dabei ergibt sich sofort ein inhaltliches Problem:
 - AnnuitaetenDarlehenMain soll eigentlich nur als Einstiegspunkt eines Programms dienen
 - weil aber ein Java-Programm nur aus Klassen und Interfaces besteht, muss es eine Klasse sein
 - welche Gestalt und welche F\u00e4higkeiten sollen die Objekte dieser Klasse haben?

© H. Brandenburg Programmlerung 2



Klassen

- · zusammenfassend stellen wir fest:
 - nach der Erzeugung befinden sich die Objekte der Klasse AnnuitaetenDarlehen stets in einem sinnvollen Zustand
 - alle Methoden des API überführen die Objekte immer von einen sinnvollen Zustand in einen sinnvollen Zustand
 - · hier, weil sie sie unverändert lassen
 - alle Methoden des API können unabhängig voneinander in beliebiger Reihenfolge benutzt werden
- diese Gütekriterien muss jede Klasse erfüllen!

© H. Brandenburg

Programmierung 2



Klassen

- in C++ stellt sich dieses Problem nicht
 - denn C++-Programme werden wie C-Programme mit Ausführung der globalen Funktion main gestartet
- als Lösung des Problems bei Java empfehlen wir, einfach die in C++ gegebene Situation wie folgt zu imitieren
 - AnnuitaetenDarlehenMain erhält keine Attribute
 - wir sorgen dafür, dass keine Objekte der Klasse erzeugt werden können
 - indem wir genau einen Konstruktor vorsehen und diesen als private deklarieren

© H. Brandenburg

Programmierung 2

108



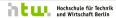
- main ist (zunächst) die einzige Methode der Klasse
 - Java schreibt vor, das main immer public und static ist
- stellt sich heraus, dass die Aufgaben von main komplex sind, setzen wir zusätzlich geeignete Hilfsmethoden ein
 - die dann stets private und static sind
- dadurch wird AnnuitaetenDarlehenMain Zu einer aus objektorientierter Sicht völlig untypischen Klasse, die natürlich die genannten Gütekriterien nicht erfüllt
 - diese Häßlichkeit nehmen wir in Kauf

© H. Brandenburg

Programmierung 2

109

111



Klassen

- es besteht allerdings aus ca. 50 Klassen, so dass es für Anfänger relativ schwer zu verstehen ist
- wir müssen daher anfänglich einige bewährte Dinge übernehmen, ohne genau zu verstehen, was sie bewirken
 - später werden wir das Ein- und Ausgabesystem systematisch studieren
- für die Ausgabe auf dem Bildschirm gibt es das Attribut out der Klasse java.lang.System

Hochschule für Technik und Wirtschaft Berlin

Klassen

- ein weiteres Problem ergibt sich daraus, dass das Programm selbstverständlich Ausgaben erzeugen soll
 - wir müssen uns daher zuerst noch mit der Ein- und Ausgabe in Java beschäftigen
- wie in C/C++ ist die Ein- und Ausgabe in Java nicht Teil der Sprache, sondern über die Java-Klassenbibliothek geregelt
- es gibt ein selbstverständlich objektorientiertes – Ein- und Ausgabesystem, das äußerst vielseitig und flexibel ist

© H. Brandenburg

Programmierung 2

110



Klassen

- weil es public und static ist, kann es direkt als System.out angesprochen werden
- System.out ist ein Objekt der Klasse java.io.PrintStream
- PrintStream stellt u.a. folgende Methoden zur Ausgabe auf dem Bildschirm zur Verfügung:

```
print
println
printf
```

© H. Brandenburg

Programmierung 2

112

© H. Brandenburg

Programmierung 2



- für die formatierte Ausgabe auf dem Bildschirm ist printf geeignet
 - die Methode wurde erst mit der J2SE 5.0 in Java eingeführt

Hinweis: die formatierte Ausgabe auf dem Bildschirm muss daher noch auf anderem Wege möglich sein

- sie ist der entsprechenden Funktion aus C nachempfunden, aber nicht identisch mit ihr
 - die Details zu printf sollten Sie sich im Selbststudium aneignen
- nach diesen Vorüberlegungen gestalten wir AnnuitaetenDarlehenMain SO:

© H. Brandenburg

Programmierung 2

113

```
Hochschule für Technik
                                                                        Klassen
    System.out.printf("%-45s%s\n", "Ende der Laufzeit:",
                      darlehen.liefereTilgungsEnde());
    System.out.printf("%-40s%10.2f %s\n", "Zinssatz",
                      darlehen.liefereZinssatz(), "%");
    System.out.println();
    System.out.printf("%-40s%10.2f Euro\n", "Jaehrliche Rate:",
                      darlehen.liefereJaehrlicheRate());
    System.out.printf("\$-40s\$10.2f \ Euro\n", \ "Zinsen \ (jaehrliche \ Rate):",
                      darlehen.liefereZinsen());
    System.out.printf("%-40s%10.2f Euro\n",
                      "Gesamtaufwand (jaehrliche Rate):",
                      darlehen.liefereGesamtAufwandBeiJaehrlicherRate());
    System.out.println();
    System.out.printf("%-40s%10.2f Euro\n", "Monatliche Rate:",
                      darlehen.liefereMonatlicheRate());
    System.out.printf("%-40s%10.2f Euro\n", "Zinsen (monatliche Rate):",
                      darlehen.liefereZinsenBeiMonatlicherRate());
    System.out.printf("%-40s%10.2f Euro\n",
                      "Gesamtaufwand (monatliche Rate):",
                      darlehen.liefereGesamtAufwandBeiMonatlicherRate());
    System.out.println();
© H. Brandenburg
                                 Programmierung 2
```

```
Hochschule für Technik und Wirtschaft Berlin
                                                                         Klassen
public class AnnuitaetenDarlehenMain
  private AnnuitaetenDarlehenMain()
  public static void main(String[] args)
    System.out.println("\nDarlehen 1:\n");
    AnnuitaetenDarlehen kredit1 = new AnnuitaetenDarlehen(100000, 16, 8.5);
    schreibeKreditdaten(kredit1);
    System.out.println("\nDarlehen 2:\n");
    AnnuitaetenDarlehen kredit2 = new AnnuitaetenDarlehen(5000, 2, 3.87);
    schreibeKreditdaten(kredit2);
  private static void schreibeKreditdaten(AnnuitaetenDarlehen darlehen)
    System.out.printf("%-40s%10.2f Euro\n", "Kredithoehe:",
                       darlehen.liefereDarlehensBetrag());
    System.out.printf("%-40s%10d Jahre\n", "Laufzeit",
                       darlehen.liefereLaufzeit()):
    System.out.printf("%-45s%s\n", "Beginn der Laufzeit:",
                       darlehen.liefereAuszahlungsDatum());
© H. Brandenburg
                                  Programmierung 2
```

Hochschule für Technik Klassen das Programm gibt auf dem Bildschirm (untereinander) aus: Darlehen 1: Kredithoehe: 100000,00 Euro Laufzeit 16 Jahre Beginn der Laufzeit: 23.10.2010 23.11.2026 Ende der Laufzeit: Zinssatz 8,50 % Jaehrliche Rate: 11661,35 Euro Zinsen (jaehrliche Rate): 86581,67 Euro Gesamtaufwand (jaehrliche Rate): 186581,67 Euro Monatliche Rate: 935,34 Euro 79585.33 Euro Zinsen (monatliche Rate): Gesamtaufwand (monatliche Rate): 179585,33 Euro © H. Brandenburg Programmierung 2

```
Hochschule für Technik
                                                                Klassen
    Darlehen 2:
    Kredithoehe:
                                                 5000.00 Euro
    Laufzeit
                                                       2 Jahre
    Beginn der Laufzeit:
                                                   23.10.2010
                                                   23.11.2012
    Ende der Laufzeit:
                                                    3,87 %
    Zinssatz
    Jaehrliche Rate:
                                                 2646,04 Euro
    Zinsen (jaehrliche Rate):
                                                  292.09 Euro
    Gesamtaufwand (jaehrliche Rate):
                                                 5292,09 Euro
    Monatliche Rate:
                                                  216,66 Euro
    Zinsen (monatliche Rate):
                                                  199,85 Euro
    Gesamtaufwand (monatliche Rate):
                                                 5199,85 Euro
© H. Brandenburg
                              Programmierung 2
```

* seit der J2SE 5.0 gibt es die Klasse java.util.Scanner, die wir jetzt in der neu gestalteten Methode main benutzen:

public static void main (String[] args)

(Control public static (No Yearling to the Normal String () args))

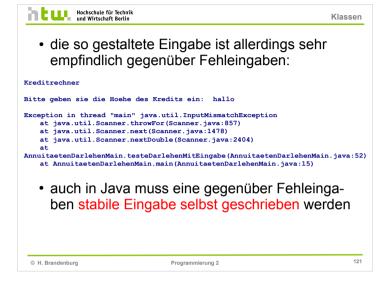
```
System.out.println("\nKreditrechner\n");
   Scanner eingabe = new Scanner(System.in):
   System.out.print("Bitte geben Sie die Hoehe des Kredits ein: ");
   double kreditHoehe = eingabe.nextDouble();
   System.out.print("Bitte geben Sie die Laufzeit des Kredits" +
                     " in Jahren ein: ");
   int laufzeit = eingabe.nextInt();
   System.out.print("Bitte geben Sie den effektiven Jahreszins" +
                    " in Prozent ein: ");
   double zinssatz = eingabe.nextDouble();
   AnnuitaetenDarlehen kredit =
                new AnnuitaetenDarlehen(kreditHoehe, laufzeit, zinssatz);
   System.out.println("\nKredit:\n");
   schreibeKreditdaten(kredit);
                                                                             119
© H. Brandenburg
                                 Programmierung 2
```

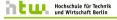


- flexibler wäre es, wenn man die gewünschten Kreditdaten über die Tastatur eingeben könnte
- für die Eingabe über die Tastatur gibt es das Attribut in der Klasse java.lang.System
- weil es public und static ist, kann es direkt als System.in angesprochen werden
- System.in ist ein Objekt der Klasse java.io.InputStream
- es gibt mehrere Möglichkeiten, die Eingabe zu gestalten

© H. Brandenburg Programmierung 2 118

```
Hochschule für Technik
                                                                Klassen
  auf dem Bildschirm wird jetzt z.B. ausgegeben:
 Kreditrechner
 Bitte geben Sie die Hoehe des Kredits ein: 12345,67
 Bitte geben Sie die Laufzeit des Kredits in Jahren ein: 7
 Bitte geben Sie den effektiven Jahreszins in Prozent ein: 5,67
 Kredit:
 Kredithoehe:
                                             12345,67 Euro
 Laufzeit
                                                    7 Jahre
 Beginn der Laufzeit:
                                                23.10.2010
 Ende der Laufzeit:
                                                23.11.2017
 Zinssatz
                                                 5,67 %
 Jaehrliche Rate:
                                              2185,67 Euro
 Zinsen (jaehrliche Rate):
                                             2954,03 Euro
 Gesamtaufwand (jaehrliche Rate):
                                             15299,70 Euro
                                               177,53 Euro
 Monatliche Rate:
 Zinsen (monatliche Rate):
                                              2566,50 Euro
 Gesamtaufwand (monatliche Rate):
                                             14912,17 Euro
                                                                     120
© H. Brandenburg
                              Programmierung 2
```





- die Java-Klassenbibliothek enthält eine Reihe derartiger Klassen
 - zum Beispiel java.lang.Math mit mehr als 30 Methoden zur Mathematik
- wir sammeln unsere selbst geschriebenen Methoden für die Eingabe über die Tastatur in der Klasse MeineEingabe
- dabei machen wir davon Gebrauch, dass nicht nur Konstruktoren, sondern auch Methoden überladen werden können, d.h. verschiedene Varianten ein und derselben Methode können denselben Namen erhalten

123 © H. Brandenburg Programmierung 2



Klassen

- das führt uns zu einem dritten Typ von Klassen:
 - Sammlungen allgemein wiederverwendbarer Dienstleister
- die Methoden derartiger Klassen stehen in engem inhaltlichen Zusammenhang und sind alle public und static
- falls erforderlich, kann es Attribute geben, die dann ebenfalls static sind
- von derartigen Klassen können keine Objekte erzeugt werden

© H. Brandenburg

Programmierung 2

Hochschule für Technik

Klassen

> diese äußerst nützliche Eigenschaft wird in Java (und auch in C++) sehr oft benutzt

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;
public class MeineEingabe
 private static BufferedReader eingabe =
            new BufferedReader(new InputStreamReader(System.in));
  private MeineEingabe()
  public static String erfasseText()
   return erfasseText("");
```

© H. Brandenburg

Programmierung 2

```
Hochschule für Technik und Wirtschaft Berlin
                                                                      Klassen
   public static String erfasseText(String eingabeAufforderung)
      System.out.print(eingabeAufforderung);
      String text = "":
     boolean eingabeOk = false;
      while (!eingabeOk)
        try
          text = eingabe.readLine();
          eingabeOk = true;
        catch (IOException ioe)
      return text;
   public static int erfasseInt()
     return erfasseInt("");
© H. Brandenburg
                                Programmierung 2
```

```
public static int erfasseInt(String eingabeAufforderung)
{
  int wert = 0;
  boolean eingabeOk = false;
  while (!eingabeOk)
    try
    {
     wert = Integer.parseInt(erfasseText(eingabeAufforderung));
     eingabeOk = true;
    }
    catch (NumberFormatException nfe)
    {
      }
     return wert;
}
```

 mit Hilfe dieser Eingabemethoden gestalten wir die Methode main unserer Klasse

Klassen

AnnuitaetenDarlehenMain jetzt so:

Hochschule für Technik und Wirtschaft Berlin

```
public static void main(String[] args)
   System.out.println("\nKreditrechner\n");
   System.out.println("Bitte geben Sie ein:\n");
   String info = "Hoehe des Kredits [1.00 - 1000000000.00]:
   double kreditHoehe = MeineEingabe.erfasseDouble(info);
   info = "Laufzeit des Kredits in Jahren [1 - 100]:
   int laufzeit = MeineEingabe.erfasseInt(info);
   info = "Effektiver Jahreszins in Prozent [0.1 - 30.0]: ";
   double zinssatz = MeineEingabe.erfasseDouble(info);
   AnnuitaetenDarlehen kredit = new AnnuitaetenDarlehen(kreditHoehe,
                                                         laufzeit,
                                                         zinssatz);
   System.out.println("\nKredit:\n");
   schreibeKreditdaten(kredit);
© H. Brandenburg
                                 Programmierung 2
```

```
Hochschule für Technik
                                                                Klassen
 Kreditrechner
 Bitte geben Sie ein:
 Hoehe des Kredits [1.00 - 1000000000.00]:
                                                  6543.21
 Laufzeit des Kredits in Jahren [1 - 100]:
 Effektiver Jahreszins in Prozent [0.1 - 30.0]: 4.56
 Kredit:
                                              6543.21 Euro
 Kredithoehe:
                                                   5 Jahre
 Laufzeit
 Beginn der Laufzeit:
                                               25.10.2010
 Ende der Laufzeit:
                                               25.11.2015
                                                4.56 %
 Zinssatz
 Jaehrliche Rate:
                                             1492,98 Euro
                                              921,70 Euro
 Zinsen (jaehrliche Rate):
 Gesamtaufwand (jaehrliche Rate):
                                             7464,91 Euro
                                              121,87 Euro
 Monatliche Rate:
 Zinsen (monatliche Rate):
                                              768,87 Euro
 Gesamtaufwand (monatliche Rate):
                                              7312,08 Euro
© H. Brandenburg
                              Programmierung 2
```



 bei größeren Programmen kann es mehrere Klassen dieser Art geben

Achtung: sie müssen aber ausschließlich allgemein wiederverwendbare Methoden enthalten, keinesfalls applikationsspezifische Methoden!

- warum haben wir das Attribut eingabe unserer Klasse MeineEingabe als static deklariert?
 - formal: weil die Methode erfasseText static ist und (als einzige der Klasse MeineEingabe!) direkt auf eingabe zugreift

© H. Brandenburg Programmierung 2 131



Klassen

- unser Programm ist jetzt so gestaltet, wie (bei uns) alle Java-Programme aussehen sollen:
- es besteht aus
 - der aus objektorientierter Sicht völlig untypischen Klasse AnnuitaetenDarlehenMain, die als Einstiegspunkt in das Programm dient
 - einer Klasse AnnuitaetenDarlehen, deren Objekte "Experten" sind
 - bei größeren Programmen wird es viele dieser aus objektorientierter Sicht typischen Klassen geben
 - einer aus objektorientierter Sicht untypischen Klasse mit allgemein wiederverwendbaren Methoden

© H. Brandenburg

Programmierung 2

130



Klassen

- warum haben wir eingabe überhaupt als Attribut deklariert?
 - die Methode würde exakt dasselbe leisten, wenn wir eingabe zu einer lokalen Variablen machen würden: