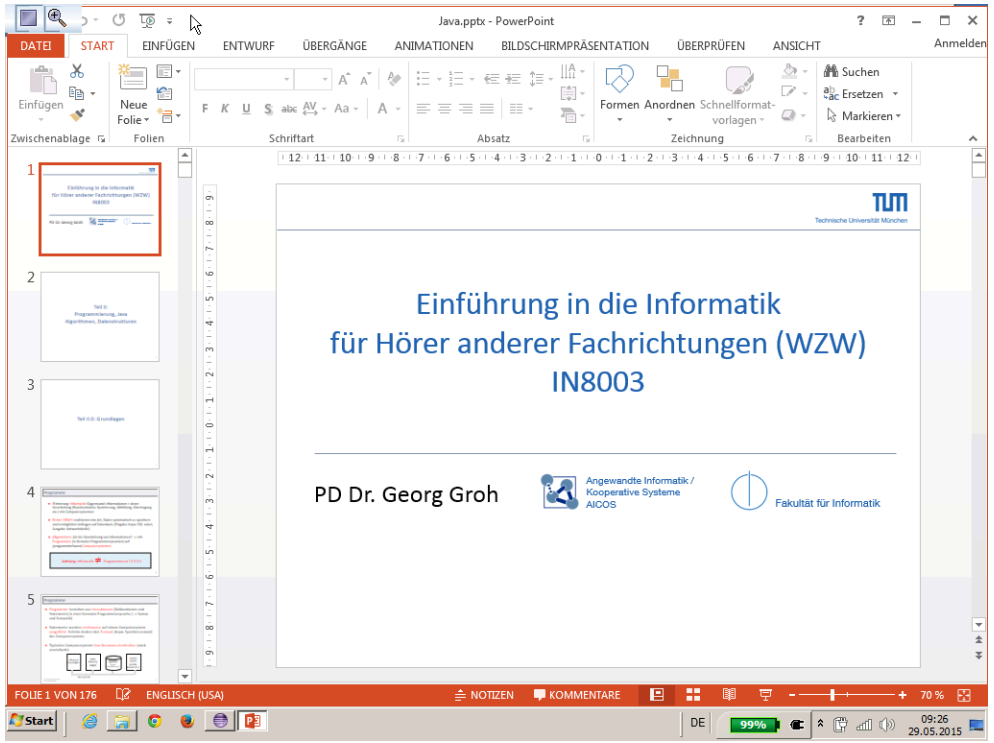
Script **generated by TTT**

Title: groh: profile1 (29.05.2015)

Date: Fri May 29 09:26:09 CEST 2015

Duration: 82:50 min

Pages: 61



Programm und vereinfachtes Speichermodell

```
...
int horst;
int heiner;
int fritz;
horst = 101;
heiner = 2;
fritz = horst + heiner;
horst = 2000;
...
```

Variablenname

Deklarationen

Statements

Vereinfachtes Speicher-Modell

Zell-Nr (Adresse)	Zell-Name (Variablenname)	Zell-Inhalt	
1124			Daten
1125	horst		
1126	heiner		
1127			
1128	fritz		
4027			Programm
4028		int horst;	
4029		int heiner;	
4030		int fritz;	
4029		horst = 101;	
4030		heiner = 2;	

Kontrollfluss

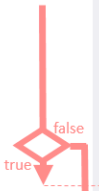
```
...
int horst;
int heiner;
int fritz;
horst = 101;
heiner = 2;
fritz = horst + heiner;
horst = 2000;
...
```

Vereinfachtes Speicher-Modell

Zell-Nr (Adresse)	Zell-Name (Variablenname)	Zell-Inhalt	
1124			Daten
1125	horst	101	
1126	heiner		
1127			
1128	fritz		
4027			Programm
4028		int horst;	
4029		int heiner;	
4030		int fritz;	
4029		horst = 101;	
4030		heiner = 2;	

Kontrollfluss: Bedingte Verzweigung

```
...
int horst;
int heiner;
int fritz;
horst = 101;
heiner = 2;
fritz = horst + heiner;
horst = 2000;
if(heiner == 2)
        horst = 10;
else
        horst = 11;
fritz = 17;
...
```



Vereinfachtes Speicher-Modell

Zell-Nr (Adresse)	Zell-Name (Variablenname)	Zell-Inhalt	
1124			Daten
1125	horst	2000	
1126	heiner	2	
1127			
1128	fritz	103	
4027			Programm
4028		int horst;	
4029		int heiner;	
4030		int fritz;	
4029		horst = 101;	
4030		heiner = 2;	

Kontrollfluss: Methodenaufruf

```
...
int horst;
int heiner;
int fritz;
horst = 101;
heiner = 2;
fritz = horst + heiner;
horst = 2000;
fritz = doSelfSumSquare(5);
fritz = doSelfSumSquare(heiner);
...
```

```
int doSelfSumSquare(int someNumber){
        int a;
        a = someNumber + someNumber;
        a = a * a;
        return a;
}
```

Vereinfachtes Speicher-Modell

Zell-Nr (Adresse)	Zell-Name (Variablenname)	Zell-Inhalt	
1124			Daten
1125	horst	2000	
1126	heiner	2	
1127			
1128	fritz	100	
2024			
2025			
4027			Programm
4028		int horst;	
4029		int heiner;	
4030		int fritz;	
4029		horst = 101;	
4030		heiner = 2;	
8756		int a;	
8757		a=someNumber+someNumber;	
8758		a = a * a;	

Prozedurale Programmierung

Gruppiere Sequenzen von Instruktionen in benamte „Prozeduren“ („Funktionen“, „Methoden“, „Subroutinen“ etc.)

```
int doSelfSumSquare(int someNumber){
        int a;
        a = someNumber + someNumber;
        a = a * a;
        return a;
}
```

$f(x) = (x + x)^2$

Vorteile:

- Sequenzen von Instruktionen müssen nicht immer wieder kopiert werden
- bessere Testbarkeit
- Modularität (bspw. betrifft eine Änderung innerhalb der Prozedur nicht die anderen Programmstellen wo diese benutzt wird.)
- Code Wiederverwendung
- etc.

Prozedurale Programmierung

Gruppiere Sequenzen von Instruktionen in benamte „Prozeduren“ („Funktionen“, „Methoden“, „Subroutinen“ etc.)

```
int doSelfSumSquare(int someNumber){
        int a;
        a = someNumber + someNumber;
        a = a * a;
        return a;
}
```

$f(x) = (x + x)^2$

Vorteile:

- Sequenzen von Instruktionen müssen nicht immer wieder kopiert werden
- bessere Testbarkeit
- Modularität (bspw. betrifft eine Änderung innerhalb der Prozedur nicht die anderen Programmstellen wo diese benutzt wird.)
- Code Wiederverwendung
- etc.

Prozedurale Programmierung

Gruppiere Sequenzen von Instruktionen in benamte „Prozeduren“ („Funktionen“, „Methoden“, „Subroutinen“ etc.)

```
int doSelfSumSquare(int someNumber){
        int a;
        a = someNumber + someNumber;
        a = a * a;
        return a;
}
```

$f(x) = (x + x)^2$

Vorteile:

- Sequenzen von Instruktionen müssen nicht immer wieder kopiert werden
- bessere Testbarkeit
- Modularität (bspw. betrifft eine Änderung innerhalb der Prozedur nicht die anderen Programmstellen wo diese benutzt wird.)
- Code Wiederverwendung
- etc.

Prozedurale Programmierung

Gruppiere Sequenzen von Instruktionen in benamte „Prozeduren“ („Funktionen“, „Methoden“, „Subroutinen“ etc.)

```
int doSelfSumSquare(int someNumber){
        int a;
        a = someNumber + someNumber;
        a = a * a;
        return a;
}
```

$f(x) = (x + x)^2$

Vorteile:

- Sequenzen von Instruktionen müssen nicht immer wieder kopiert werden
- bessere Testbarkeit
- Modularität (bspw. betrifft eine Änderung innerhalb der Prozedur nicht die anderen Programmstellen wo diese benutzt wird.)
- Code Wiederverwendung
- etc.

Kontrollfluss: Methodenaufruf

```
...
int horst;
int heiner;
int fritz;
horst = 101;
heiner = 2;
fritz = horst + heiner;
horst = 2000;
fritz = doSelfSumSquare(5);
fritz = doSelfSumSquare(heiner);
...
```

```
int doSelfSumSquare(int someNumber){
        int a;
        a = someNumber + someNumber;
        a = a * a;
        return a;
}
```

Vereinfachtes Speicher-Modell

Zell-Nr (Adresse)	Zell-Name (Variablenname)	Zell-Inhalt	
1124			Daten
1125	horst	2000	
1126	heiner	2	
1127			
1128	fritz	100	
2024			
2025			
4027			Programm
4028		int horst;	
4029		int heiner;	
4030		int fritz;	
4029		horst = 101;	
4030		heiner = 2;	
8756		int a;	
8757		a=someNumber+someNumber;	
8758		a = a * a;	

Literatur zu Teil II.0

- Lernziele der Veranstaltung: Praktische Fähigkeit, kleinere Probleme mit Datenbanken und Java lösen zu können + Grundlagen für vertiefte Einarbeitung legen.
- Begrenzte Zeit in Veranstaltung --> Viele grundlegende Informatikkonzepte können nur vereinfacht behandelt werden.
- Die meisten Bücher ≈„Einführung in die Informatik“ haben Informatiker als Zielgruppe --> hier nur eingeschränkt zu empfehlen
- --> Wikipedia-Artikel als Hintergrundliteratur. Ergänzend / alternativ: [1] und [2]

WIKIPEDIA
The Free Encyclopedia

Grundidee:

Gruppiere Daten und Prozeduren in Objekte ←→

Modelle von Zustand und Verhalten von (meist realweltlichen) Objekten

Attribute
(engl. fields, attributes, properties)

Methoden
(engl. methods)

Klasse: *Bauplan* für Objekte

Objekte: *Instanzen* ihrer Klasse

Grundidee:

Gruppiere Daten und Prozeduren in Objekte ←→

Modelle von Zustand und Verhalten von (meist realweltlichen) Objekten

Attribute
(engl. fields, attributes, properties)

Methoden
(engl. methods)

- Methoden sollten bevorzugt auf den eigenen Attributen arbeiten


```
class Bicycle {
       int cadence = 0;
       int speed = 0;
       int gear = 1;

       void changeCadence(int newValue) {
            cadence = newValue;
       }

       void changeGear(int newValue) {
            gear = newValue;
       }

       void speedUp(int increment) {
            speed = speed + increment;
       }

       void applyBrakes(int decrement) {
            speed = speed - decrement;
       }
}
```

Attribute (Zustand)

Methoden (Verhalten)

Klassen-Definition


```
class Bicycle {
       int cadence = 0;
       int speed = 0;
       int gear = 1;

       void changeCadence(int newValue) {
            cadence = newValue;
       }

       void changeGear(int newValue) {
            gear = newValue;
       }

       void speedUp(int increment) {
            speed = speed + increment;
       }

       void applyBrakes(int decrement) {
            speed = speed - decrement;
       }
}
```

Attribute (Zustand)

Methoden (Verhalten)

Klassen-Definition

Klassen und Objekte in Java

```
class BicycleDemo {
     public static void main(String[] args) {
          // Create two different Bicycle objects
          Bicycle bike1 = new Bicycle();
          Bicycle bike2 = new Bicycle();

          // Invoke methods on these objects
          bike1.changeCadence(50);
          bike1.speedUp(10);
          bike1.changeGear(2);

          bike2.changeCadence(50);
          bike2.speedUp(10);
          bike2.changeGear(2);
          bike2.changeCadence(40);
          bike2.speedUp(10);
          bike2.changeGear(3);
       }
}
```

```
class Bicycle {
       int cadence = 0;
       int speed = 0;
       int gear = 1;

       void changeCadence(int newValue) {
            cadence = newValue;
       }

       void changeGear(int newValue) {
            gear = newValue;
       }

       void speedUp(int increment) {
            speed = speed + increment;
       }

       void applyBrakes(int decrement) {
            speed = speed - decrement;
       }
}
```


Klassen und Objekte in Java

```
class BicycleDemo {
     public static void main(String[] args) {
          // Create two different Bicycle objects
          Bicycle bike1 = new Bicycle();
          Bicycle bike2 = new Bicycle();

          // Invoke methods on these objects
          bike1.changeCadence(50);
          bike1.speedUp(10);
          bike1.changeGear(2);

          bike2.changeCadence(50);
          bike2.speedUp(10);
          bike2.changeGear(2);
          bike2.changeCadence(40);
          bike2.speedUp(10);
          bike2.changeGear(3);
       }
}
```

```
class Bicycle {
       int cadence = 0;
       int speed = 0;
       int gear = 1;

       void changeCadence(int newValue) {
            cadence = newValue;
       }

       void changeGear(int newValue) {
            gear = newValue;
       }

       void speedUp(int increment) {
            speed = speed + increment;
       }

       void applyBrakes(int decrement) {
            speed = speed - decrement;
       }
}
```

42

Klassen und Objekte in Java

```
class BicycleDemo {
     public static void main(String[] args) {
          // Create two different Bicycle objects
          Bicycle bike1 = new Bicycle();
          Bicycle bike2 = new Bicycle();

          // Invoke methods on these objects
          bike1.changeCadence(50);
          bike1.speedUp(10);
          bike1.changeGear(2);

          bike2.changeCadence(50);
          bike2.speedUp(10);
          bike2.changeGear(2);
          bike2.changeCadence(40);
          bike2.speedUp(10);
          bike2.changeGear(3);
       }
}
```

```
class Bicycle {
       int cadence = 0;
       int speed = 0;
       int gear = 1;

       void changeCadence(int newValue) {
            cadence = newValue;
       }

       void changeGear(int newValue) {
            gear = newValue;
       }

       void speedUp(int increment) {
            speed = speed + increment;
       }

       void applyBrakes(int decrement) {
            speed = speed - decrement;
       }
}
```

42

Datenbanken

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

Java

```
public class SomeCode {

  public static void main(String[] args) {
     Professor prof2125 = new Professor("Sokrates", "C4", 226);
     Professor russelTheOldLad = new Professor("Russel", "C4", 232);
     Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
     Professor gtuwegghf678 = new Professor("Popper", "C3", 52);
     Professor gustl  = new Professor("Augustinus", "C3", 309);
     Professor oldMary = new Professor("Curie", "C4", 36);
     Professor prof_2144 = new Professor("Kant", "C4", 7);
      ...
  }
}
```

```
public class Professor {
   public String name;
   public String rang;
   public int raum;

   public Professor(String name, String rang, int raum){
      this.name = name;
      this.rang = rang;
      this.raum = raum;
   }

   public void teach(){
      System.out.println("... now teaching something :-)");
   }
}
```

Datenbanken

Professoren

PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

Java

```
public class SomeCode {

  public static void main(String[] args) {
     Professor prof2125 = new Professor("Sokrates", "C4", 226);
     Professor russelTheOldLad = new Professor("Russel", "C4", 232);
     Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
     Professor gtuwegghf678 = new Professor("Popper", "C3", 52);
     Professor gustl  = new Professor("Augustinus", "C3", 309);
     Professor oldMary = new Professor("Curie", "C4", 36);
     Professor prof_2144 = new Professor("Kant", "C4", 7);
     ...
  }
}
```

```
public class Professor {
   public String name;
   public String rang;
   public int raum;

   public Professor(String name, String rang, int raum){
      this.name = name;
      this.rang = rang;
      this.raum = raum;
   }

   public void teach(){
      System.out.println("... now teaching something :-)");
   }
}
```


Datenbanken

Professoren

PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

Java

```
public class SomeCode {

  public static void main(String[] args) {
     Professor prof2125 = new Professor("Sokrates", "C4", 226);
     Professor russelTheOldLad = new Professor("Russel", "C4", 232);
     Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
     Professor gtuwegghf678 = new Professor("Popper", "C3", 52);
     Professor gustl  = new Professor("Augustinus", "C3", 309);
     Professor oldMary = new Professor("Curie", "C4", 36);
     Professor prof_2144 = new Professor("Kant", "C4", 7);
     ...
  }
}
```

```
public class Professor {
   public String name;
   public String rang;
   public int raum;

   public Professor(String name, String rang, int raum){
      this.name = name;
      this.rang = rang;
      this.raum = raum;
   }

   public void teach(){
      System.out.println("... now teaching something :-)");
   }
}
```


Datenbanken

Professoren

PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

Java

```
public class SomeCode {

  public static void main(String[] args) {
     Professor prof2125 = new Professor("Sokrates", "C4", 226);
     Professor russelTheOldLad = new Professor("Russel", "C4", 232);
     Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
     Professor gtuwegghf678 = new Professor("Popper", "C3", 52);
     Professor gustl  = new Professor("Augustinus", "C3", 309);
     Professor oldMary = new Professor("Curie", "C4", 36);
     Professor prof_2144 = new Professor("Kant", "C4", 7);
     ...
  }
}
```

```
public class Professor {
   public String name;
   public String rang;
   public int raum;

   public Professor(String name, String rang, int raum){
      this.name = name;
      this.rang = rang;
      this.raum = raum;
   }

   public void teach(){
      System.out.println("... now teaching something :-)");
   }
}
```


Datenbanken

Professoren

PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

Java

```
public class SomeCode {

  public static void main(String[] args) {
     Professor prof2125 = new Professor("Sokrates", "C4", 226);
     Professor russelTheOldLad = new Professor("Russel", "C4", 232);
     Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
     Professor gtuwegghf678 = new Professor("Popper", "C3", 52);
     Professor gustl  = new Professor("Augustinus", "C3", 309);
     Professor oldMary = new Professor("Curie", "C4", 36);
     Professor prof_2144 = new Professor("Kant", "C4", 7);
     ...
  }
}
```

```
public class Professor {
   public String name;
   public String rang;
   public int raum;

   public Professor(String name, String rang, int raum){
      this.name = name;
      this.rang = rang;
      this.raum = raum;
   }

   public void teach(){
      System.out.println("... now teaching something :-)");
   }
}
```

Datenbanken

Java

???

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

```
public class SomeCode {

  public static void main(String[] args) {
    Professor prof2125 = new Professor("Sokrates", "C4", 226);
    Professor russelTheOldLad = new Professor("Russel", "C4", 232);
    Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
    Professor gtuweggghf678 = new Professor("Popper", "C3", 52);
    Professor gustl  = new Professor("Augustinus", "C3", 309);
    Professor oldMary = new Professor("Curie", "C4", 36);
    Professor prof_2144 = new Professor("Kant", "C4", 7);
    ...
  }
}
```

```
public class Professor {
  public String name;
  public String rang;
  public int raum;

  public Professor(String name, String rang, int raum){
    this.name = name;
    this.rang = rang;
    this.raum = raum;
  }

  public void teach(){
    System.out.println("... now teaching something :-)");
  }
}
```


Datenbanken

Java

???

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

```
public class SomeCode {

  public static void main(String[] args) {
    Professor prof2125 = new Professor("Sokrates", "C4", 226);
    Professor russelTheOldLad = new Professor("Russel", "C4", 232);
    Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
    Professor gtuweggghf678 = new Professor("Popper", "C3", 52);
    Professor gustl  = new Professor("Augustinus", "C3", 309);
    Professor oldMary = new Professor("Curie", "C4", 36);
    Professor prof_2144 = new Professor("Kant", "C4", 7);
    ...
  }
}
```

```
public class Professor {
  public String name;
  public String rang;
  public int raum;

  public Professor(String name, String rang, int raum){
    this.name = name;
    this.rang = rang;
    this.raum = raum;
  }

  public void teach(){
    System.out.println("... now teaching something :-)");
  }
}
```

46

Datenbanken

Java

???

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

```
public class SomeCode {

  public static void main(String[] args) {
    Professor prof2125 = new Professor("Sokrates", "C4", 226);
    Professor russelTheOldLad = new Professor("Russel", "C4", 232);
    Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
    Professor gtuweggghf678 = new Professor("Popper", "C3", 52);
    Professor gustl  = new Professor("Augustinus", "C3", 309);
    Professor oldMary = new Professor("Curie", "C4", 36);
    Professor prof_2144 = new Professor("Kant", "C4", 7);
    ...
  }
}
```

```
public class Professor {
  public String name;
  public String rang;
  public int raum;

  public Professor(String name, String rang, int raum){
    this.name = name;
    this.rang = rang;
    this.raum = raum;
  }

  public void teach(){
    System.out.println("... now teaching something :-)");
  }
}
```

46

Datenbanken

Java

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Professoren: {[PersNr: integer,
Name: *varchar(40)*,
Rang: *char(3)*,
Raum: *integer*]}

```
public class SomeCode {

  public static void main(String[] args) {
    Professor prof2125 = new Professor("Sokrates", "C4", 226);
    Professor russelTheOldLad = new Professor("Russel", "C4", 232);
    Professor kopiWopi = new Professor("Kopernikus", "C3", 310);
    Professor gtuweggghf678 = new Professor("Popper", "C3", 52);
    Professor gustl  = new Professor("Augustinus", "C3", 309);
    Professor oldMary = new Professor("Curie", "C4", 36);
    Professor prof_2144 = new Professor("Kant", "C4", 7);
    ...
  }
}
```

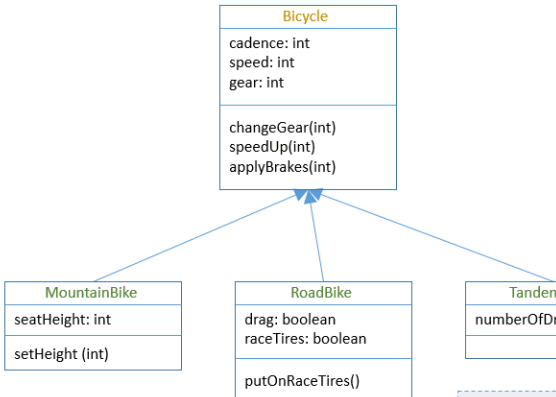
```
public class Professor {
  public String name;
  public String rang;
  public int raum;

  public Professor(String name, String rang, int raum){
    this.name = name;
    this.rang = rang;
    this.raum = raum;
  }

  public void teach(){
    System.out.println("... now teaching something :-)");
  }
}
```

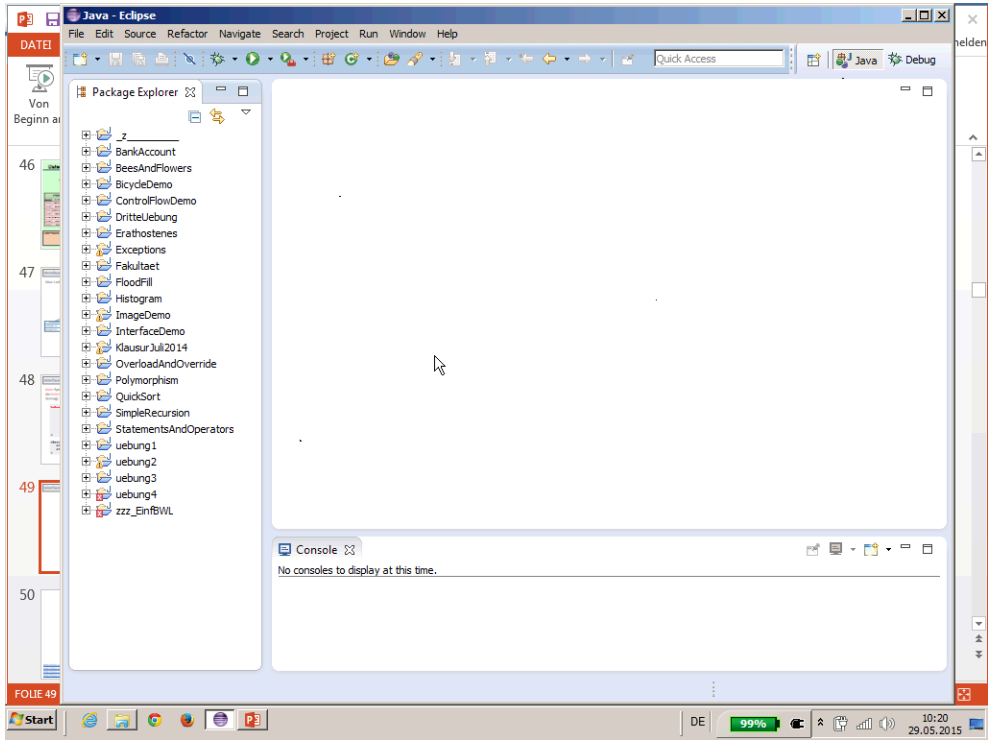
Vererbung

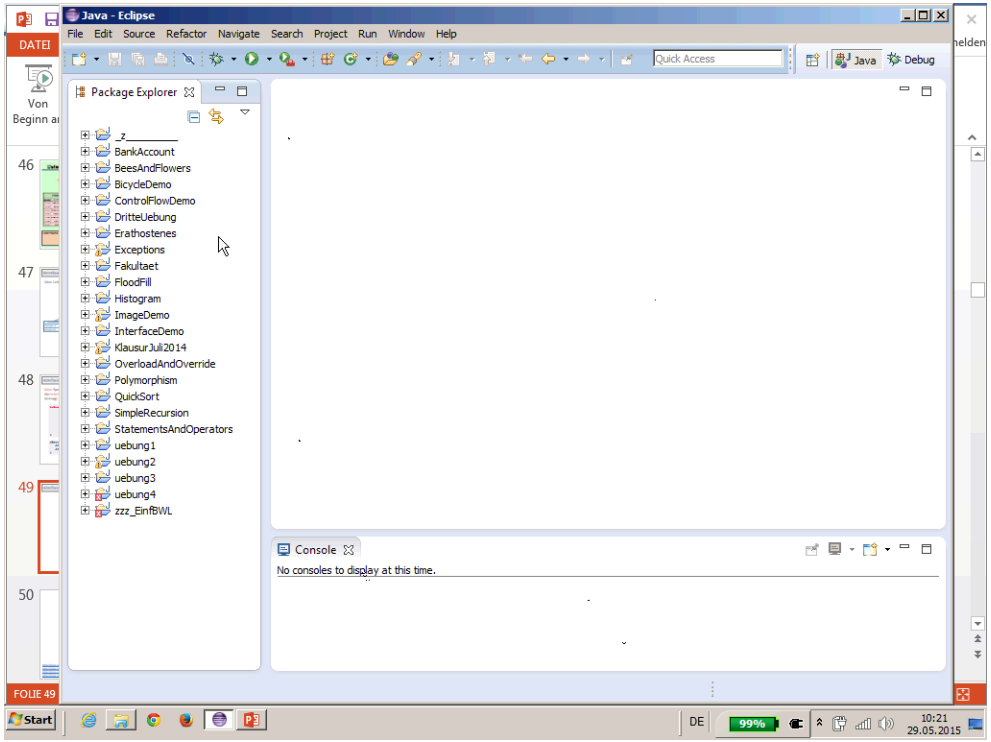
Idee: Leite speziellere Klassen von existierenden Klassen ab.

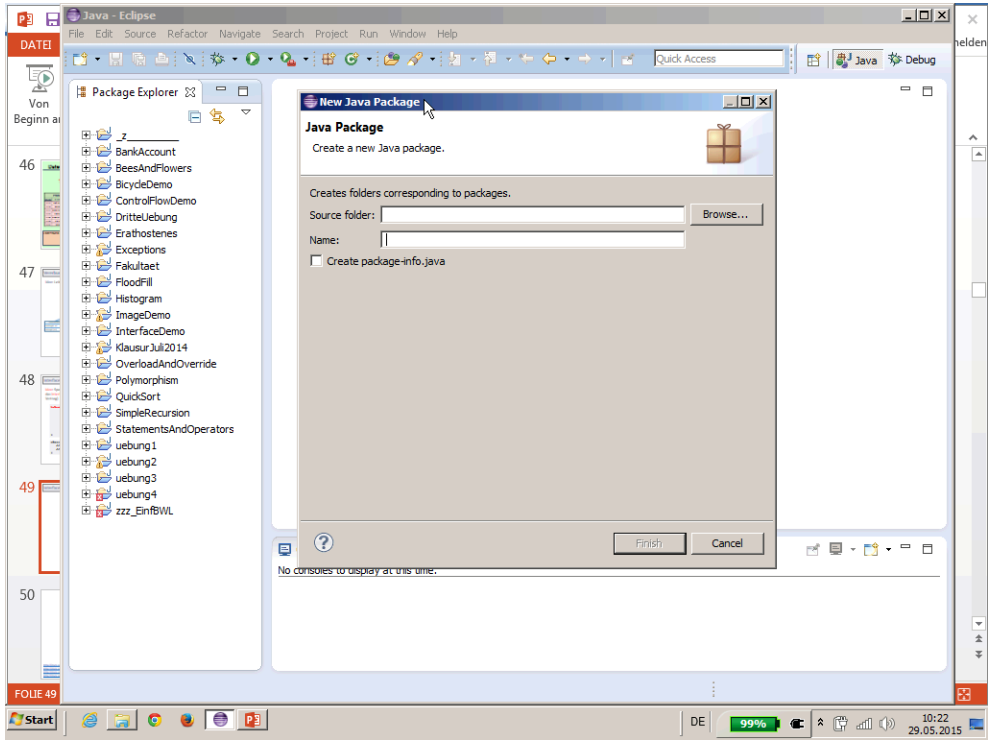


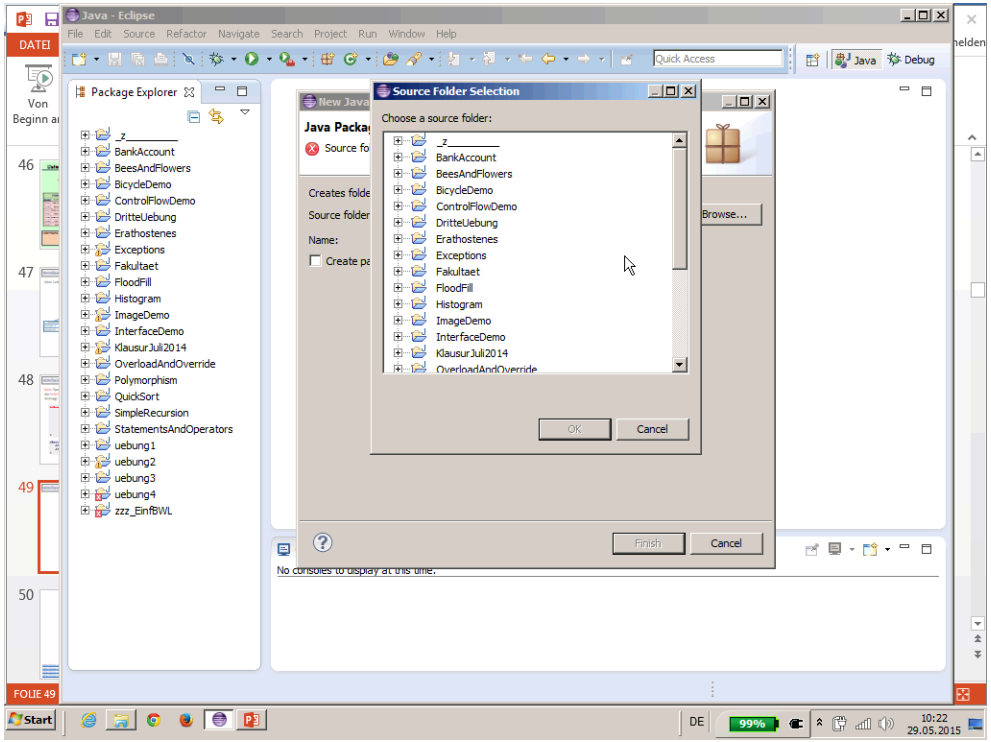

```
class RoadBike extends Bicycle {
    // additional fields and
methods
    // that define a road bike
    // go here
}
```

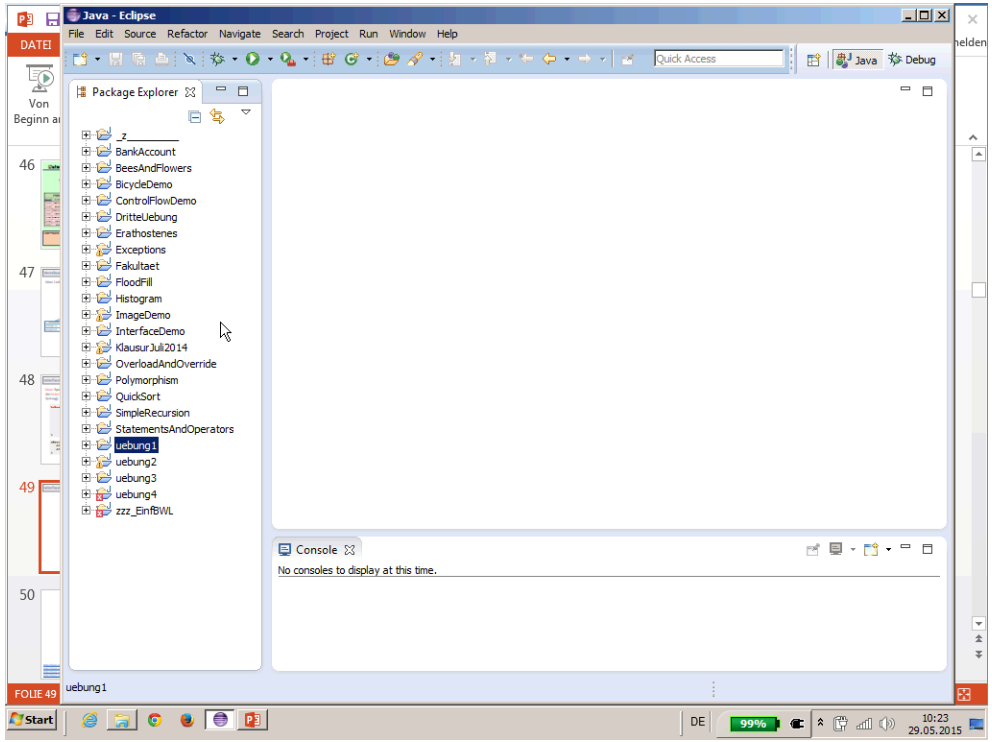
47

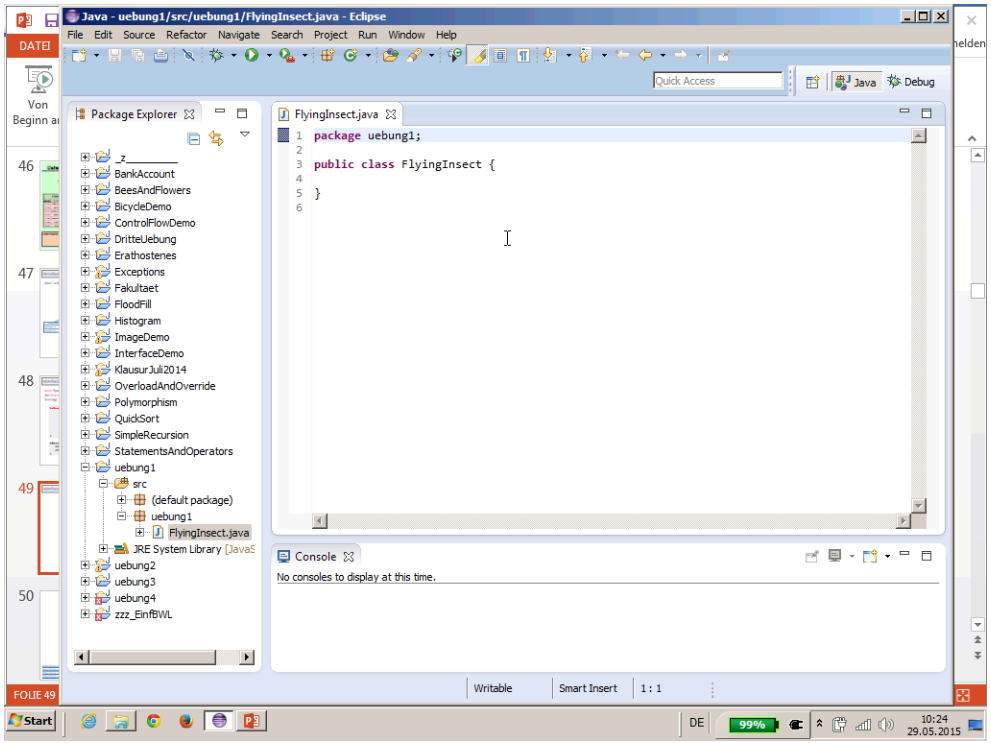


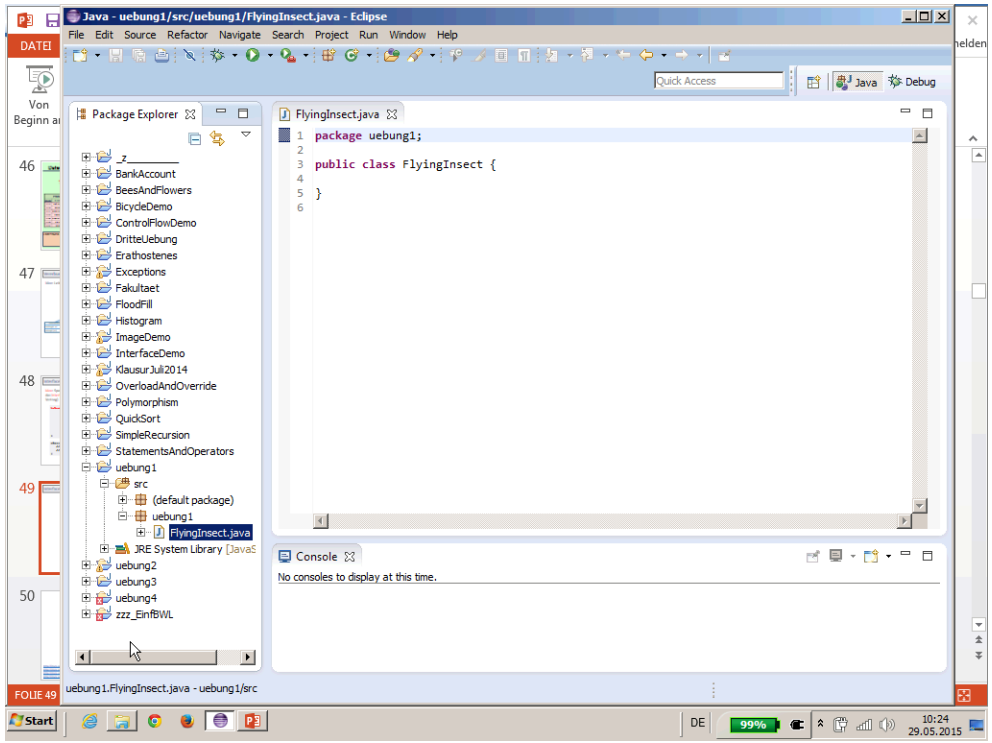


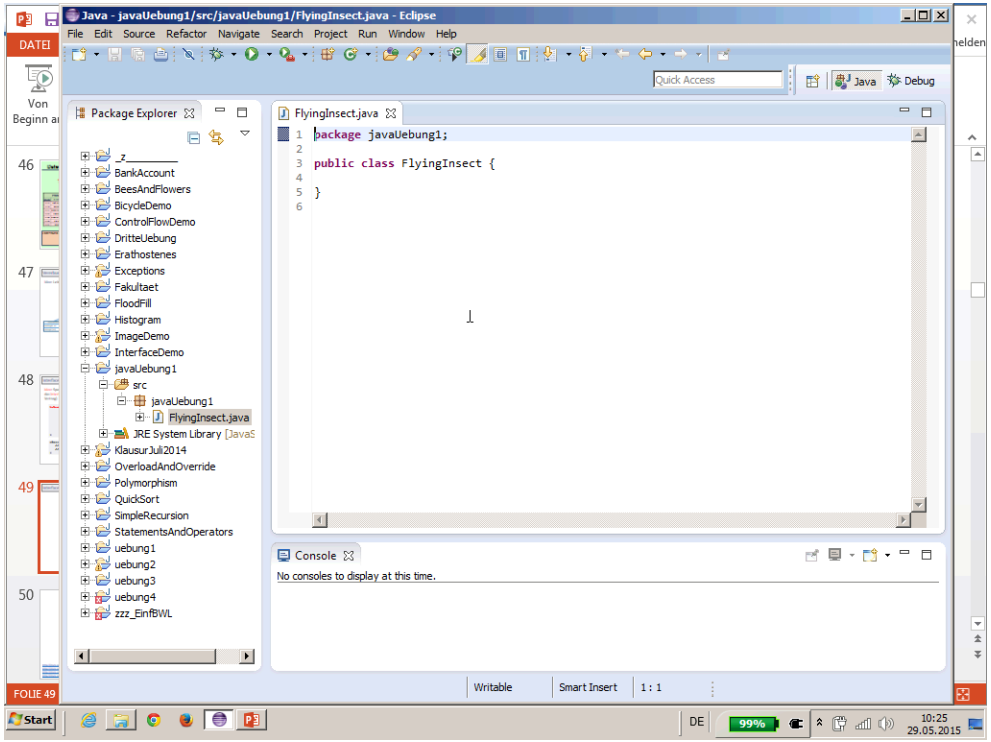


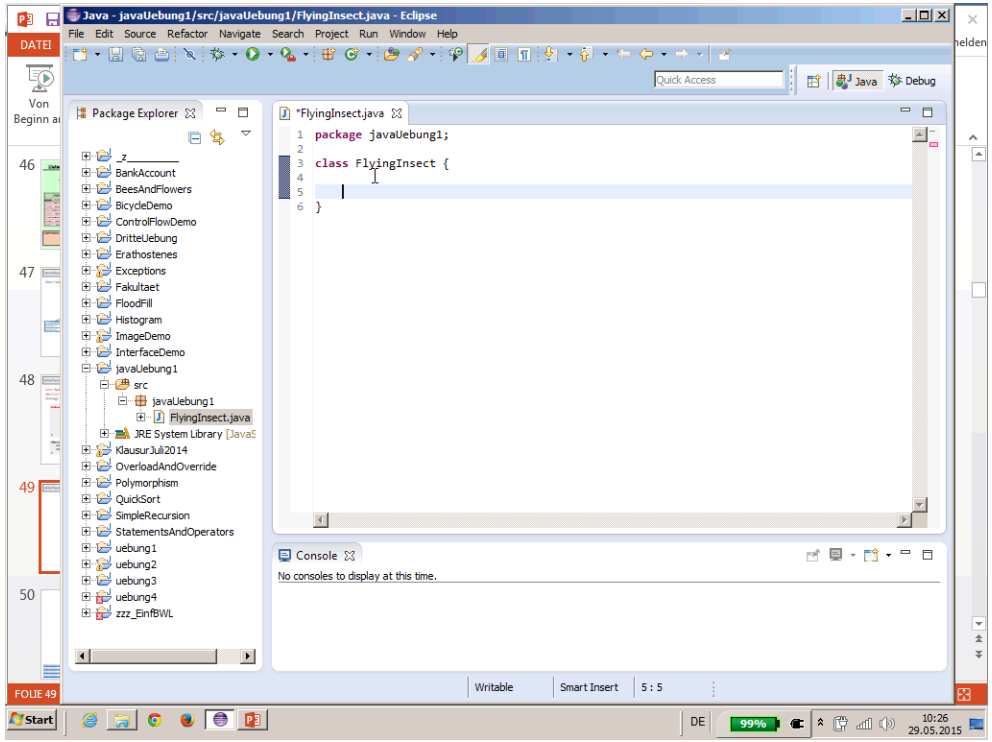

Java - Eclipse
Source Folder Selection
Choose a source folder:
OK
Cancel
Finish
Package Explorer
No consoles to display at this time.

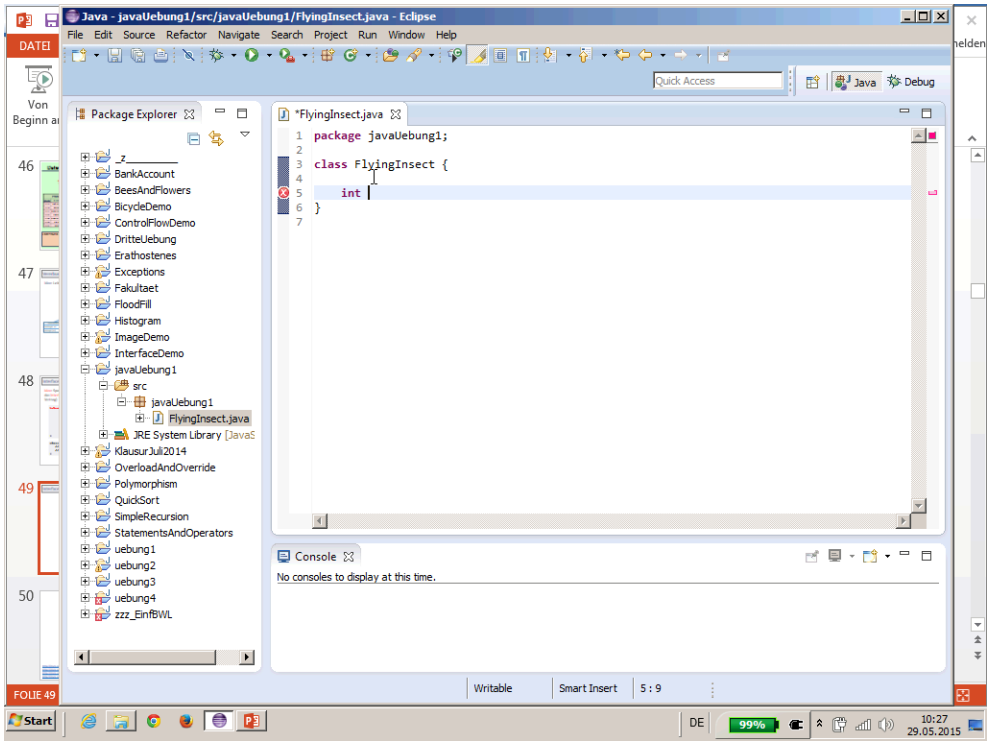

Java - Eclipse
Package Explorer
Console
No consoles to display at this time.
uebung1

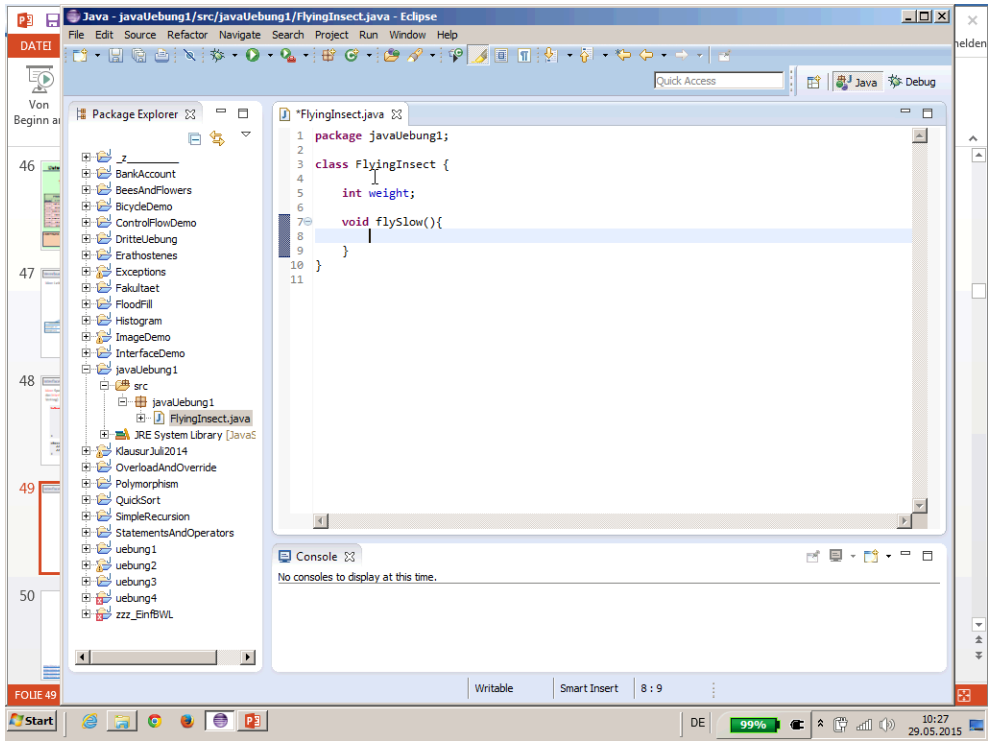

Java - uebung1/src/uebung1/FlyingInsect.java - Eclipse
FlyingInsect.java
package uebung1;
public class FlyingInsect {
}
Console
No consoles to display at this time.
Writable
Smart Insert
1 : 1

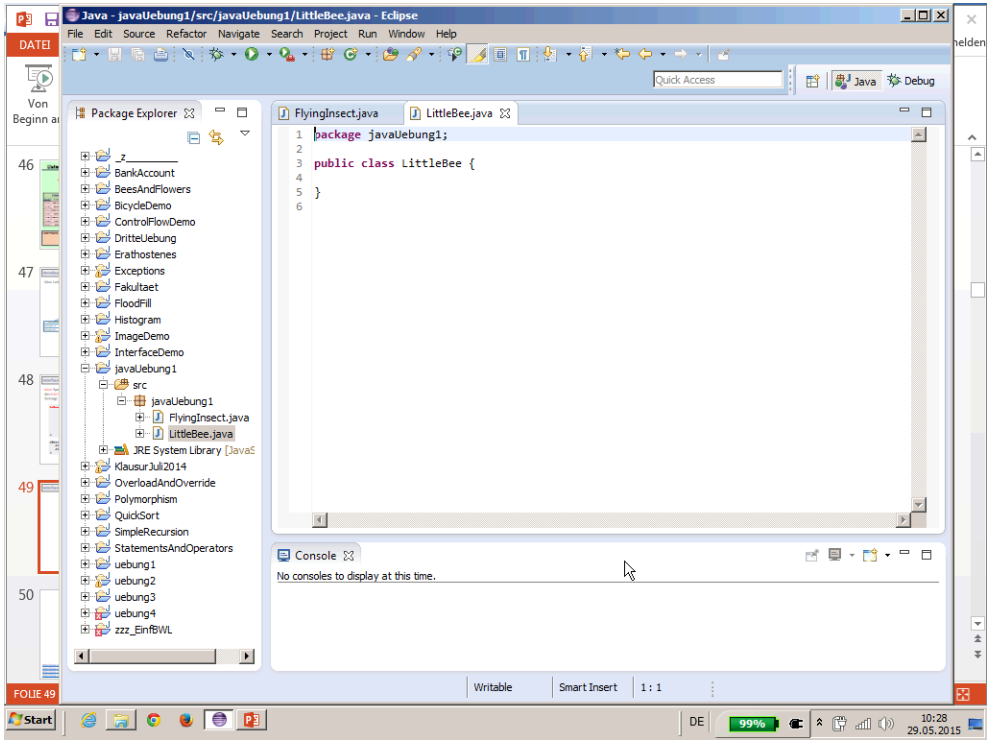

Java - uebung1/src/uebung1/FlyingInsect.java - Eclipse
FlyingInsect.java
package uebung1;
public class FlyingInsect {
}
Console
No consoles to display at this time.
uebung1.FlyingInsect.java - uebung1/src

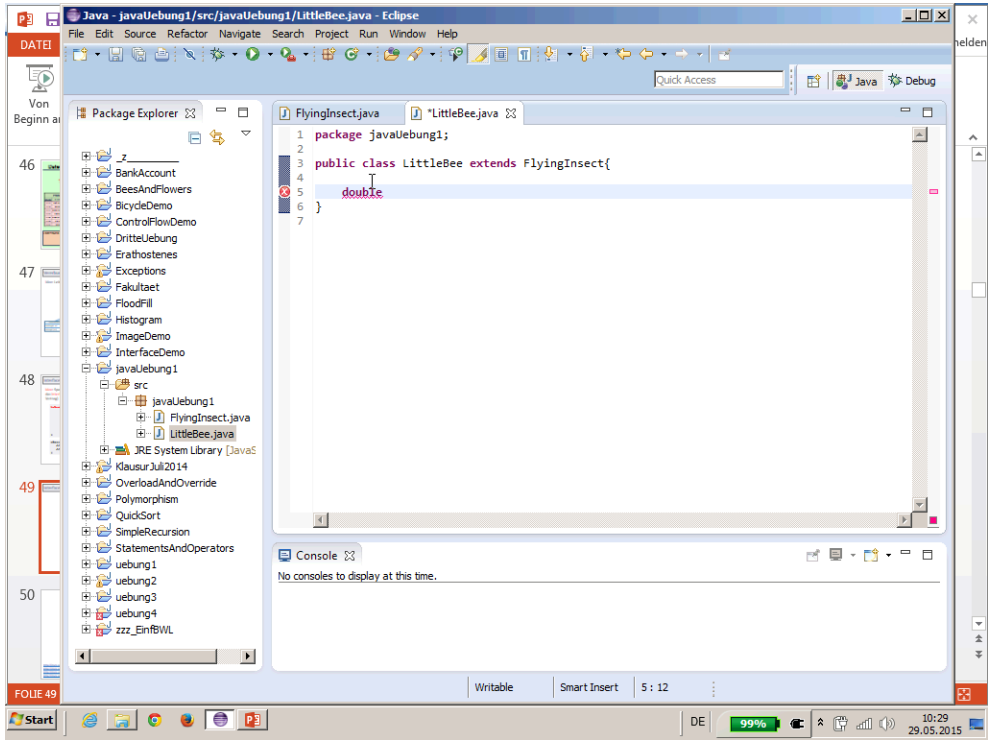

Java - javaUebung1/src/javaUebung1/FlyingInsect.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
1 package javaUebung1;
2
3 public class FlyingInsect {
4
5 }
6
Console
No consoles to display at this time.
Writable
Smart Insert
1 : 1

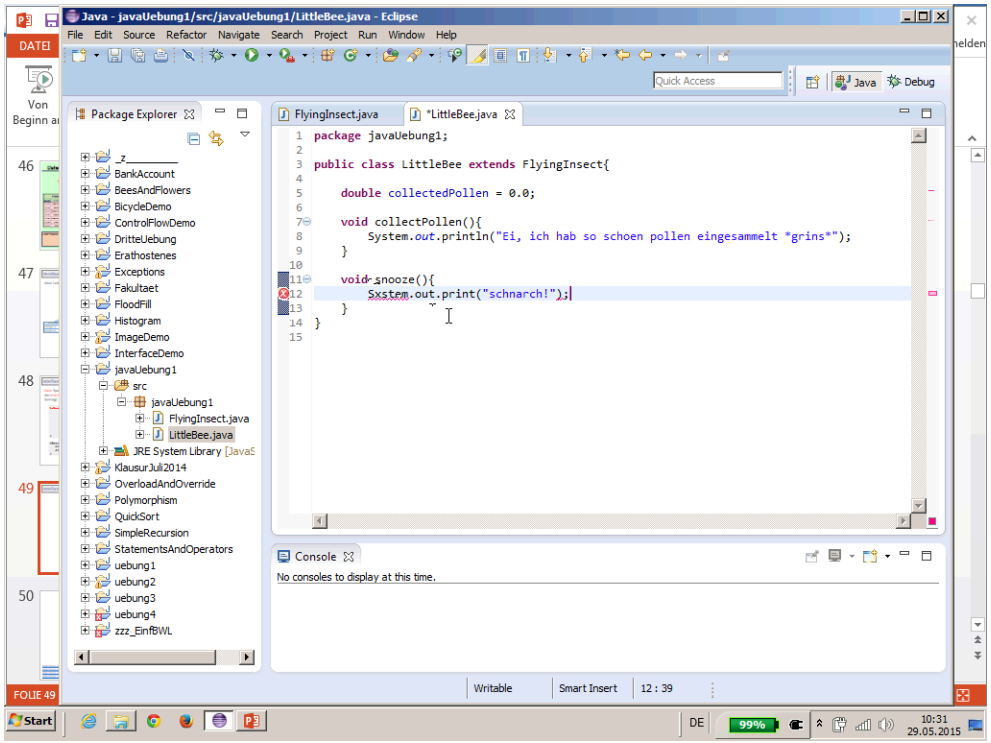

Java - javaUebung1/src/javaUebung1/FlyingInsect.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
*FlyingInsect.java
1 package javaUebung1;
2
3 class FlyingInsect {
4
5
6 }
Console
No consoles to display at this time.
Writable
Smart Insert
5 : 5

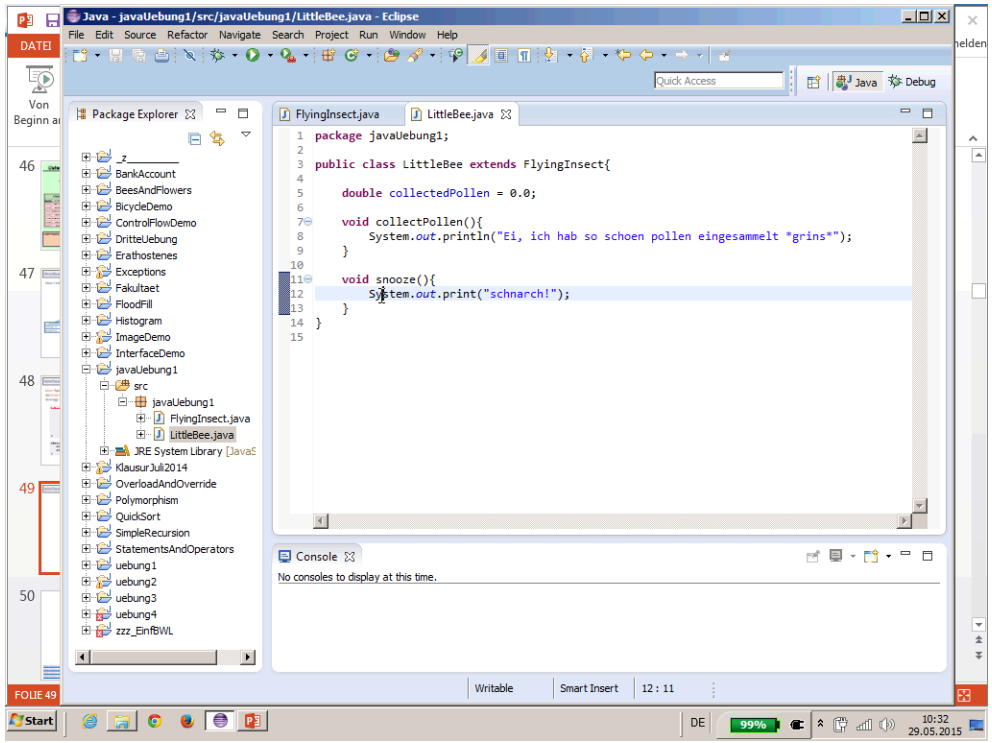

Java - javaUebung1/src/javaUebung1/FlyingInsect.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
*FlyingInsect.java
1 package javaUebung1;
2
3 class FlyingInsect {
4
5 int
6 }
7
Console
No consoles to display at this time.
Writable
Smart Insert
5 : 9

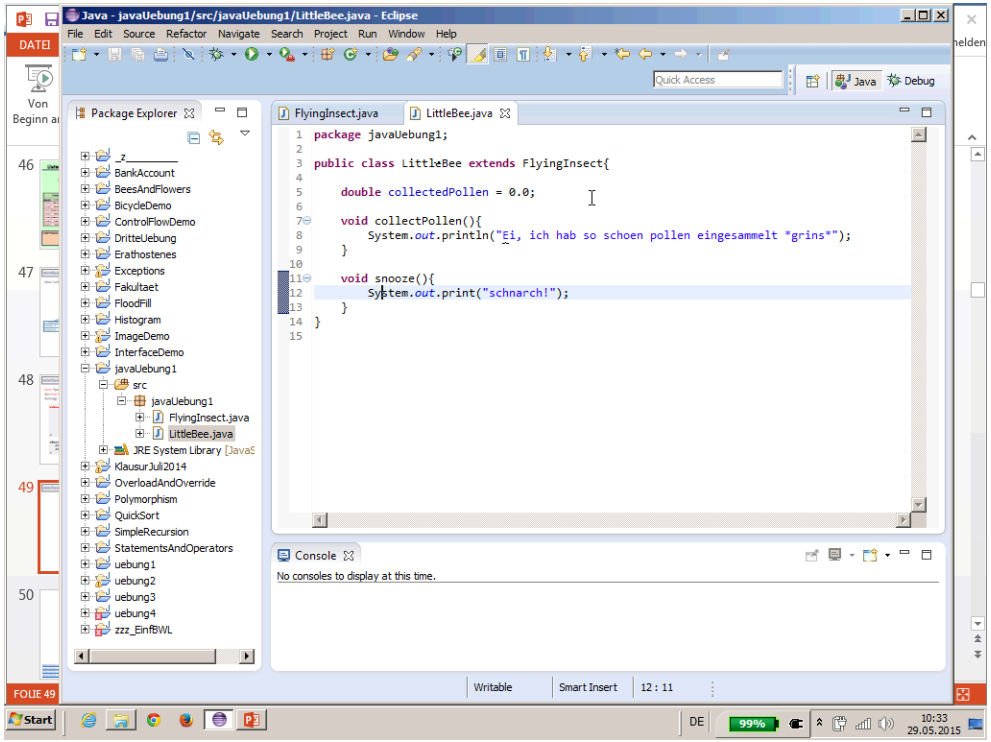

Java - javaUebung1/src/javaUebung1/FlyingInsect.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
*FlyingInsect.java
1 package javaUebung1;
2
3 class FlyingInsect {
4
5 int weight;
6
7 void flySlow(){
8
9 }
10 }
11
Console
No consoles to display at this time.
Writable
Smart Insert
8 : 9

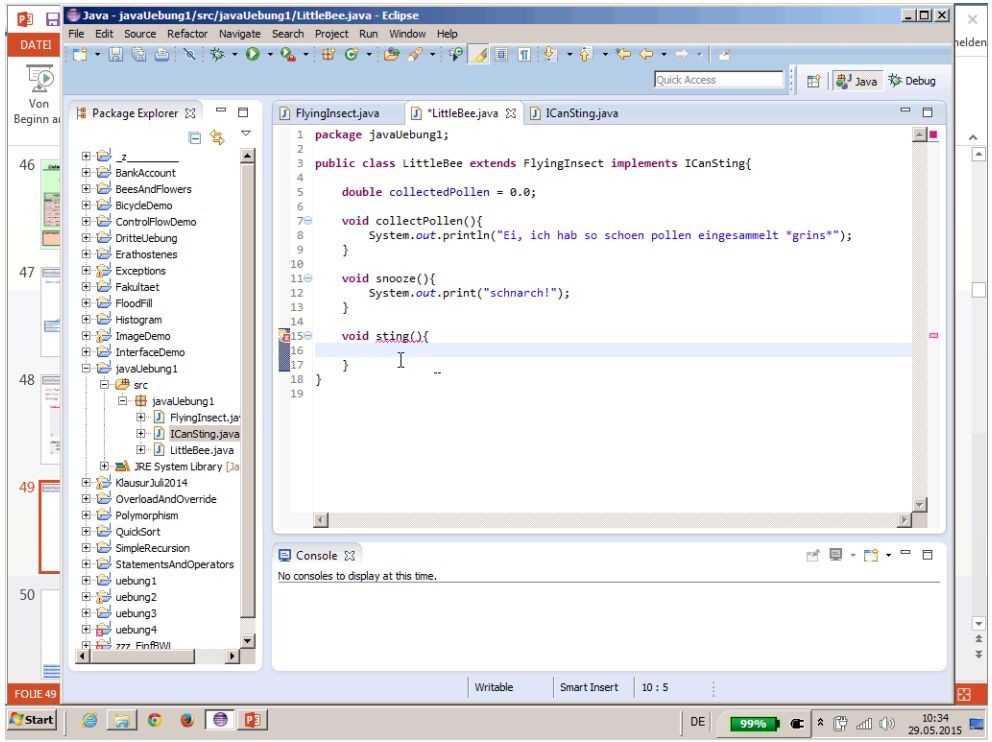

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee {
4
5 }
6
Console
No consoles to display at this time.
Writable
Smart Insert
1 : 1

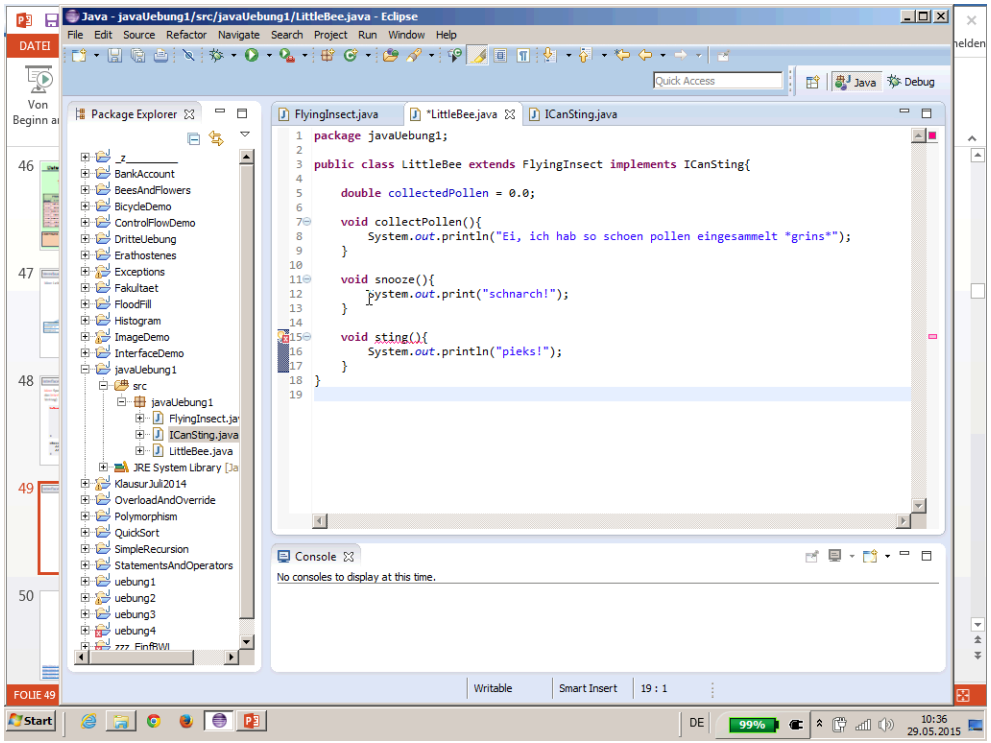

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
*LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect{
4
5 double
6 }
7
Console
No consoles to display at this time.
Writable
Smart Insert
5 : 12

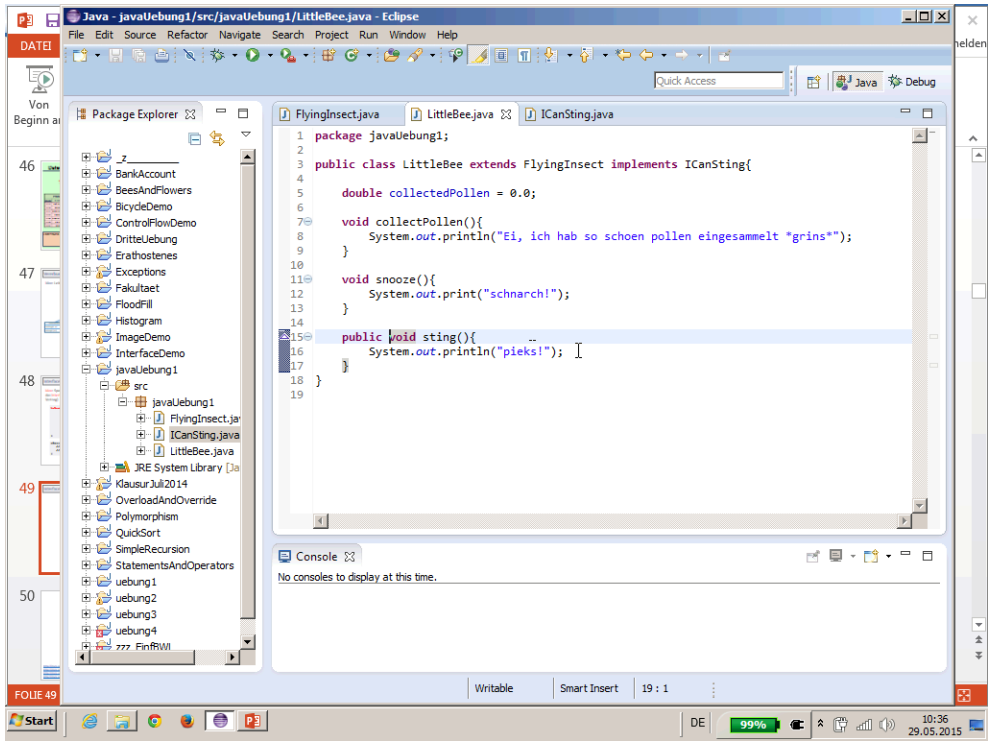

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
*LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect{
4
5 double collectedPollen = 0.0;
6
7 void collectPollen(){
8 System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
9 }
10
11 void snooze(){
12 Sxstem.out.print("schnarch!");
13 }
14 }
15
Console
No consoles to display at this time.
Writable
Smart Insert
12 : 39

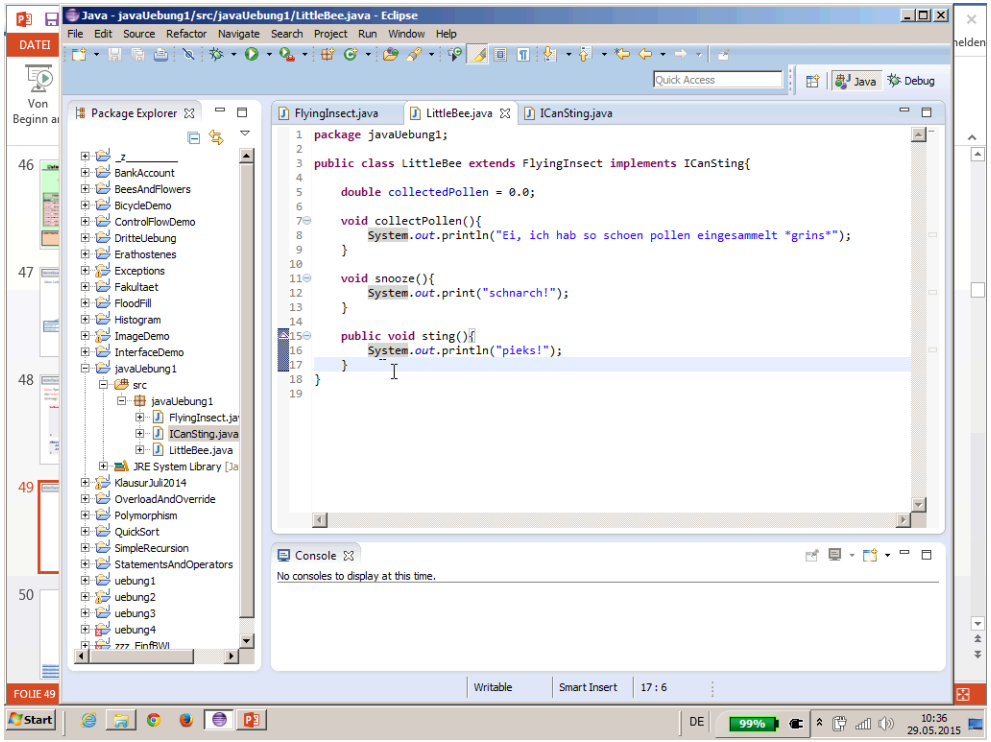

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
LittleBee.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect{
4
5 double collectedPollen = 0.0;
6
7 void collectPollen(){
8 System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
9 }
10
11 void snooze(){
12 System.out.print("schnarch!");
13 }
14 }
15
Console
No consoles to display at this time.
Writable
Smart Insert
12 : 11

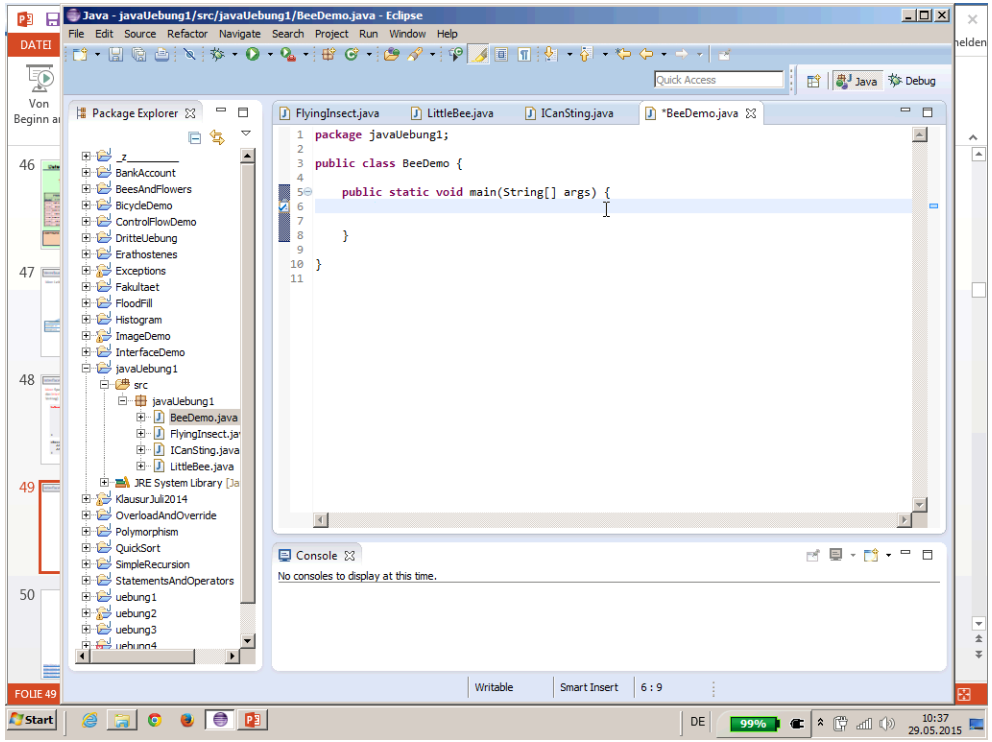

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
FlyingInsect.java
LittleBee.java
package javaUebung1;
public class LittleBee extends FlyingInsect{
double collectedPollen = 0.0;
void collectPollen(){
System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
}
void snooze(){
System.out.print("schnarch!");
}
}
No consoles to display at this time.
Writable
Smart Insert
12 : 11

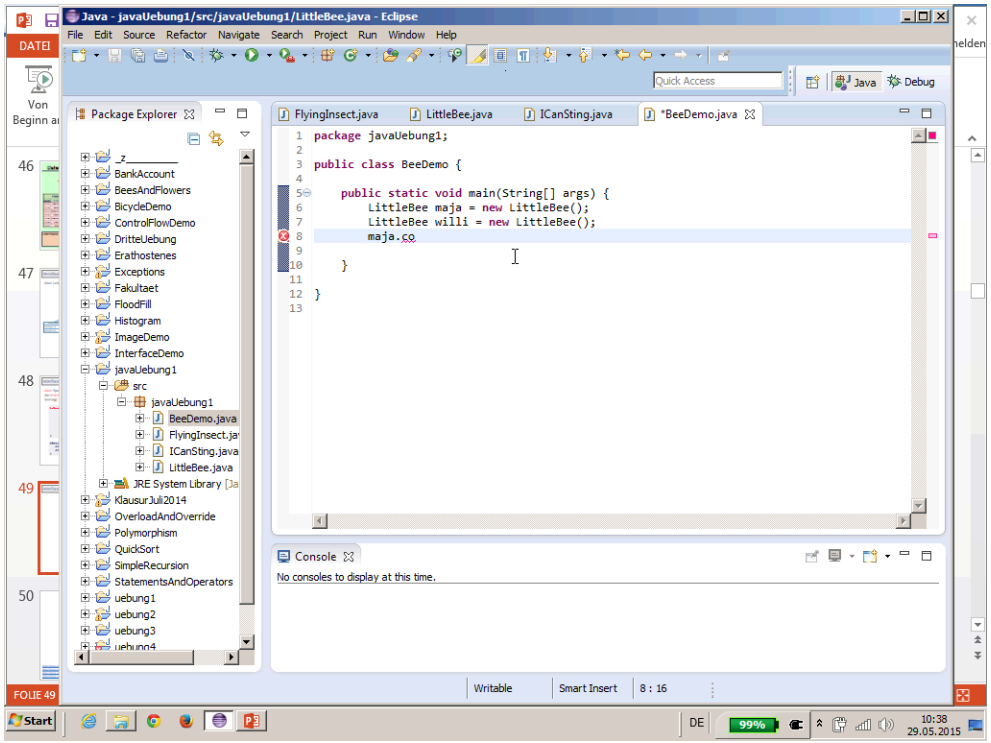

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
FlyingInsect.java
*LittleBee.java
ICanSting.java
package javaUebung1;
public class LittleBee extends FlyingInsect implements ICanSting{
double collectedPollen = 0.0;
void collectPollen(){
System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
}
void snooze(){
System.out.print("schnarch!");
}
void sting(){
}
}
No consoles to display at this time.
Writable
Smart Insert
10 : 5

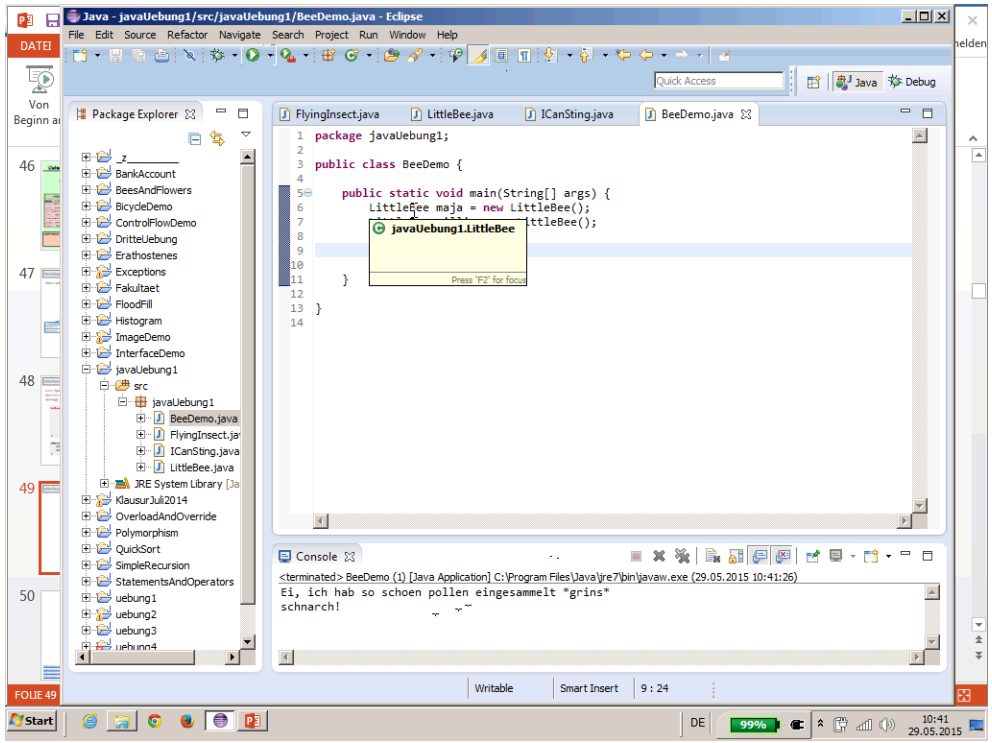

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
FlyingInsect.java
*LittleBee.java
ICanSting.java
package javaUebung1;
public class LittleBee extends FlyingInsect implements ICanSting{
double collectedPollen = 0.0;
void collectPollen(){
System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
}
void snooze(){
System.out.print("schnarch!");
}
void sting(){
System.out.println("pieks!");
}
}
No consoles to display at this time.
Writable
Smart Insert
19 : 1

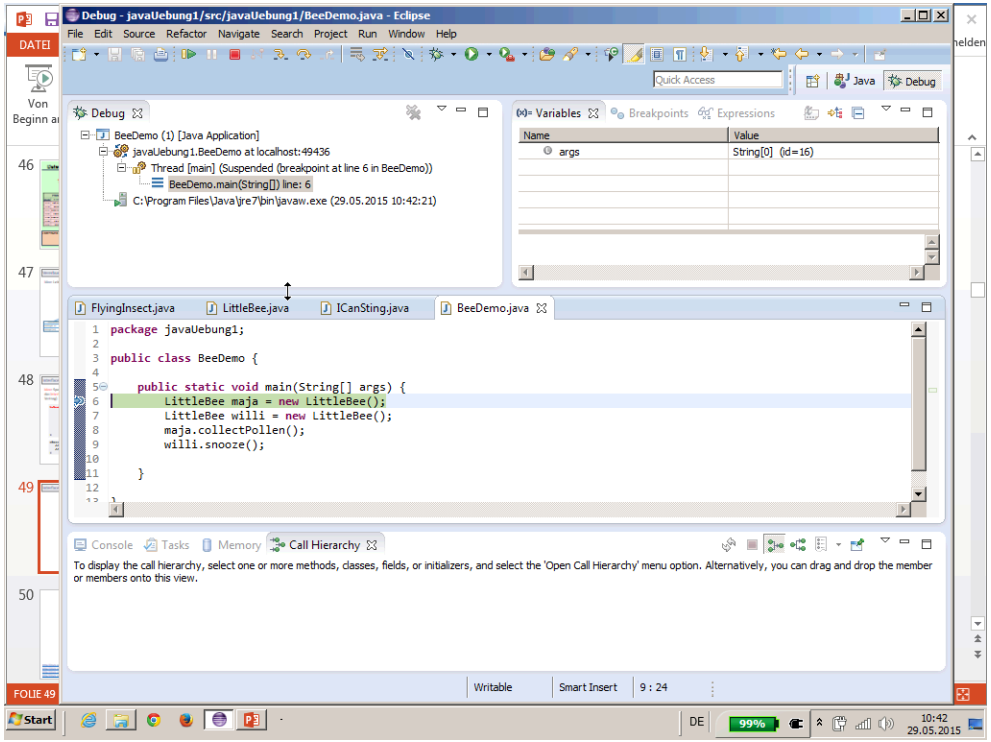

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
FlyingInsect.java
LittleBee.java
ICanSting.java
package javaUebung1;
public class LittleBee extends FlyingInsect implements ICanSting{
double collectedPollen = 0.0;
void collectPollen(){
System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
}
void snooze(){
System.out.print("schnarch!");
}
public void sting(){
System.out.println("pieks!");
}
}
No consoles to display at this time.
Writable
Smart Insert
19 : 1

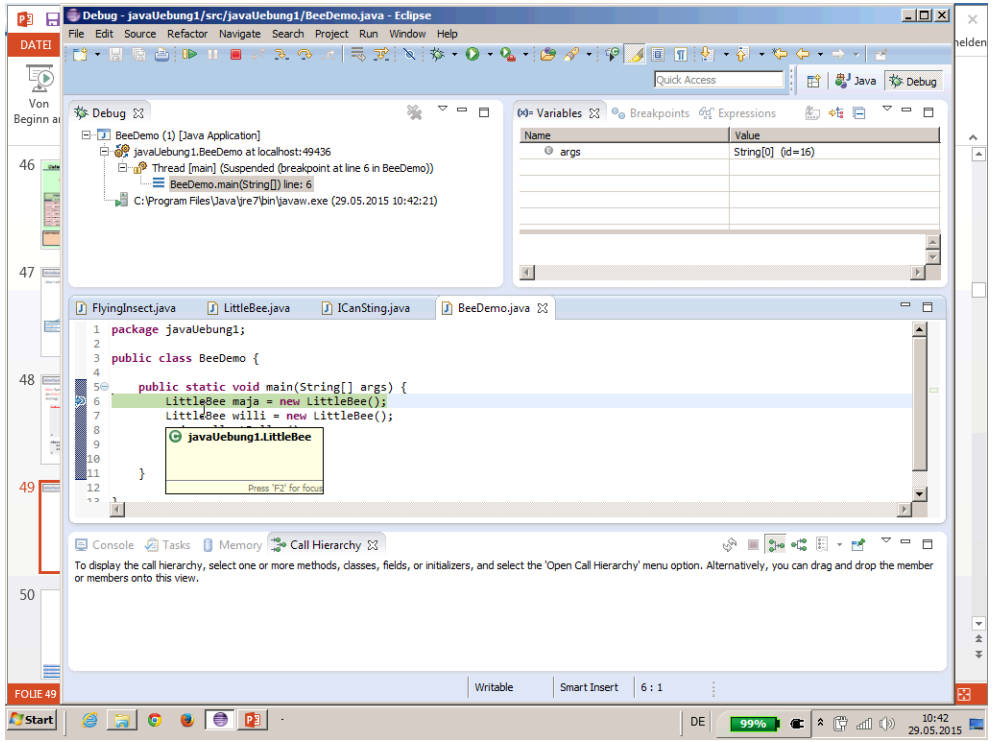

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
LittleBee.java
ICanSting.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSting{
4
5 double collectedPollen = 0.0;
6
7 void collectPollen(){
8 System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
9 }
10
11 void snooze(){
12 System.out.print("schnarch!");
13 }
14
15 public void sting(){
16 System.out.println("pieks!");
17 }
18 }
19
Console
No consoles to display at this time.
Writable
Smart Insert
17 : 6

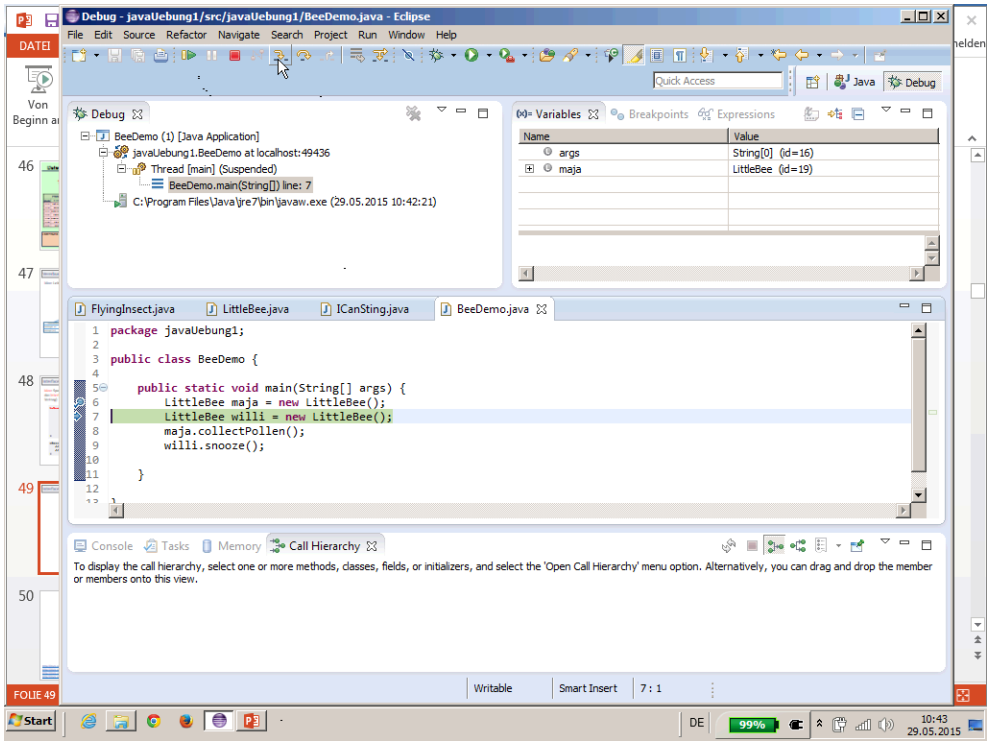

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
LittleBee.java
ICanSting.java
*BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6
7
8 }
9
10 }
11
Console
No consoles to display at this time.
Writable
Smart Insert
6 : 9

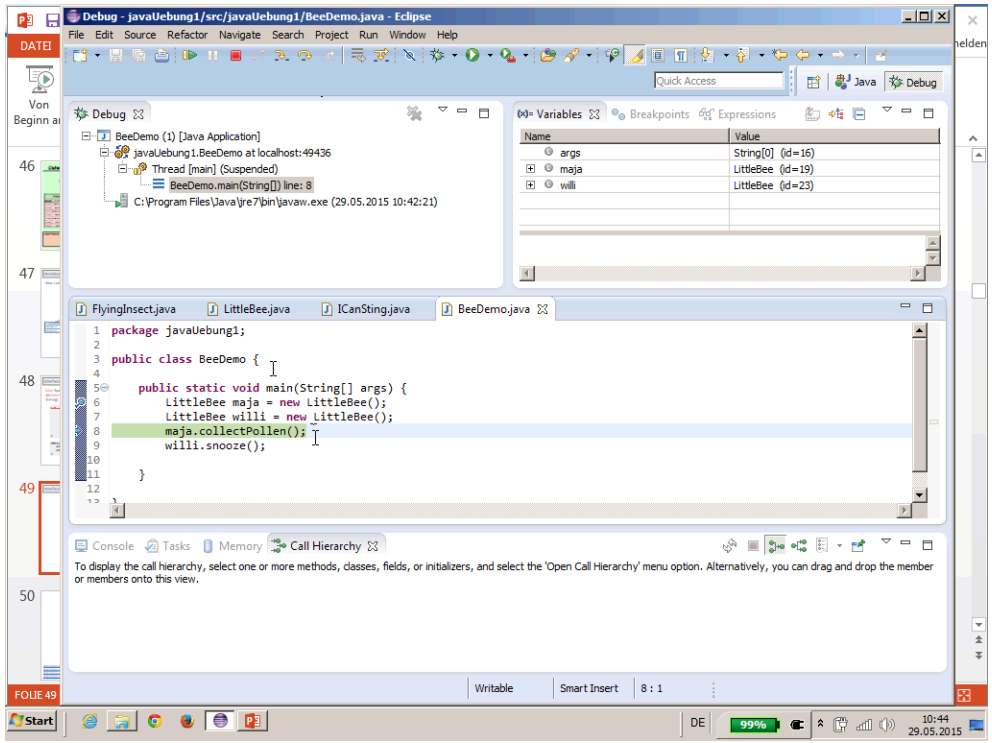

Java - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
LittleBee.java
ICanSting.java
*BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee willi = new LittleBee();
8 maja.co
9
10 }
11
12 }
13
Console
No consoles to display at this time.
Writable
Smart Insert
8 : 16

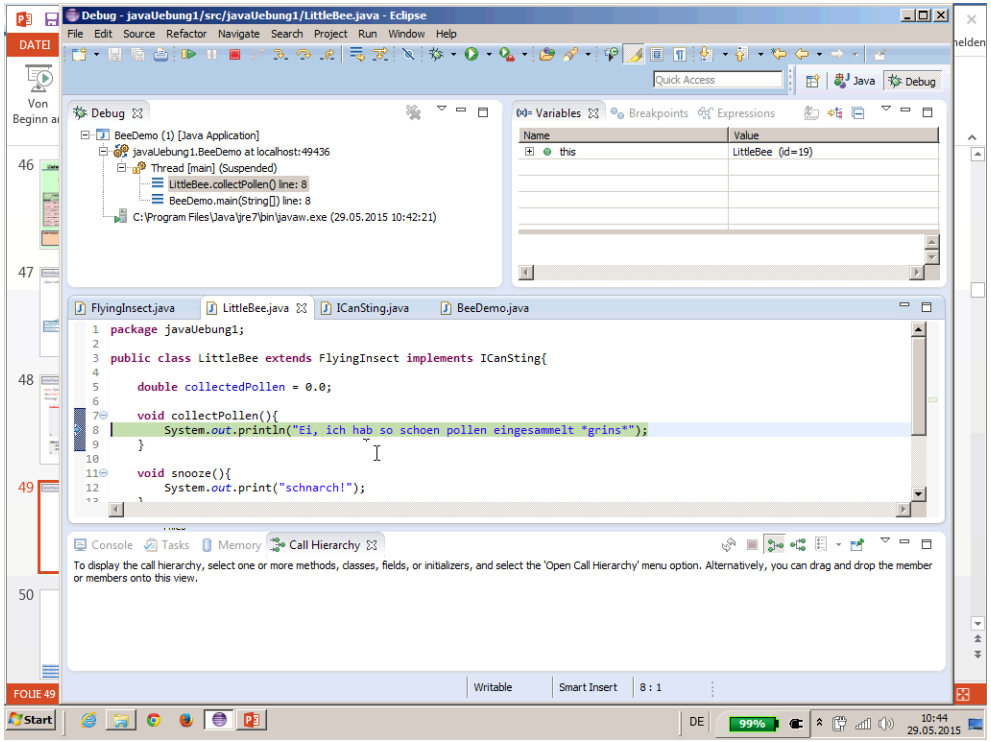

Java - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee();
8
9
10
11 }
12
13 }
14
javaUebung1.LittleBee
Press 'F2' for focus
Console
<terminated> BeeDemo (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:41:26)
Ei, ich hab so schoen pollen eingesammelt *grins*
schnarch!
Writable
Smart Insert
9 : 24

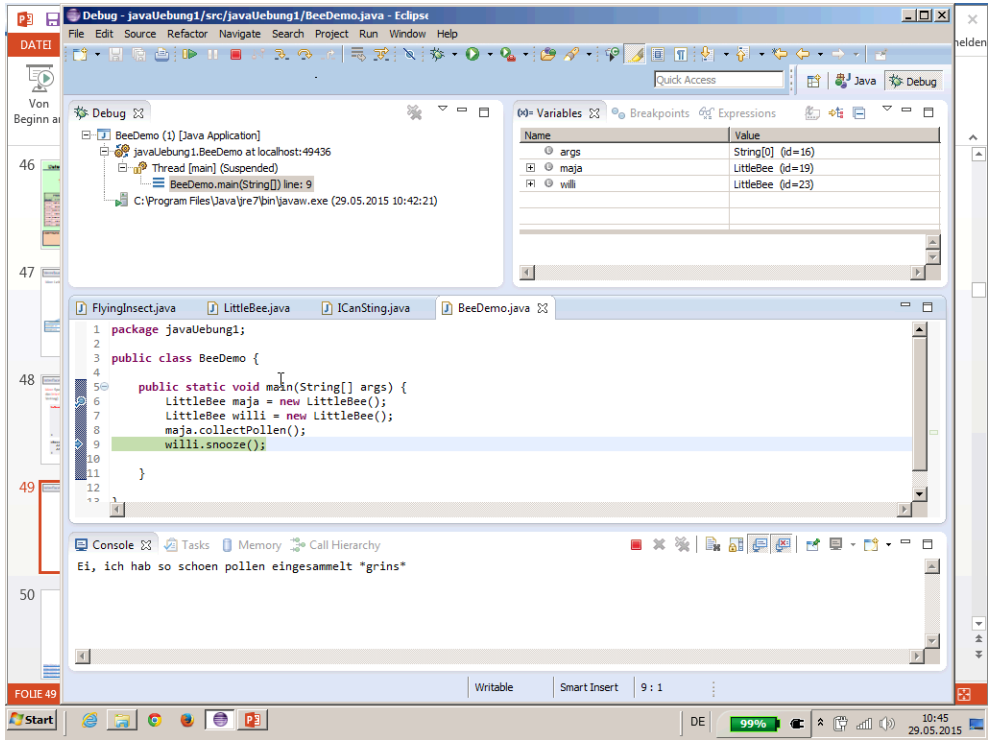

Debug - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49436
Thread [main] (Suspended (breakpoint at line 6 in BeeDemo))
BeeDemo.main(String[]) line: 6
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:42:21)
Name
Value
args
String[0] (id=16)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee willi = new LittleBee();
8 maja.collectPollen();
9 willi.snooze();
10
11 }
12 }
Console
Tasks
Memory
Call Hierarchy
To display the call hierarchy, select one or more methods, classes, fields, or initializers, and select the 'Open Call Hierarchy' menu option. Alternatively, you can drag and drop the member or members onto this view.
Writable
Smart Insert
9 : 24

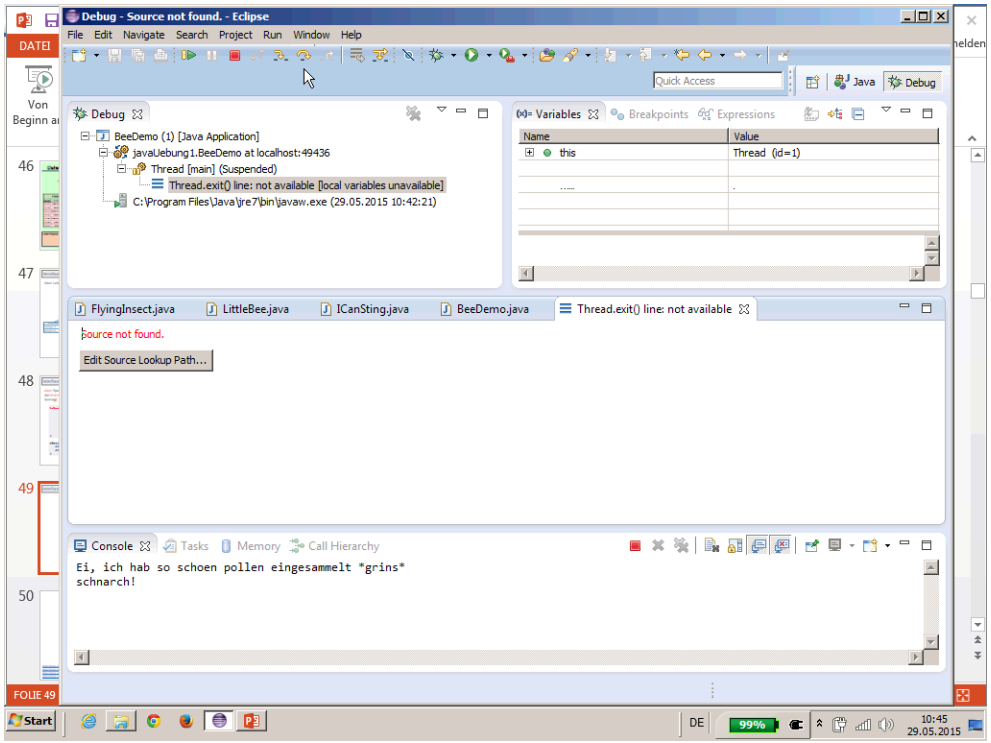

Debug - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49436
Thread [main] (Suspended (breakpoint at line 6 in BeeDemo))
BeeDemo.main(String[]) line: 6
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:42:21)
Name
Value
args
String[0] (id=16)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee willi = new LittleBee();
8
9
10
11 }
12 }
javaUebung1.LittleBee
Press 'F2' for focus
Console
Tasks
Memory
Call Hierarchy
To display the call hierarchy, select one or more methods, classes, fields, or initializers, and select the 'Open Call Hierarchy' menu option. Alternatively, you can drag and drop the member or members onto this view.
Writable
Smart Insert
6 : 1

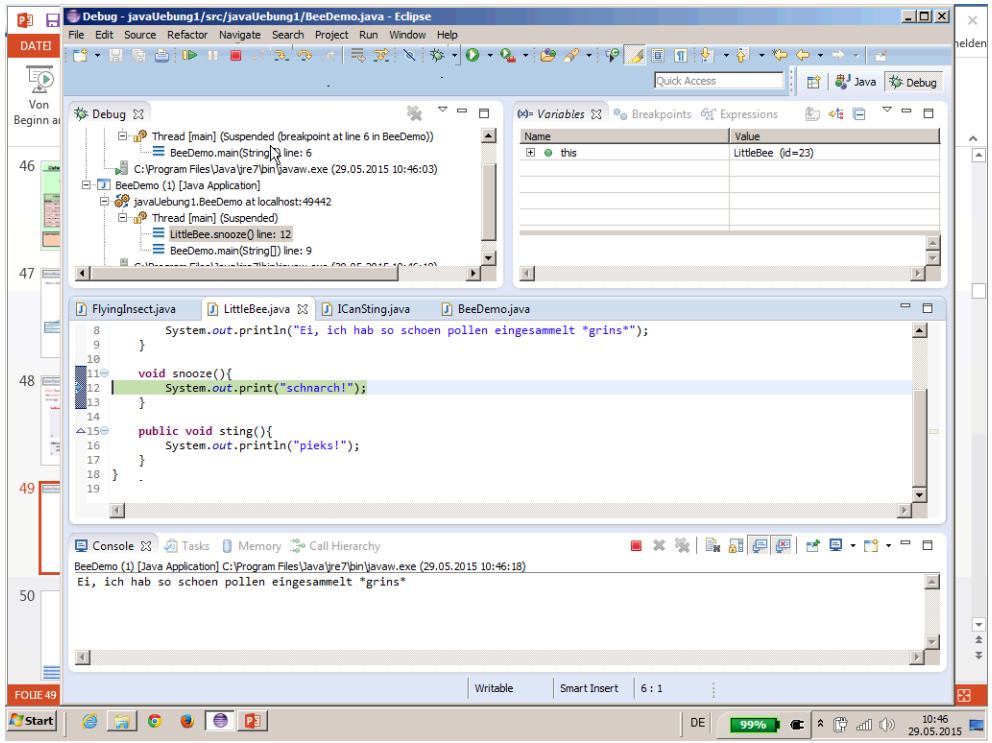

Debug - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49436
Thread [main] (Suspended)
BeeDemo.main(String[]) line: 7
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:42:21)
Name
Value
args
String[0] (id=16)
maja
LittleBee (id=19)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee willi = new LittleBee();
8 maja.collectPollen();
9 willi.snooze();
10
11 }
12 }
Console
Tasks
Memory
Call Hierarchy
To display the call hierarchy, select one or more methods, classes, fields, or initializers, and select the 'Open Call Hierarchy' menu option. Alternatively, you can drag and drop the member or members onto this view.
Writable
Smart Insert
7 : 1

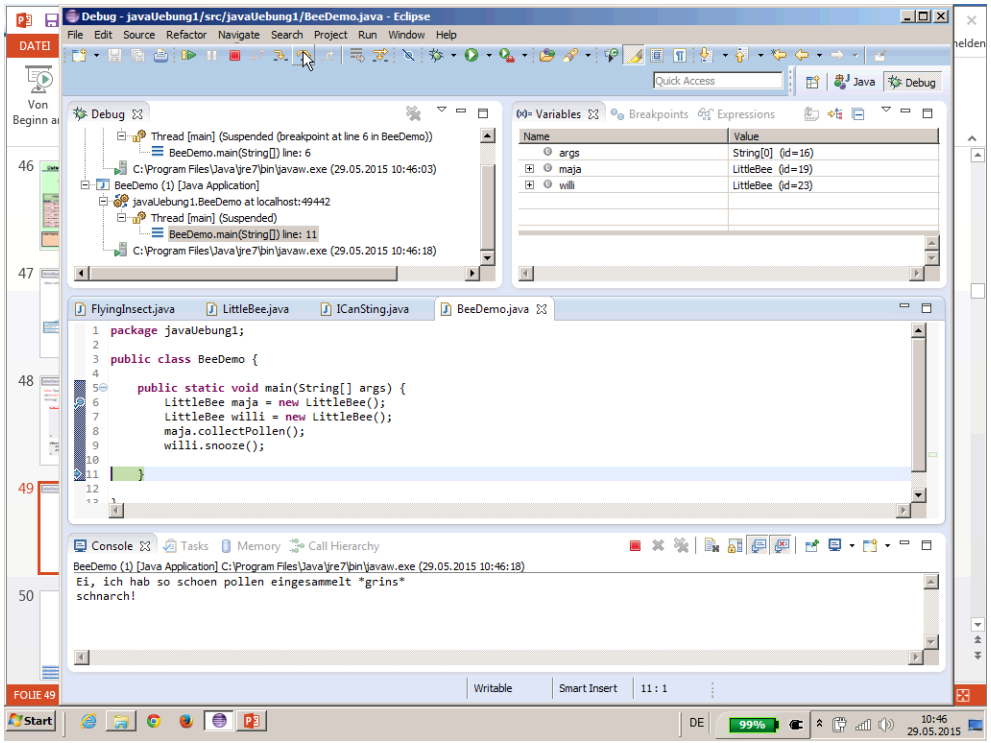

Debug - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49436
Thread [main] (Suspended)
BeeDemo.main(String[]) line: 8
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:42:21)
Name
Value
args
String[0] (id=16)
maja
LittleBee (id=19)
willi
LittleBee (id=23)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee willi = new LittleBee();
8 maja.collectPollen();
9 willi.snooze();
10
11 }
12 }
Console
Tasks
Memory
Call Hierarchy
To display the call hierarchy, select one or more methods, classes, fields, or initializers, and select the 'Open Call Hierarchy' menu option. Alternatively, you can drag and drop the member or members onto this view.
Writable
Smart Insert
8 : 1

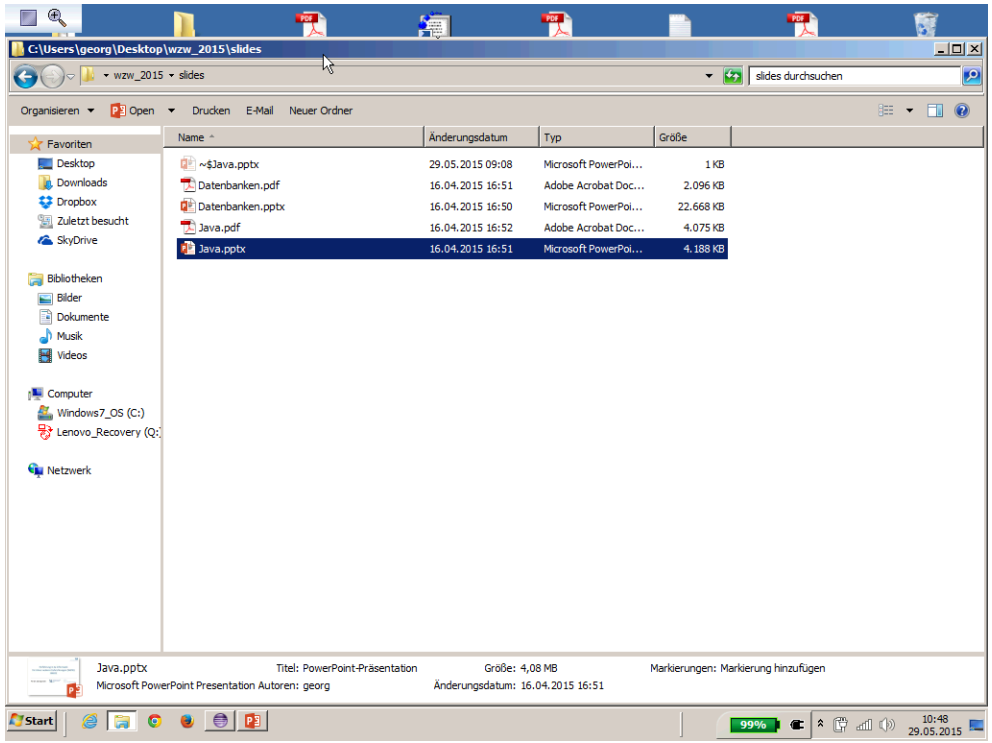

Debug - javaUebung1/src/javaUebung1/LittleBee.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49436
Thread [main] (Suspended)
LittleBee.collectPollen() line: 8
BeeDemo.main(String[]) line: 8
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:42:21)
Variables
Breakpoints
Expressions
Name
Value
this
LittleBee (id=19)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class LittleBee extends FlyingInsect implements ICanSting{
4
5 double collectedPollen = 0.0;
6
7 void collectPollen(){
8 System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
9 }
10
11 void snooze(){
12 System.out.print("schnarch!");
Console
Tasks
Memory
Call Hierarchy
To display the call hierarchy, select one or more methods, classes, fields, or initializers, and select the 'Open Call Hierarchy' menu option. Alternatively, you can drag and drop the member or members onto this view.
Writable
Smart Insert
8 : 1
FOLIE 49
10:44
29.05.2015


Debug - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49436
Thread [main] (Suspended)
BeeDemo.main(String[]) line: 9
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:42:21)
Variables
Breakpoints
Expressions
Name
Value
args
String[0] (id=16)
maja
LittleBee (id=19)
willi
LittleBee (id=23)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee willi = new LittleBee();
8 maja.collectPollen();
9 willi.snooze();
10
11 }
12
Console
Tasks
Memory
Call Hierarchy
Ei, ich hab so schoen pollen eingesammelt *grins*
Writable
Smart Insert
9 : 1
FOLIE 49
10:45
29.05.2015


Debug - Source not found. - Eclipse
File Edit Navigate Search Project Run Window Help
Debug
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49436
Thread [main] (Suspended)
Thread.exit() line: not available [local variables unavailable]
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:42:21)
Variables
Breakpoints
Expressions
Name
Value
this
Thread (id=1)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
Thread.exit() line: not available
ource not found.
Edit Source Lookup Path...
Console
Tasks
Memory
Call Hierarchy
Ei, ich hab so schoen pollen eingesammelt *grins*
schnarch!
FOLIE 49
10:45
29.05.2015


Debug - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Thread [main] (Suspended (breakpoint at line 6 in BeeDemo))
BeeDemo.main(String[]) line: 6
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:46:03)
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49442
Thread [main] (Suspended)
LittleBee.snooze() line: 12
BeeDemo.main(String[]) line: 9
Variables
Breakpoints
Expressions
Name
Value
this
LittleBee (id=23)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
8 System.out.println("Ei, ich hab so schoen pollen eingesammelt *grins*");
9 }
10
11 void snooze(){
12 System.out.print("schnarch!");
13 }
14
15 public void sting(){
16 System.out.println("pieks!");
17 }
18 }
19
Console
Tasks
Memory
Call Hierarchy
BeeDemo (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:46:18)
Ei, ich hab so schoen pollen eingesammelt *grins*
Writable
Smart Insert
6 : 1
FOLIE 49
10:46
29.05.2015


Debug - javaUebung1/src/javaUebung1/BeeDemo.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Thread [main] (Suspended (breakpoint at line 6 in BeeDemo))
BeeDemo.main(String[]) line: 6
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:46:03)
BeeDemo (1) [Java Application]
javaUebung1.BeeDemo at localhost:49442
Thread [main] (Suspended)
BeeDemo.main(String[]) line: 11
C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:46:18)
Variables
Breakpoints
Expressions
Name
Value
args
String[0] (id=16)
maja
LittleBee (id=19)
willi
LittleBee (id=23)
FlyingInsect.java
LittleBee.java
ICanSting.java
BeeDemo.java
1 package javaUebung1;
2
3 public class BeeDemo {
4
5 public static void main(String[] args) {
6 LittleBee maja = new LittleBee();
7 LittleBee willi = new LittleBee();
8 maja.collectPollen();
9 willi.snooze();
10
11 }
12
Console
Tasks
Memory
Call Hierarchy
BeeDemo (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (29.05.2015 10:46:18)
Ei, ich hab so schoen pollen eingesammelt *grins*
schnarch!
Writable
Smart Insert
11 : 1
FOLIE 49
Start
DE
99%
10:46
29.05.2015


C:\Users\georg\Desktop\wzw_2015\slides
wzw_2015 slides
slides durchsuchen
Organisieren
Open
Drucken
E-Mail
Neuer Ordner
Favoriten
Desktop
Downloads
Dropbox
Zuletzt besucht
SkyDrive
Bibliotheken
Bilder
Dokumente
Musik
Videos
Computer
Windows7_OS (C:)
Lenovo_Recovery (Q:)
Netzwerk
Name
Änderungsdatum
Typ
Größe
~$Java.pptx 29.05.2015 09:08 Microsoft PowerPoi... 1 KB
Datenbanken.pdf 16.04.2015 16:51 Adobe Acrobat Doc... 2.096 KB
Datenbanken.pptx 16.04.2015 16:50 Microsoft PowerPoi... 22.668 KB
Java.pdf 16.04.2015 16:52 Adobe Acrobat Doc... 4.075 KB
Java.pptx 16.04.2015 16:51 Microsoft PowerPoi... 4.188 KB
Java.pptx Titel: PowerPoint-Präsentation Größe: 4,08 MB Markierungen: Markierung hinzufügen
Microsoft PowerPoint Presentation Autoren: georg Änderungsdatum: 16.04.2015 16:51
Start
99%
10:48
29.05.2015