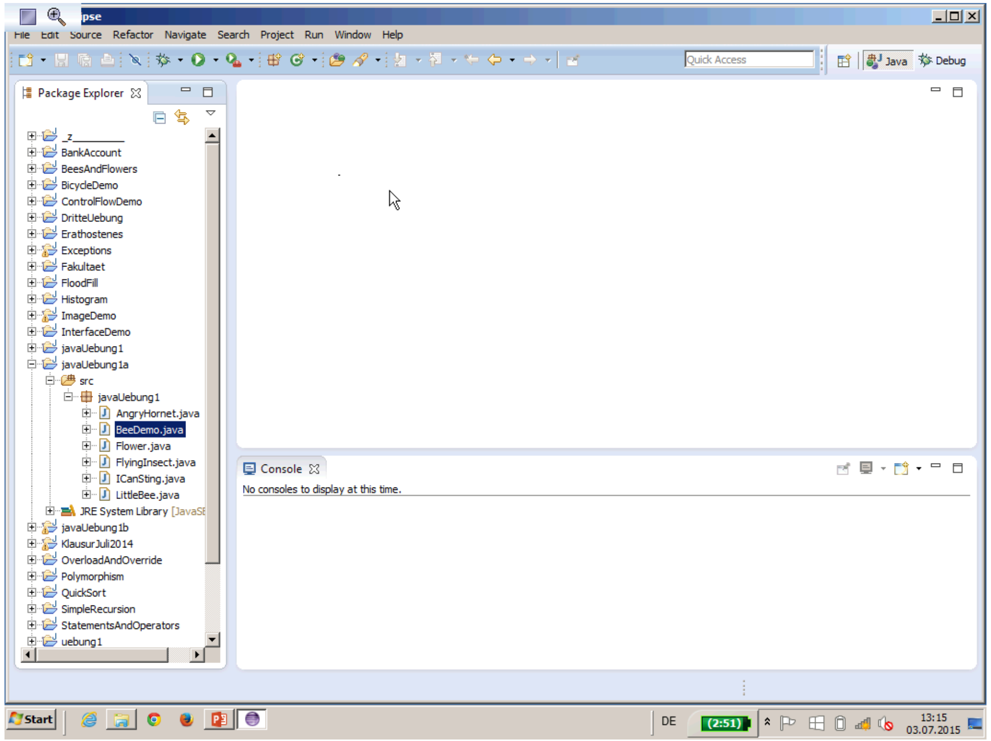
Script generated by TTT

Title: groh: profile1 (03.07.2015)

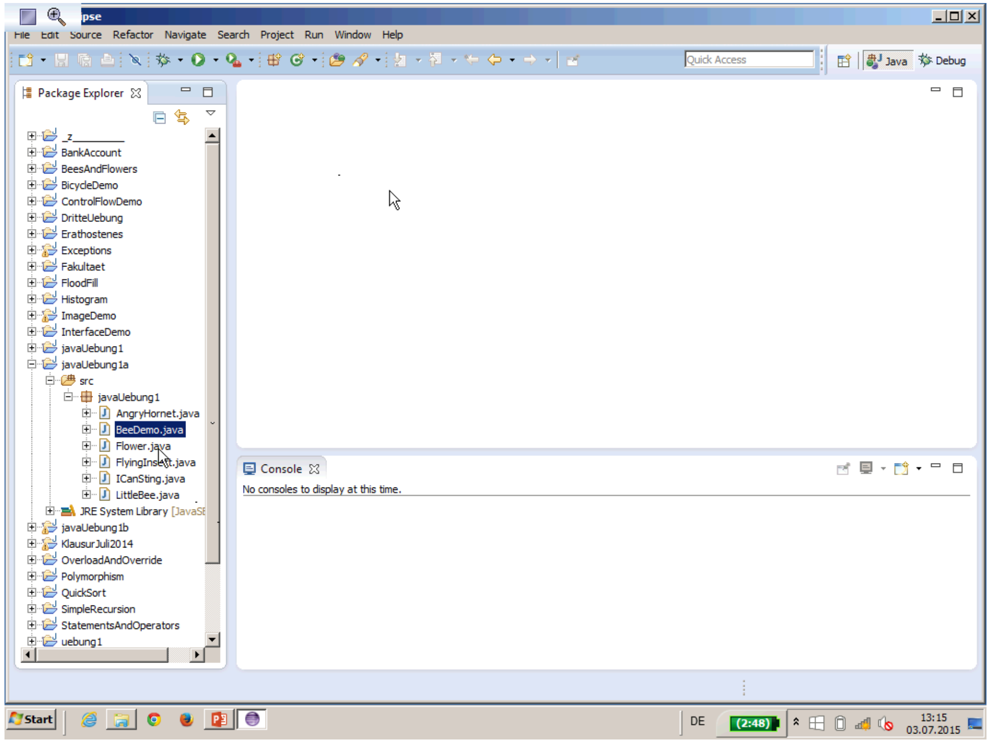
Date: Fri Jul 03 13:15:20 CEST 2015

Duration: 101:01 min

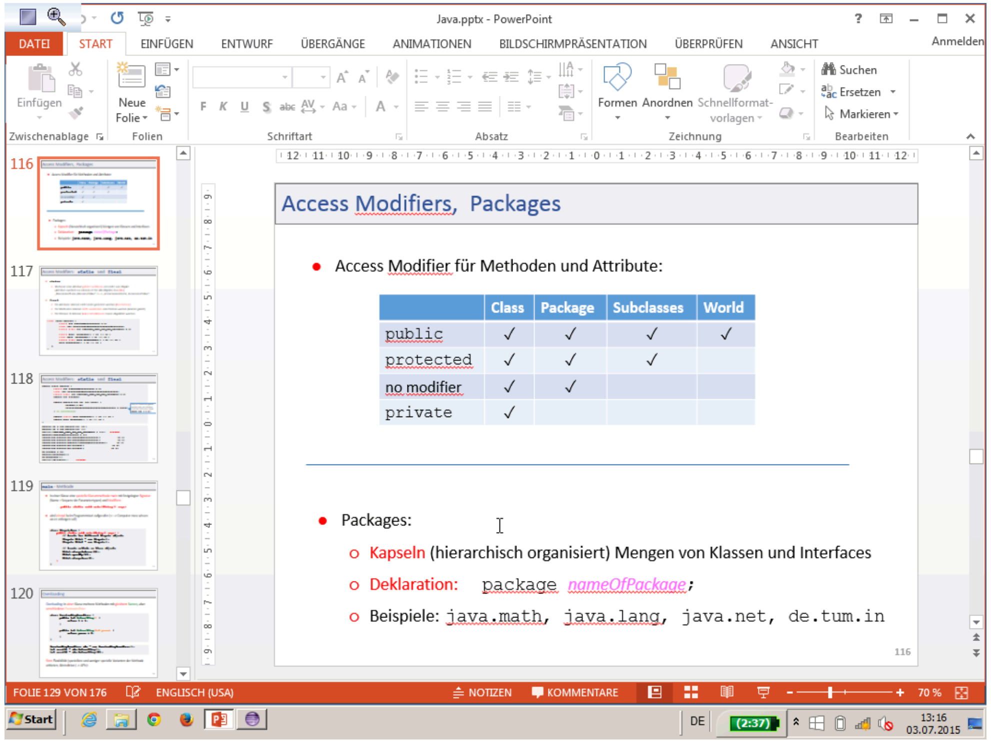
Pages: 148



No consoles to display at this time.



No consoles to display at this time.



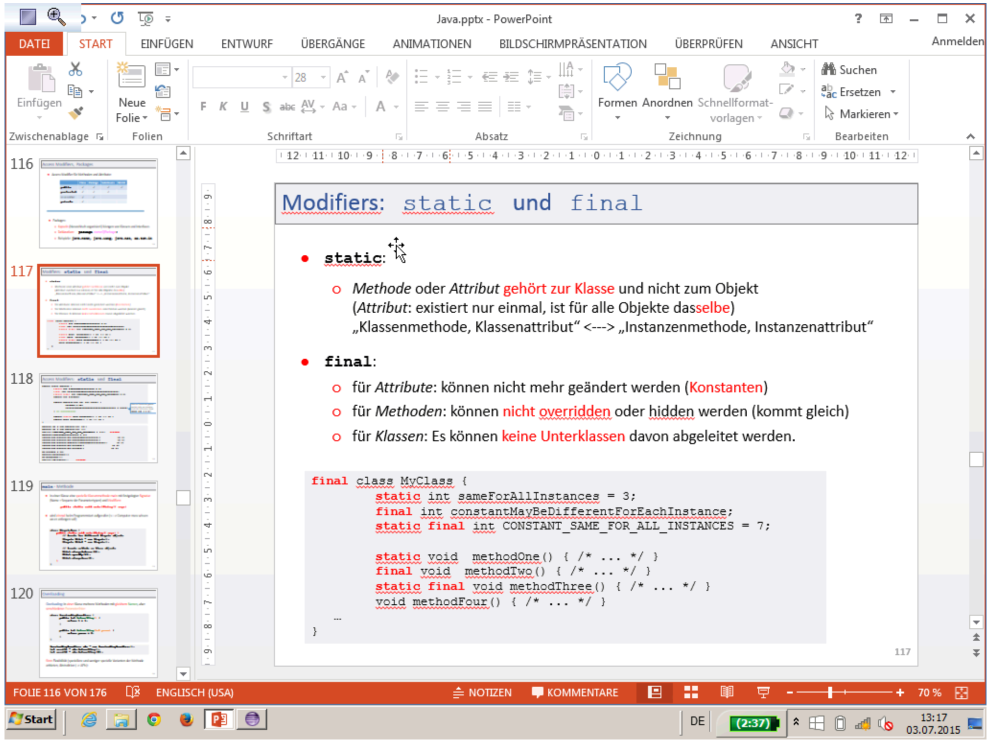
Access Modifiers, Packages

- Access Modifier für Methoden und Attribute:

	Class	Package	Subclasses	World
public	✓	✓	✓	✓
protected	✓	✓	✓	
no modifier	✓	✓		
private	✓			

- Packages:
  - Kapseln (hierarchisch organisiert) Mengen von Klassen und Interfaces
  - Deklaration: `package nameOfPackage;`
  - Beispiele: `java.math, java.lang, java.net, de.tum.in`

116

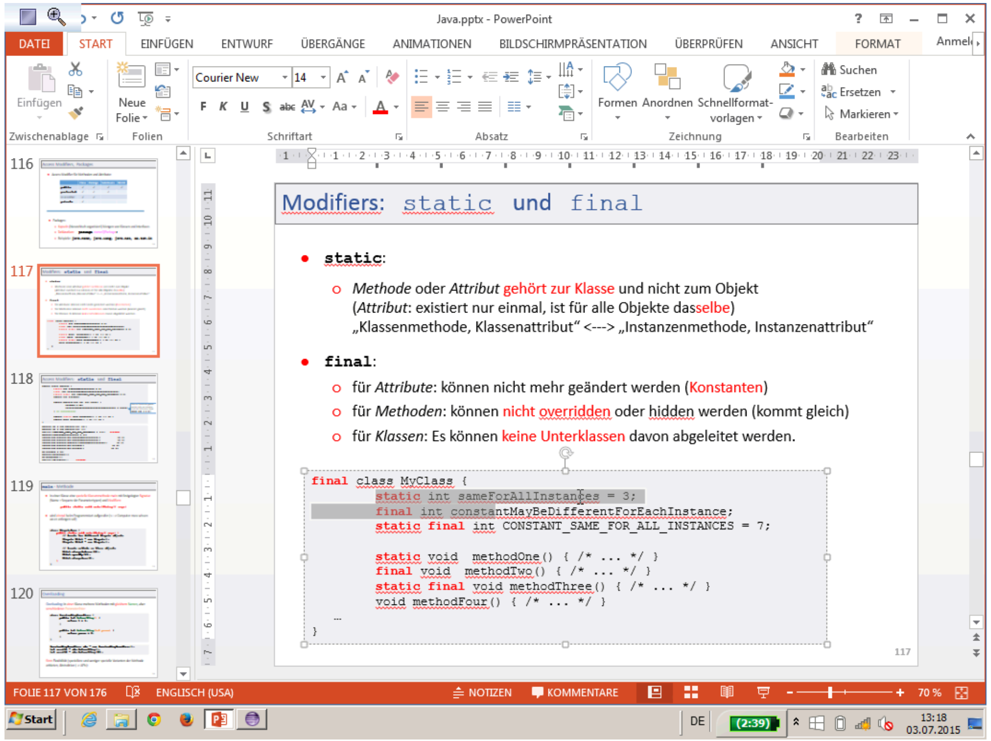


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }
    …
}
```

117

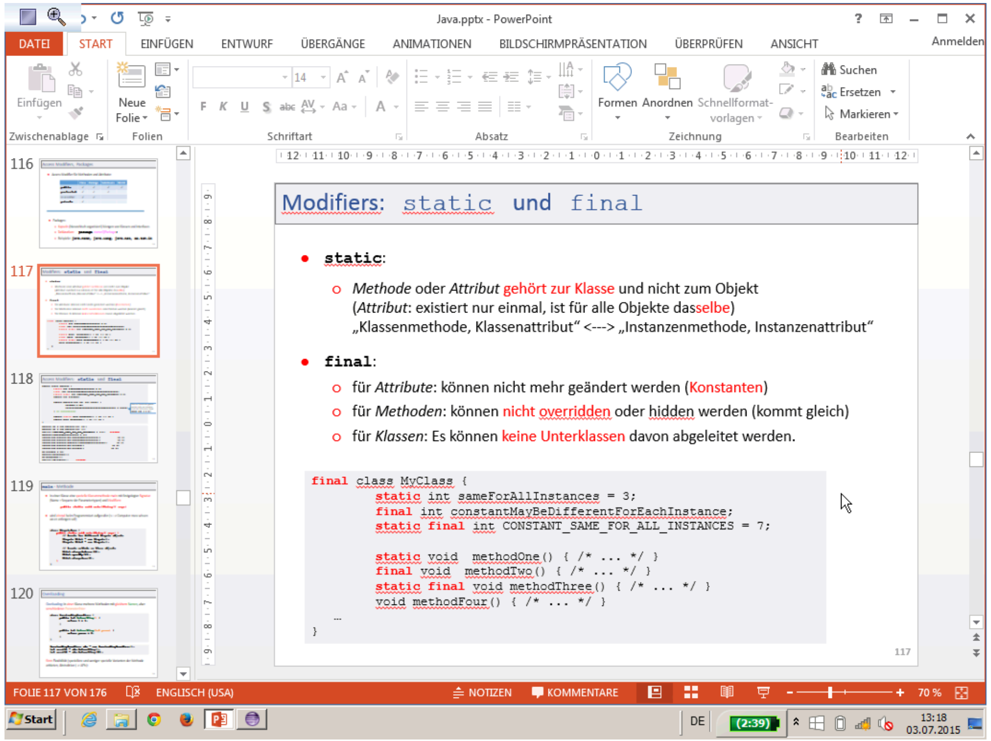


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }
    …
}
```

117

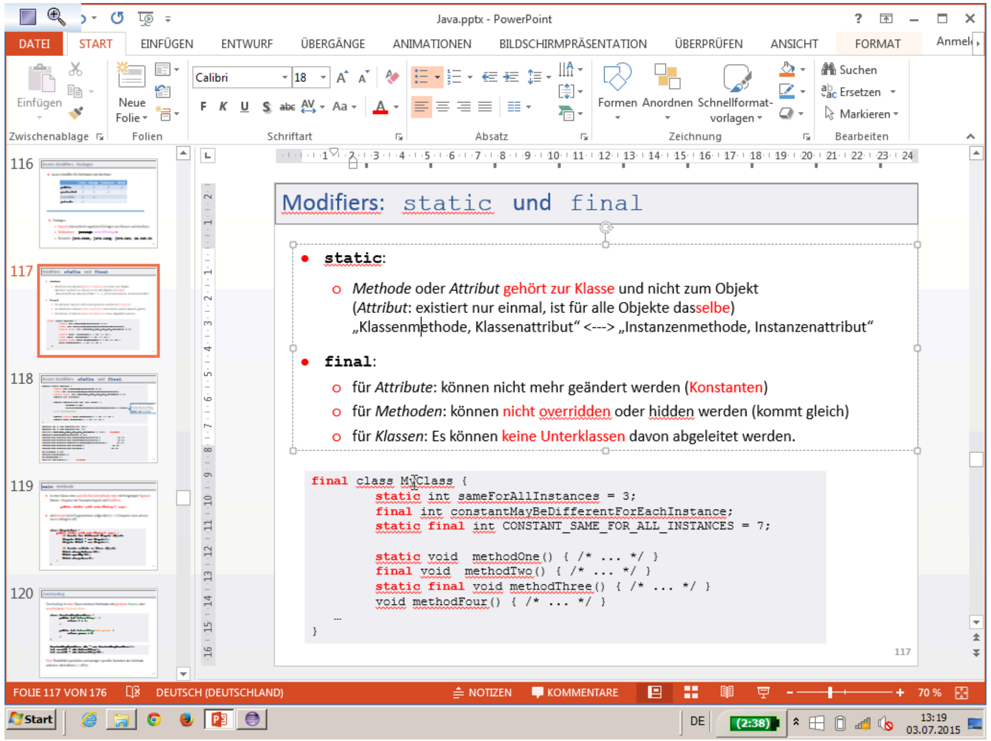


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }
    …
}
```

117

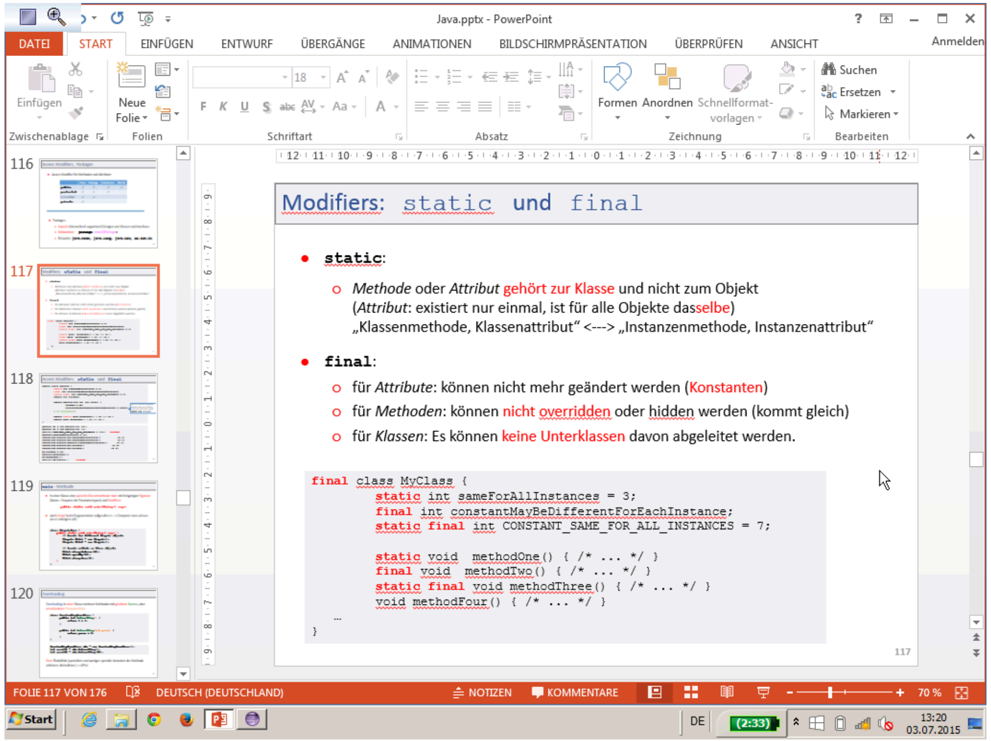


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }
    …
}
```

117

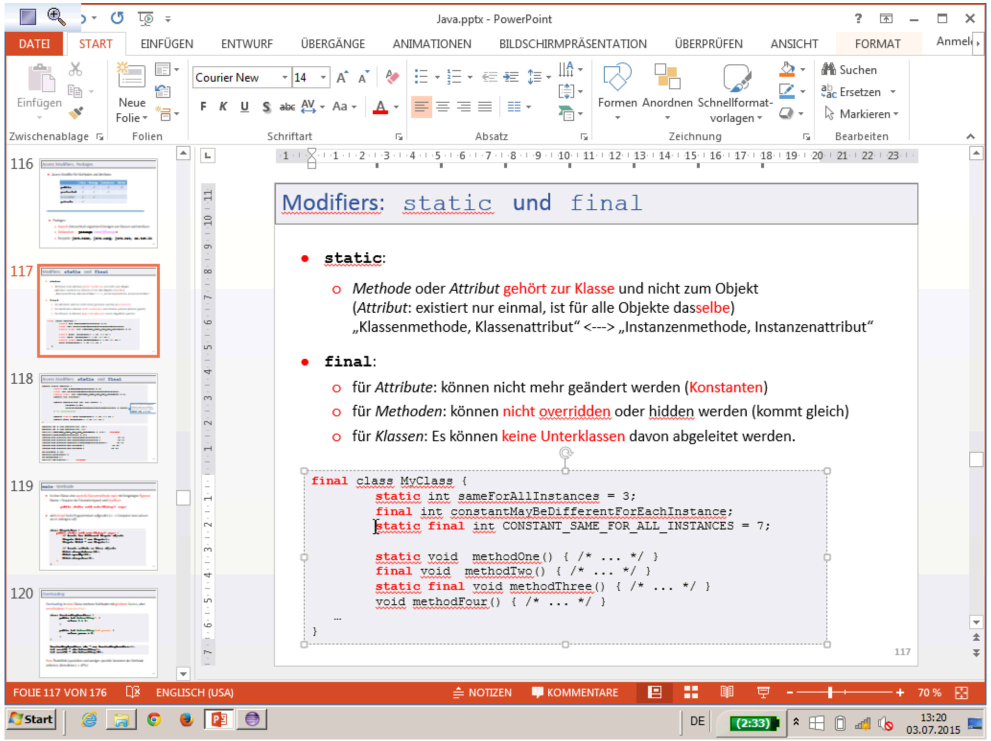


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }

        ...
}
```

117

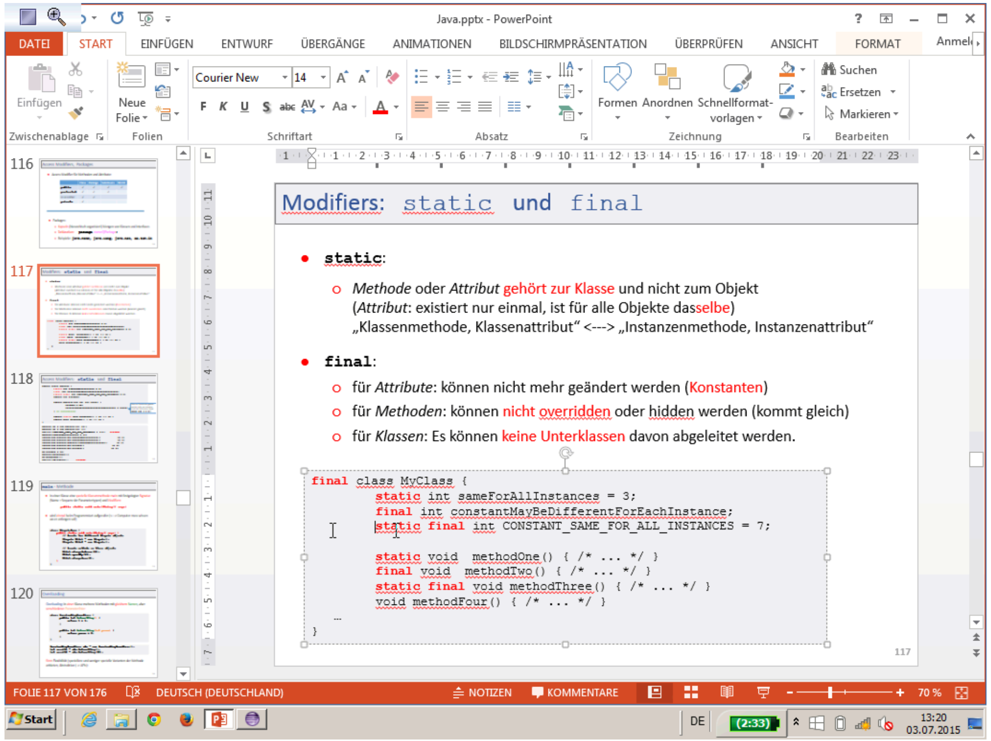


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }

        ...
}
```

117

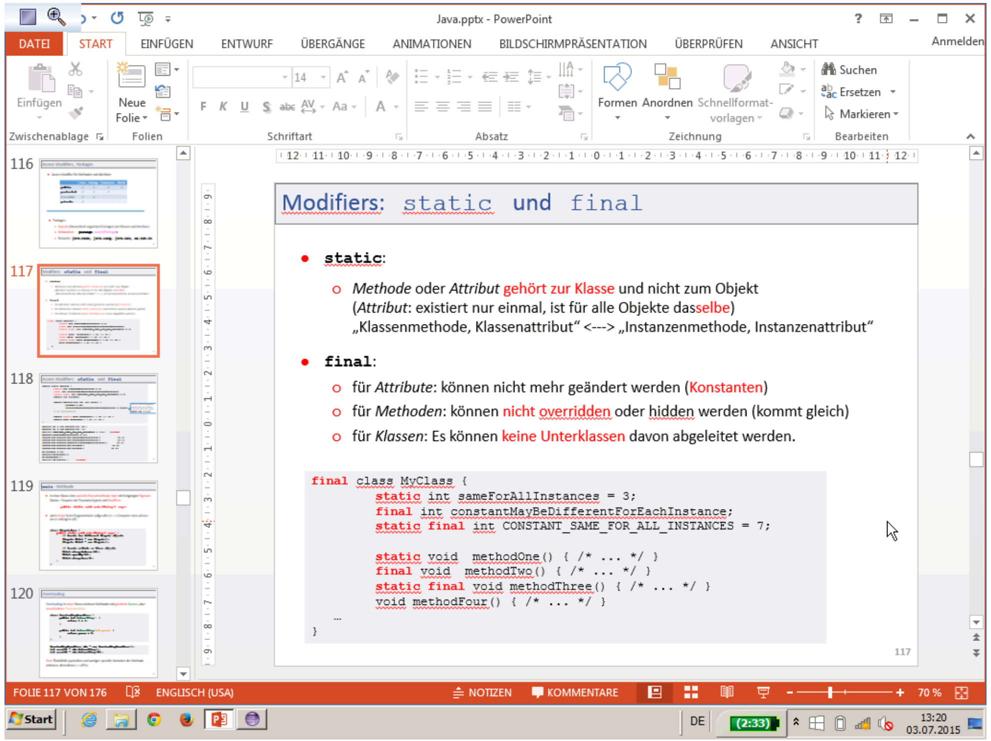


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }

        ...
}
```

117

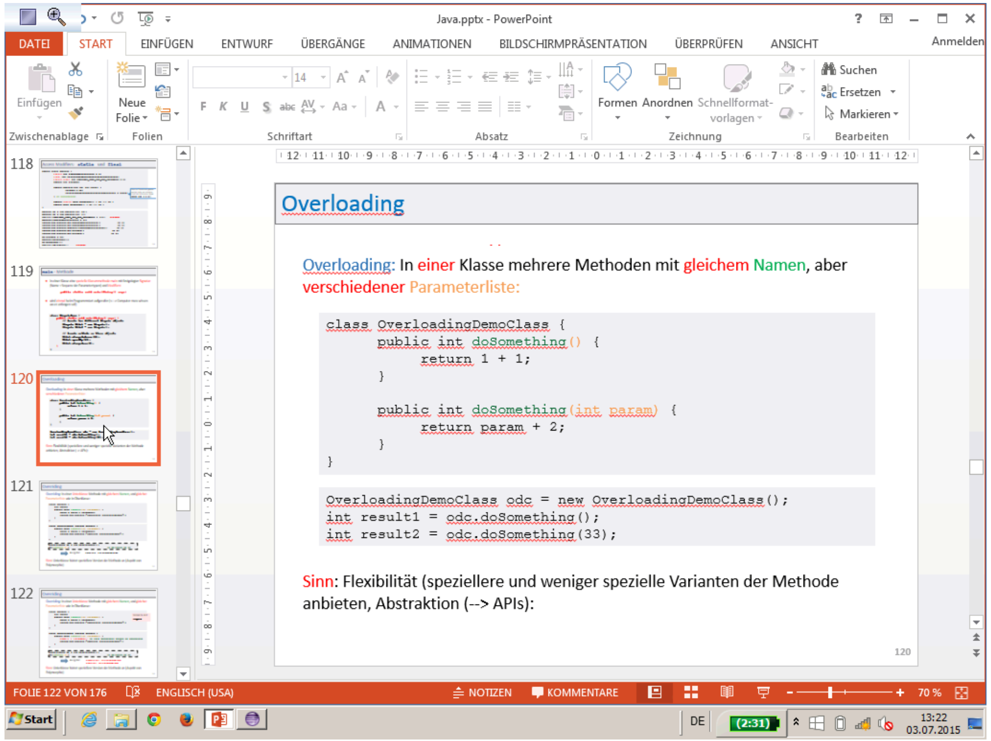


Modifiers: static und final

- **static**:
  - *Methode* oder *Attribut* gehört zur Klasse und nicht zum Objekt (*Attribut*: existiert nur einmal, ist für alle Objekte dasselbe) „Klassenmethode, Klassenattribut“ <---> „Instanzenmethode, Instanzenattribut“
- **final**:
  - für *Attribute*: können nicht mehr geändert werden (Konstanten)
  - für *Methoden*: können nicht overridden oder hidden werden (kommt gleich)
  - für *Klassen*: Es können keine Unterklassen davon abgeleitet werden.

```
final class MyClass {
        static int sameForAllInstances = 3;
        final int constantMayBeDifferentForEachInstance;
        static final int CONSTANT_SAME_FOR_ALL_INSTANCES = 7;

        static void  methodOne() { /* ... */ }
        final void  methodTwo() { /* ... */ }
        static final void methodThree() { /* ... */ }
        void methodFour() { /* ... */ }

        ...
}
```

117


Java.pptx - PowerPoint
DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHIRMPRÄSENTATION ÜBERPRÜFEN ANSICHT
Anmelden
Einfügen
Neue Folie
Formen Anordnen Schnellformatvorlagen
Suchen
Ersetzen
Markieren
Zwischenablage Folien Schriftart Absatz Zeichnung Bearbeiten
118
119
120
121
122
FOLIE 122 VON 176 ENGLISCH (USA) NOTIZEN KOMMENTARE 70 %
Start
DE (2:31) 13:22 03.07.2015

Overloading

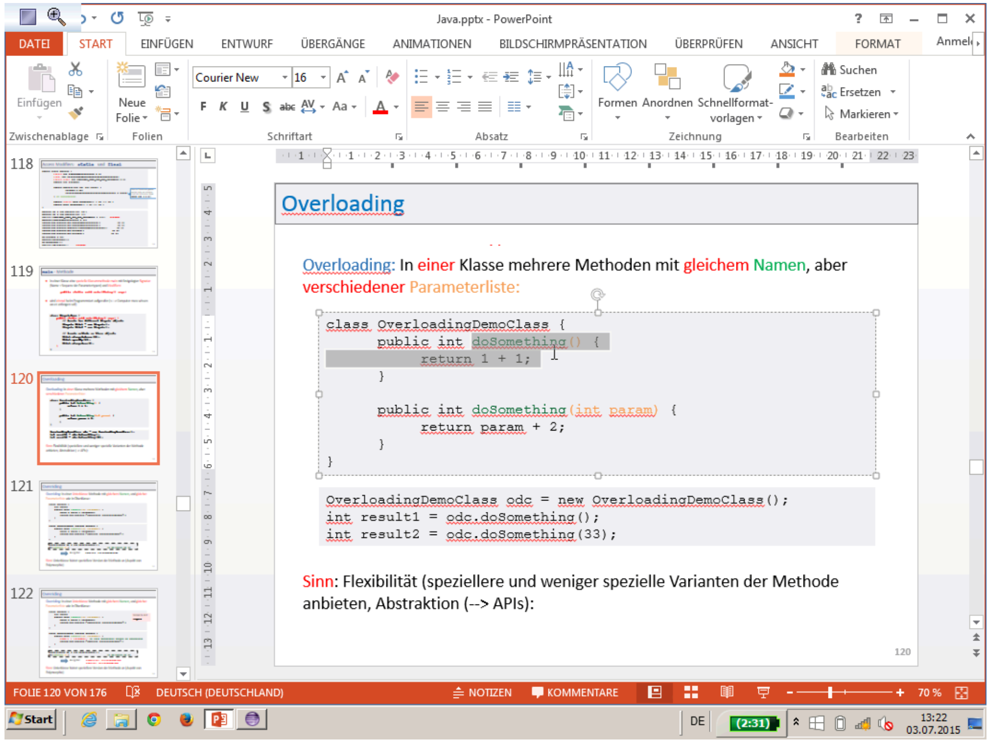
Overloading: In einer Klasse mehrere Methoden mit gleichem Namen, aber verschiedener Parameterliste:

```
class OverloadingDemoClass {
       public int doSomething() {
             return 1 + 1;
       }

       public int doSomething(int param) {
             return param + 2;
       }
}
```

```
OverloadingDemoClass odc = new OverloadingDemoClass();
int result1 = odc.doSomething();
int result2 = odc.doSomething(33);
```

Sinn: Flexibilität (speziellere und weniger spezielle Varianten der Methode anbieten, Abstraktion (--> APIs):

120


Java.pptx - PowerPoint
DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHIRMPRÄSENTATION ÜBERPRÜFEN ANSICHT FORMAT
Anmel
Courier New 16
Einfügen
Neue Folie
Formen Anordnen Schnellformatvorlagen
Suchen
Ersetzen
Markieren
Zwischenablage Folien Schriftart Absatz Zeichnung Bearbeiten
118
119
120
121
122
FOLIE 120 VON 176 DEUTSCH (DEUTSCHLAND) NOTIZEN KOMMENTARE 70 %
Start
DE (2:31) 13:22 03.07.2015

Overloading

Overloading: In einer Klasse mehrere Methoden mit gleichem Namen, aber verschiedener Parameterliste:

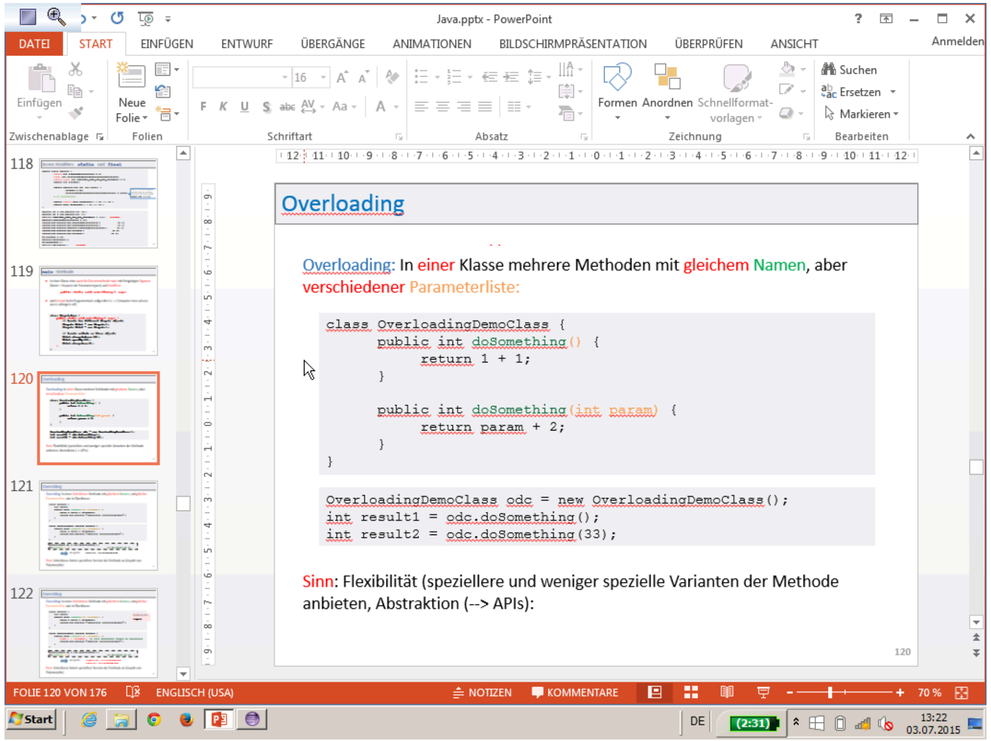
class OverloadingDemoClass {
       public int doSomething() {
             return 1 + 1;
       }

       public int doSomething(int param) {
             return param + 2;
       }
}

OverloadingDemoClass odc = new OverloadingDemoClass();
int result1 = odc.doSomething();
int result2 = odc.doSomething(33);

Sinn: Flexibilität (speziellere und weniger spezielle Varianten der Methode anbieten, Abstraktion (--> APIs):

120


Java.pptx - PowerPoint
DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHIRMPRÄSENTATION ÜBERPRÜFEN ANSICHT
Anmelden
16
Einfügen
Neue Folie
Formen Anordnen Schnellformatvorlagen
Suchen
Ersetzen
Markieren
Zwischenablage Folien Schriftart Absatz Zeichnung Bearbeiten
118
119
120
121
122
FOLIE 120 VON 176 ENGLISCH (USA) NOTIZEN KOMMENTARE 70 %
Start
DE (2:31) 13:22 03.07.2015

Overloading

Overloading: In einer Klasse mehrere Methoden mit gleichem Namen, aber verschiedener Parameterliste:

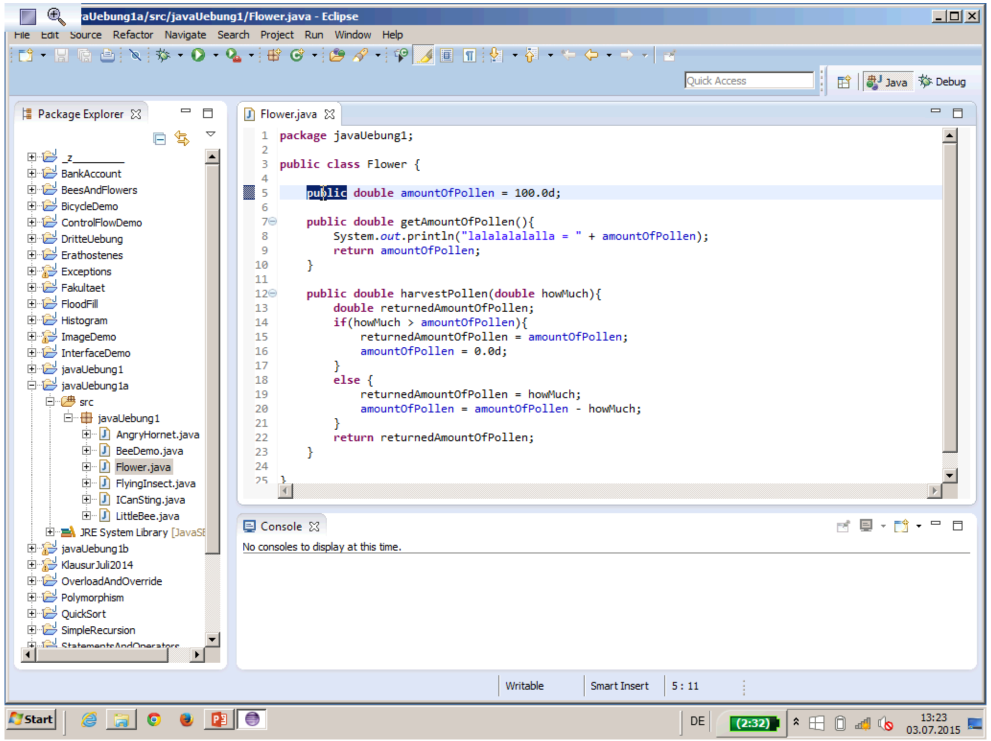
class OverloadingDemoClass {
       public int doSomething() {
             return 1 + 1;
       }

       public int doSomething(int param) {
             return param + 2;
       }
}

OverloadingDemoClass odc = new OverloadingDemoClass();
int result1 = odc.doSomething();
int result2 = odc.doSomething(33);

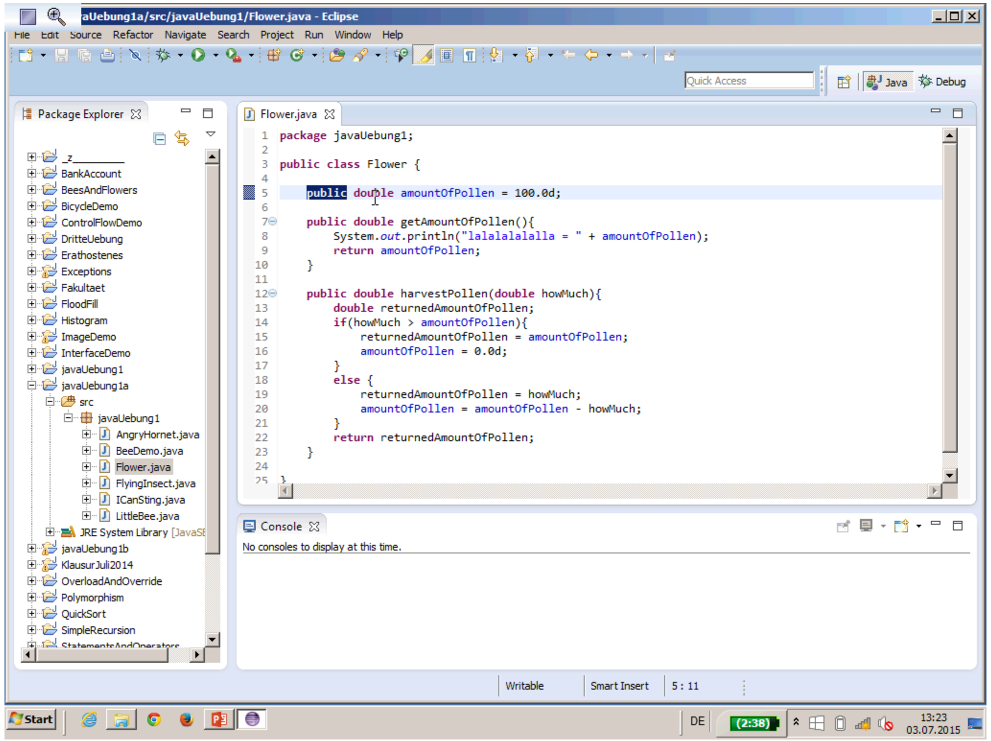
Sinn: Flexibilität (speziellere und weniger spezielle Varianten der Methode anbieten, Abstraktion (--> APIs):

120



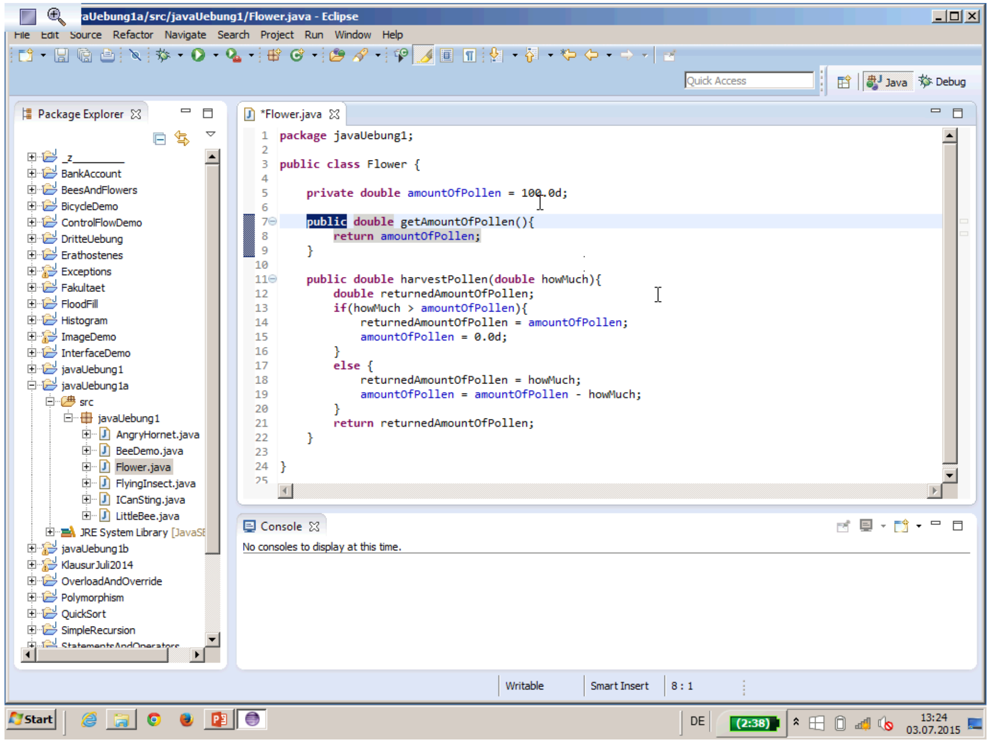
```
package javaUebung1;

public class Flower {

    public double amountOfPollen = 100.0d;

    public double getAmountOfPollen(){
        System.out.println("lalalalalalla = " + amountOfPollen);
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }

}
```

```
No consoles to display at this time.
```



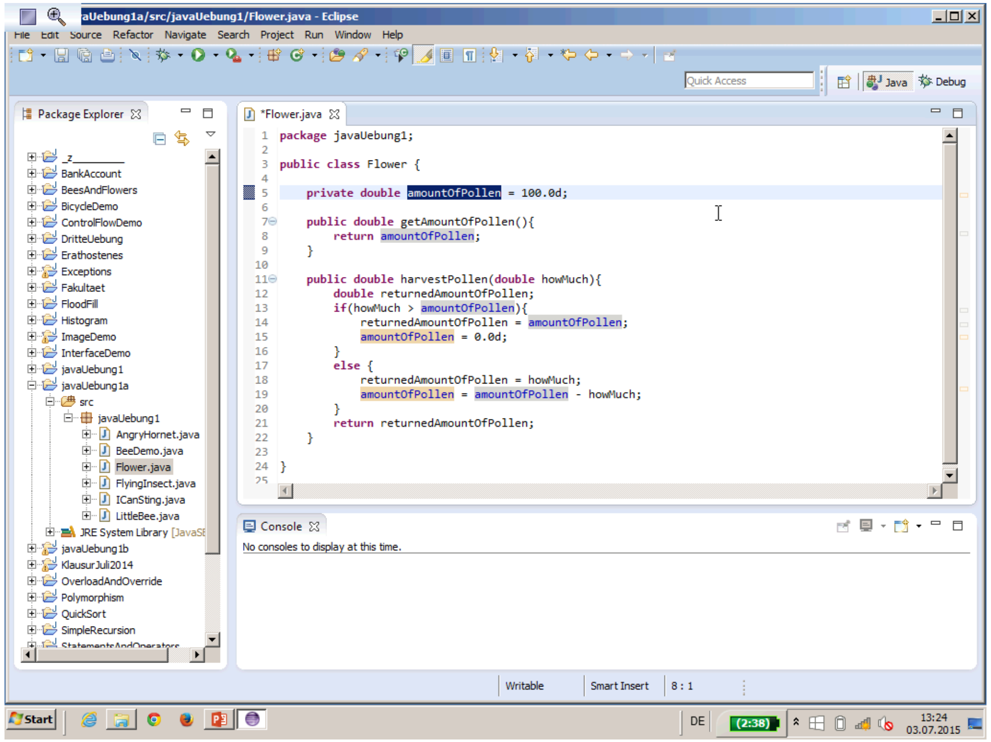
```java
package javaUebung1;

public class Flower {

    public double amountOfPollen = 100.0d;

    public double getAmountOfPollen(){
        System.out.println("lalalalalalla = " + amountOfPollen);
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }

}
```

No consoles to display at this time.



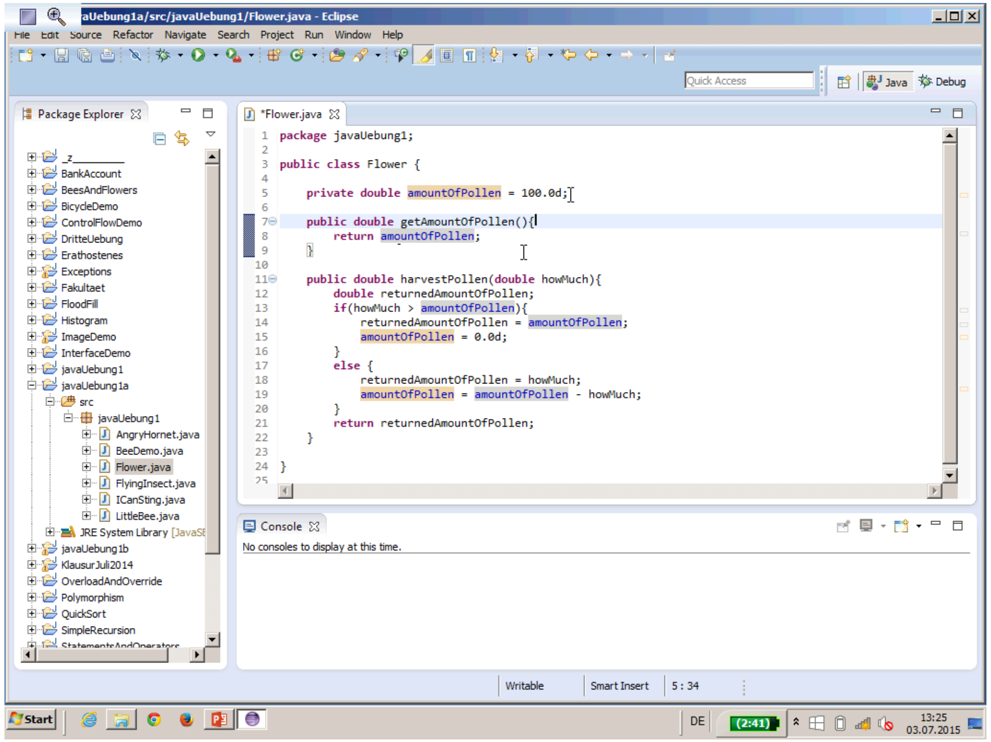
```java
package javaUebung1;

public class Flower {

    private double amountOfPollen = 100.0d;

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }

}
```

No consoles to display at this time.



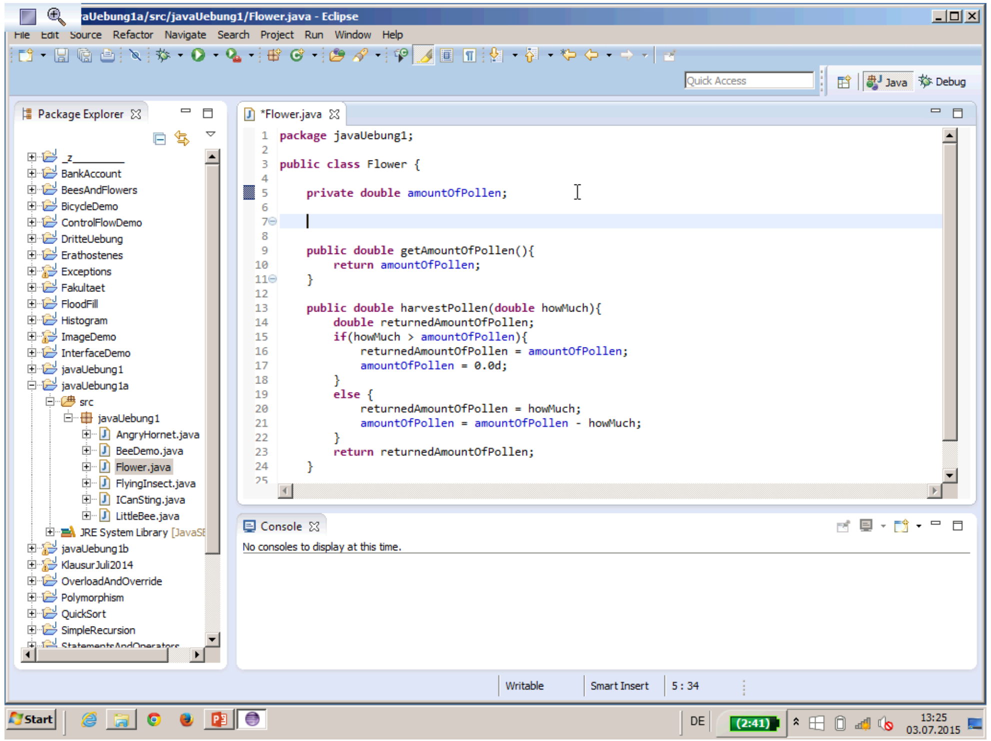
```java
package javaUebung1;

public class Flower {

    private double amountOfPollen = 100.0d;

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }

}
```

No consoles to display at this time.



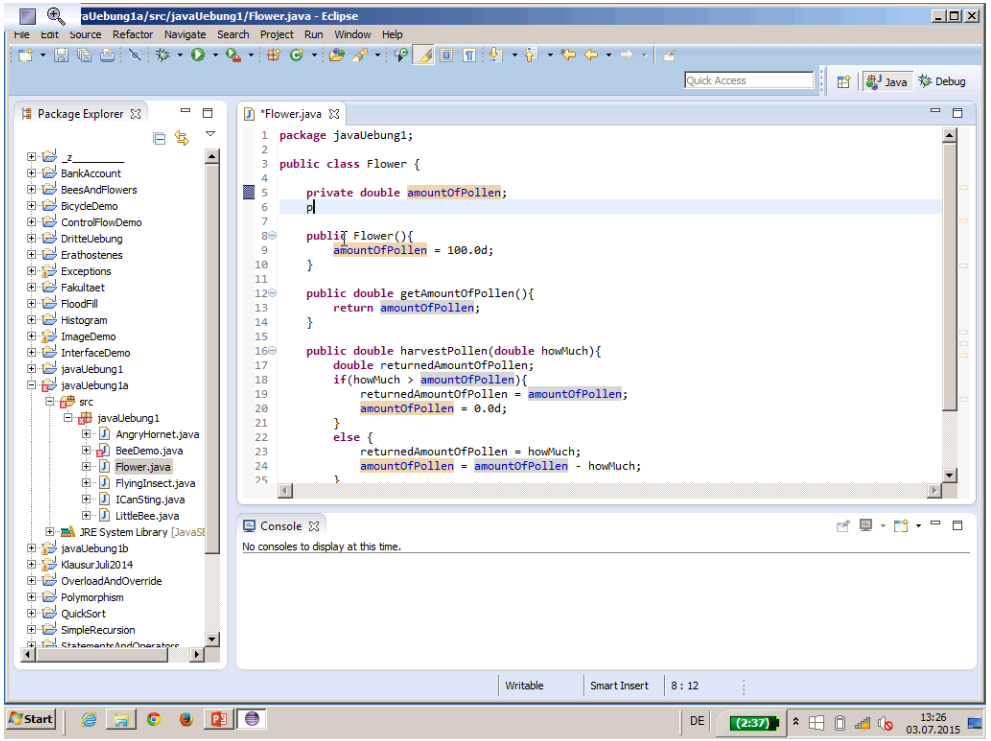
```java
package javaUebung1;

public class Flower {

    private double amountOfPollen = 100.0d;

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }

}
```

No consoles to display at this time.



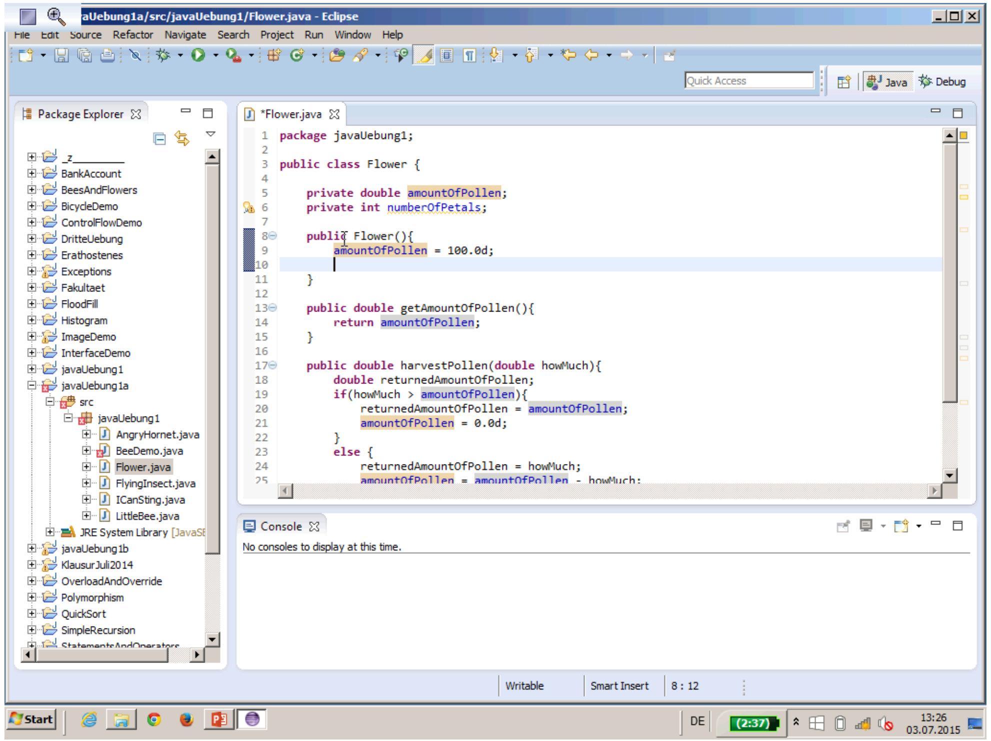
```java
package javaUebung1;

public class Flower {

    private double amountOfPollen;

    

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }
```

No consoles to display at this time.



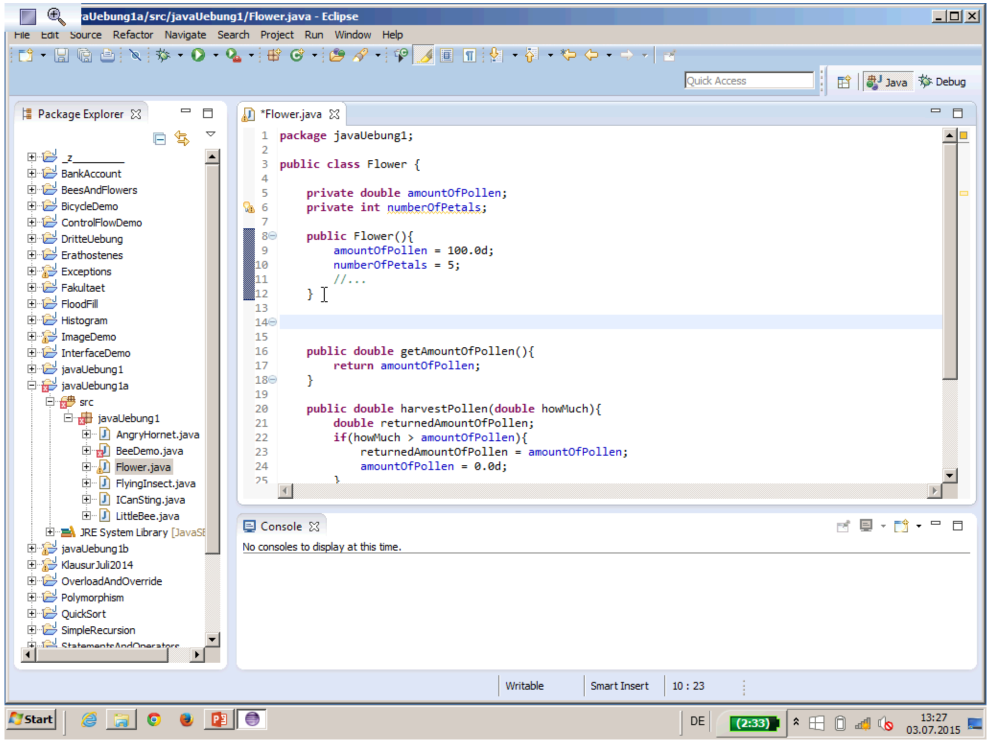
```java
package javaUebung1;

public class Flower {

    private double amountOfPollen;
    p

    public Flower(){
        amountOfPollen = 100.0d;
    }

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
```

No consoles to display at this time.



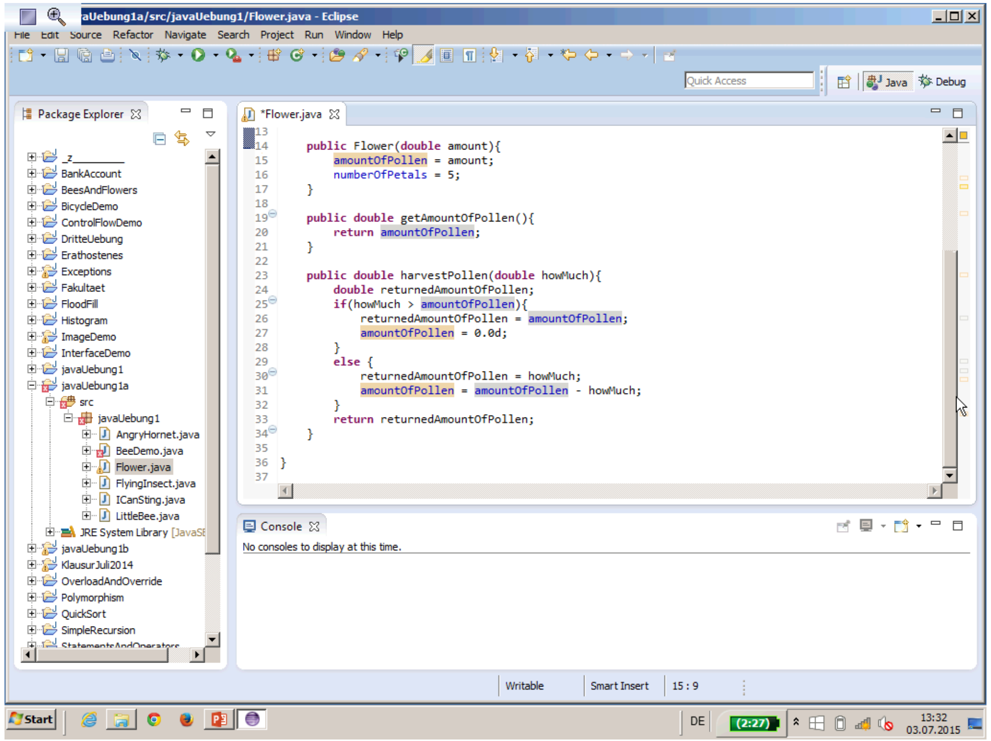
```java
package javaUebung1;

public class Flower {

    private double amountOfPollen;
    private int numberOfPetals;

    public Flower(){
        amountOfPollen = 100.0d;
        
    }

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
```

No consoles to display at this time.



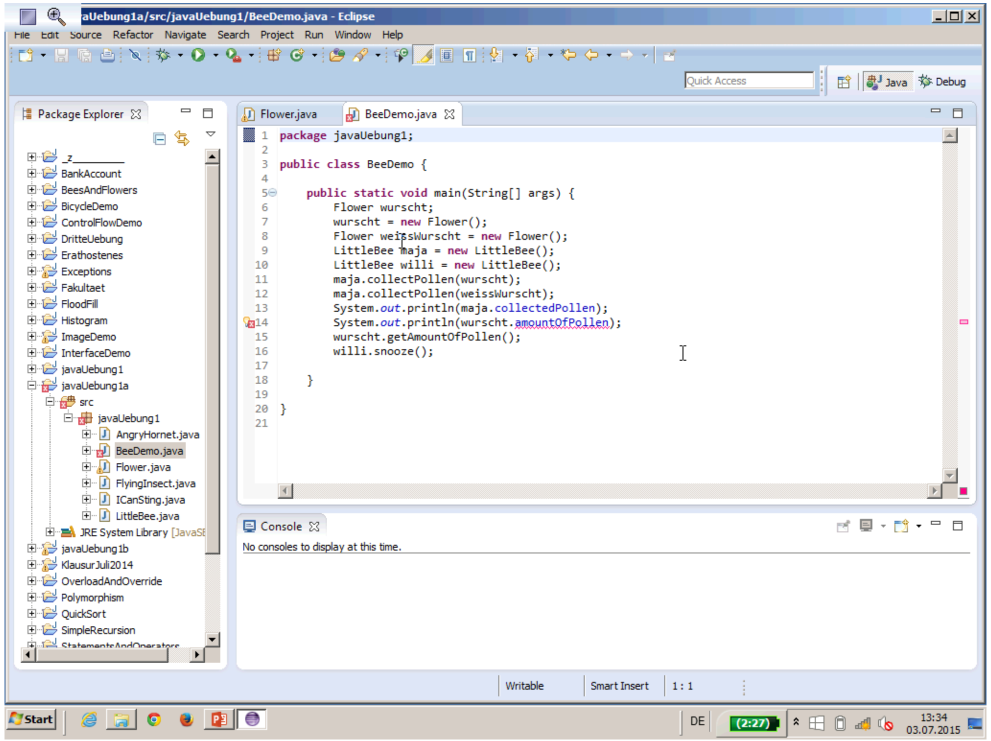
```java
package javaUebung1;

public class Flower {

    private double amountOfPollen;
    private int numberOfPetals;

    public Flower(){
        amountOfPollen = 100.0d;
        numberOfPetals = 5;
        //...
    }



    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
```

No consoles to display at this time.



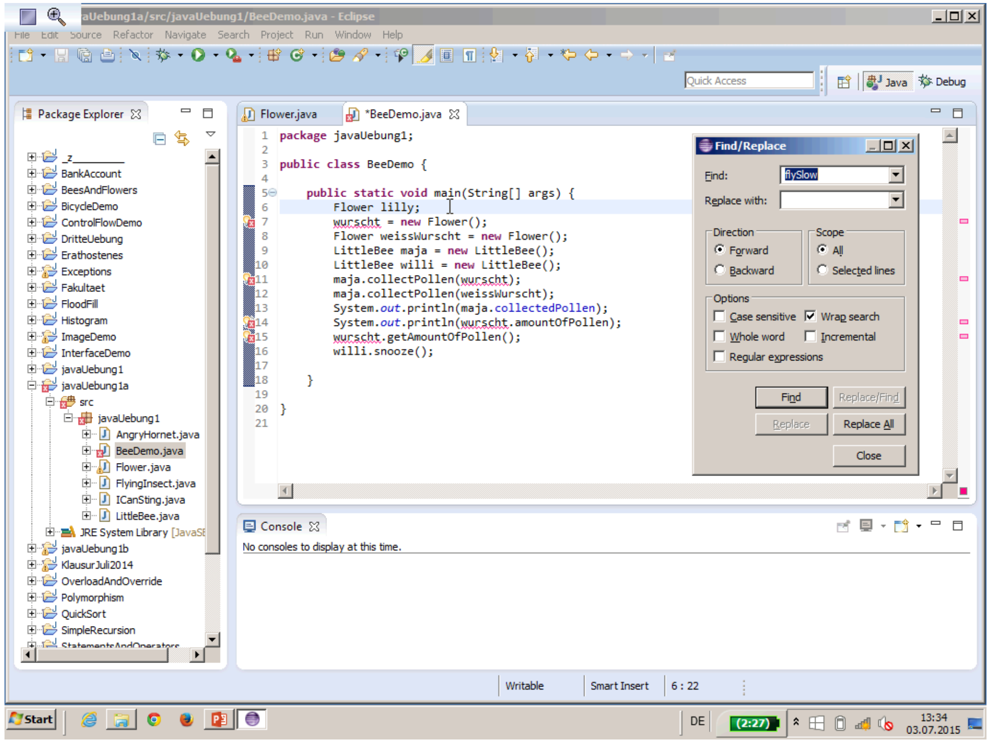
```java

    public Flower(double amount){
        amountOfPollen = amount;
        numberOfPetals = 5;
    }

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }

}
```

No consoles to display at this time.



```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower wurscht;
        wurscht = new Flower();
        Flower weissWurscht = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(wurscht);
        maja.collectPollen(weissWurscht);
        System.out.println(maja.collectedPollen);
        System.out.println(wurscht.amountOfPollen);
        wurscht.getAmountOfPollen();
        willi.snooze();

    }

}
```

No consoles to display at this time.



```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly;
        wurscht = new Flower();
        Flower weissWurscht = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(wurscht);
        maja.collectPollen(weissWurscht);
        System.out.println(maja.collectedPollen);
        System.out.println(wurscht.amountOfPollen);
        wurscht.getAmountOfPollen();
        willi.snooze();

    }

}
```

Find: flySlow

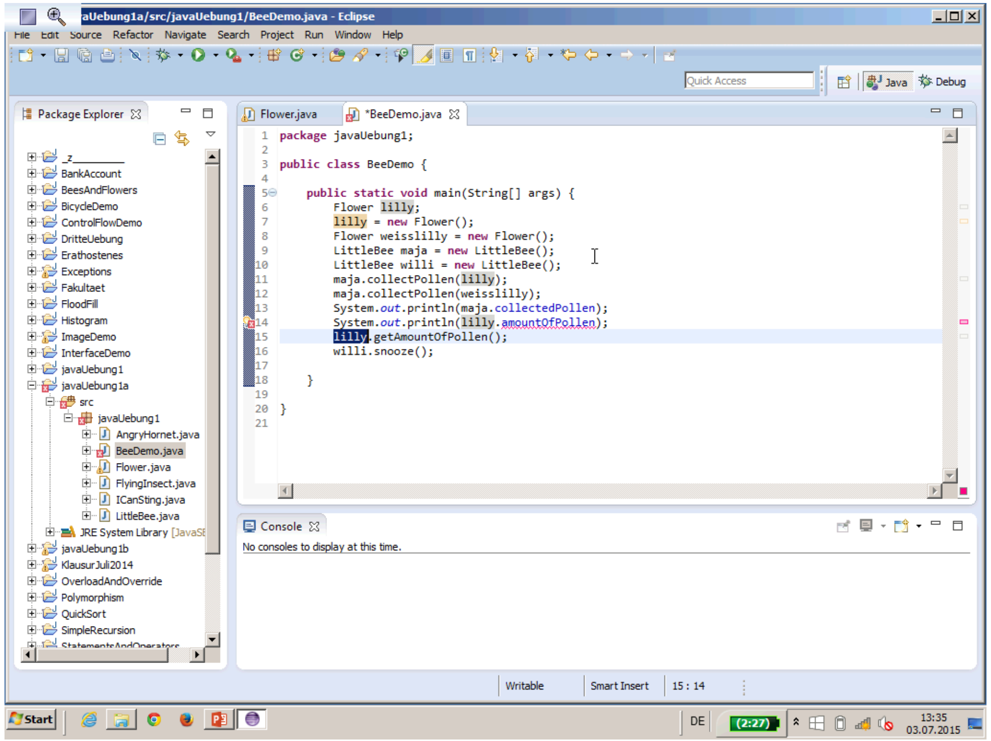
Replace with:

Direction: ● Forward ○ Backward

Scope: ● All ○ Selected lines

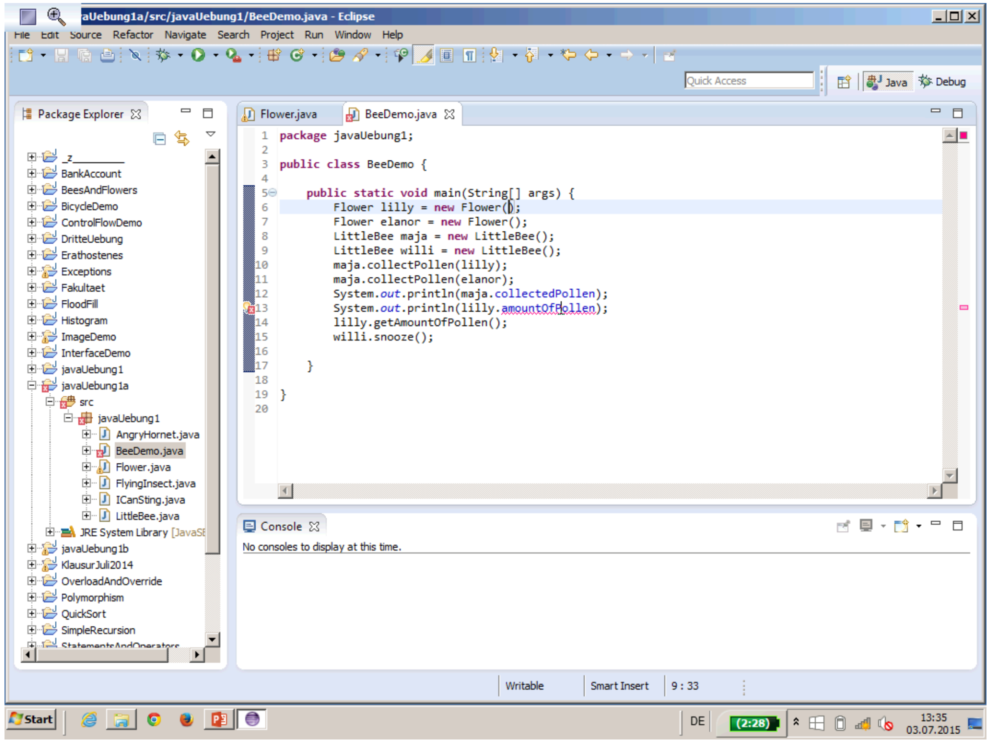
Options: □ Case sensitive ☑ Wrap search □ Whole word □ Incremental □ Regular expressions

No consoles to display at this time.



```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly;
        lilly = new Flower();
        Flower weisslilly = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(weisslilly);
        System.out.println(maja.collectedPollen);
        System.out.println(lilly.amountOfPollen);
        lilly.getAmountOfPollen();
        willi.snooze();

    }

}
```

No consoles to display at this time.

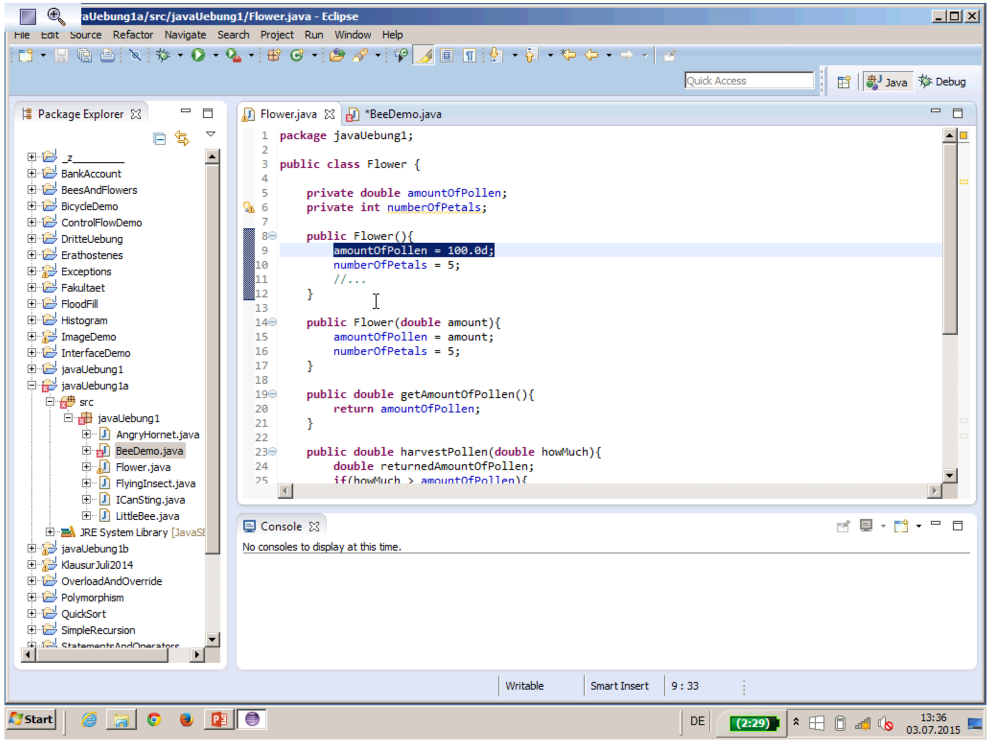


aUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower();
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.collectedPollen);
        System.out.println(lilly.amountOfPollen);
        lilly.getAmountOfPollen();
        willi.snooze();

    }

}
```

No consoles to display at this time.

Writable Smart Insert 9 : 33

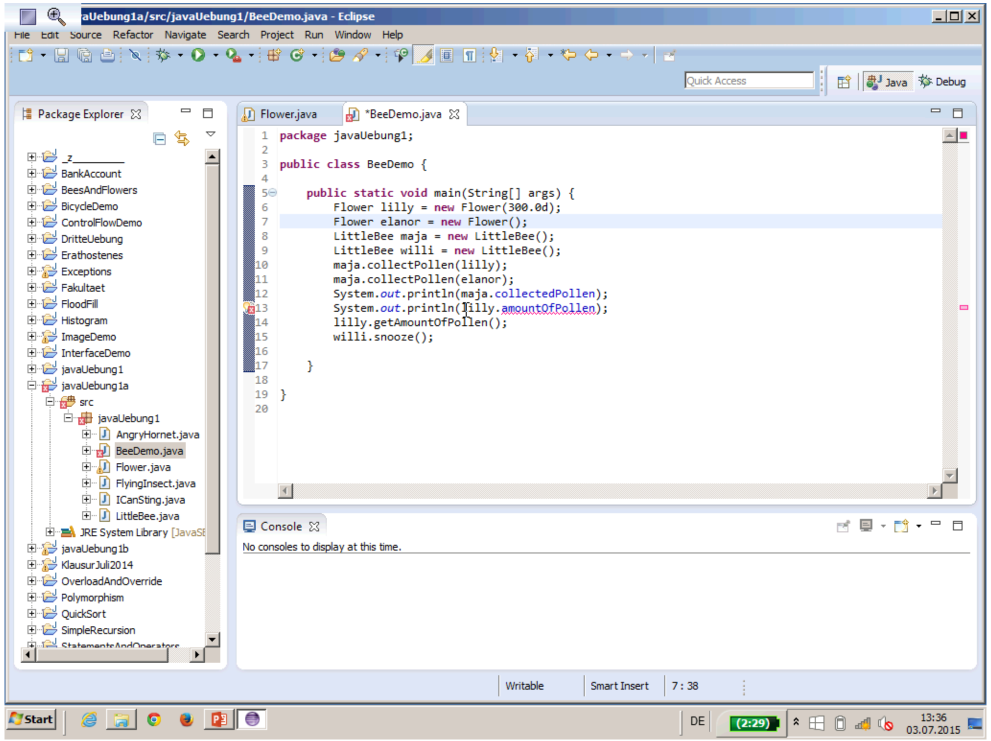


aUebung1a/src/javaUebung1/Flower.java - Eclipse

```java
package javaUebung1;

public class Flower {

    private double amountOfPollen;
    private int numberOfPetals;

    public Flower(){
        amountOfPollen = 100.0d;
        numberOfPetals = 5;
        //...
    }

    public Flower(double amount){
        amountOfPollen = amount;
        numberOfPetals = 5;
    }

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
```

No consoles to display at this time.

Writable Smart Insert 9 : 33

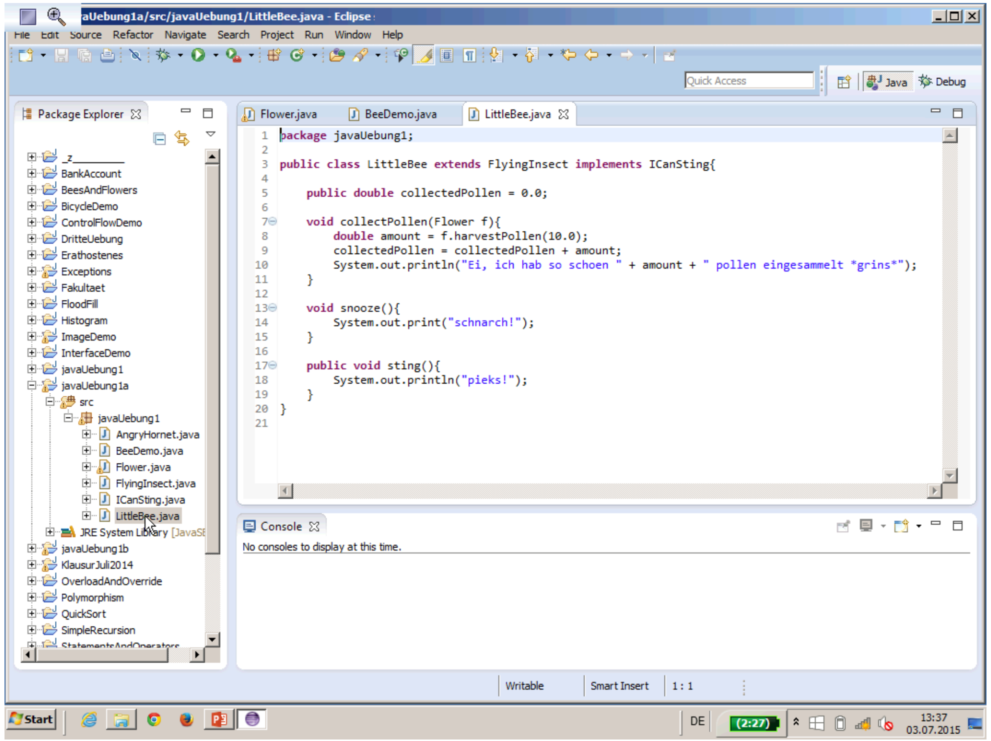


aUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.collectedPollen);
        System.out.println(lilly.amountOfPollen);
        lilly.getAmountOfPollen();
        willi.snooze();

    }

}
```

No consoles to display at this time.

Writable Smart Insert 7 : 38

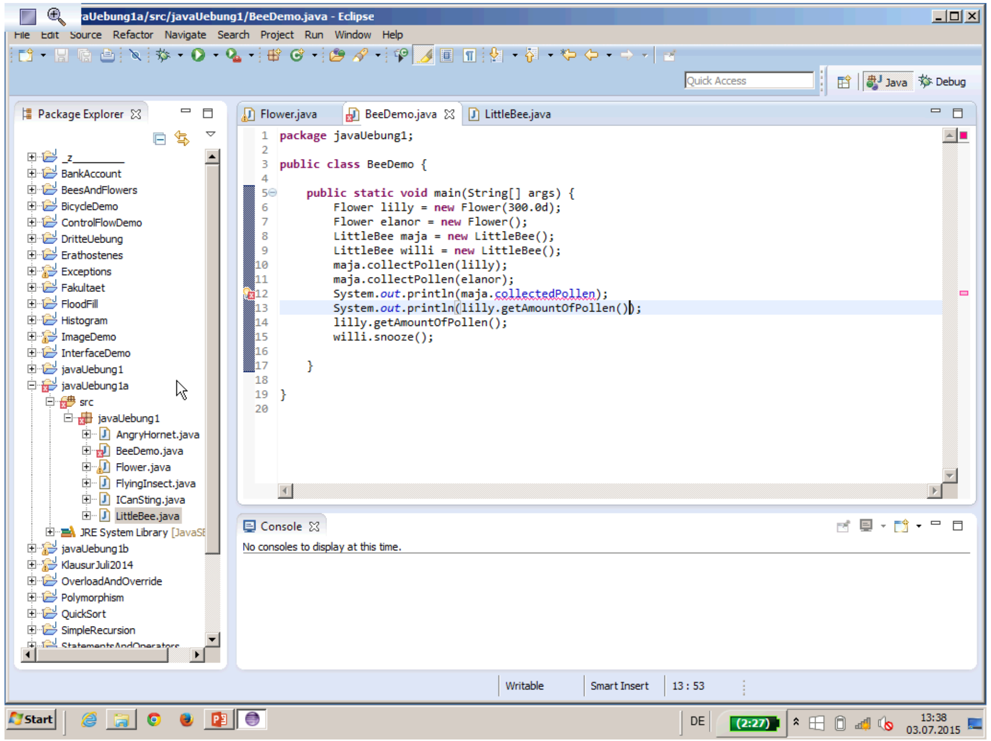


aUebung1a/src/javaUebung1/LittleBee.java - Eclipse

```java
package javaUebung1;

public class LittleBee extends FlyingInsect implements ICanSting{

    public double collectedPollen = 0.0;

    void collectPollen(Flower f){
        double amount = f.harvestPollen(10.0);
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void snooze(){
        System.out.print("schnarch!");
    }

    public void sting(){
        System.out.println("pieks!");
    }
}
```

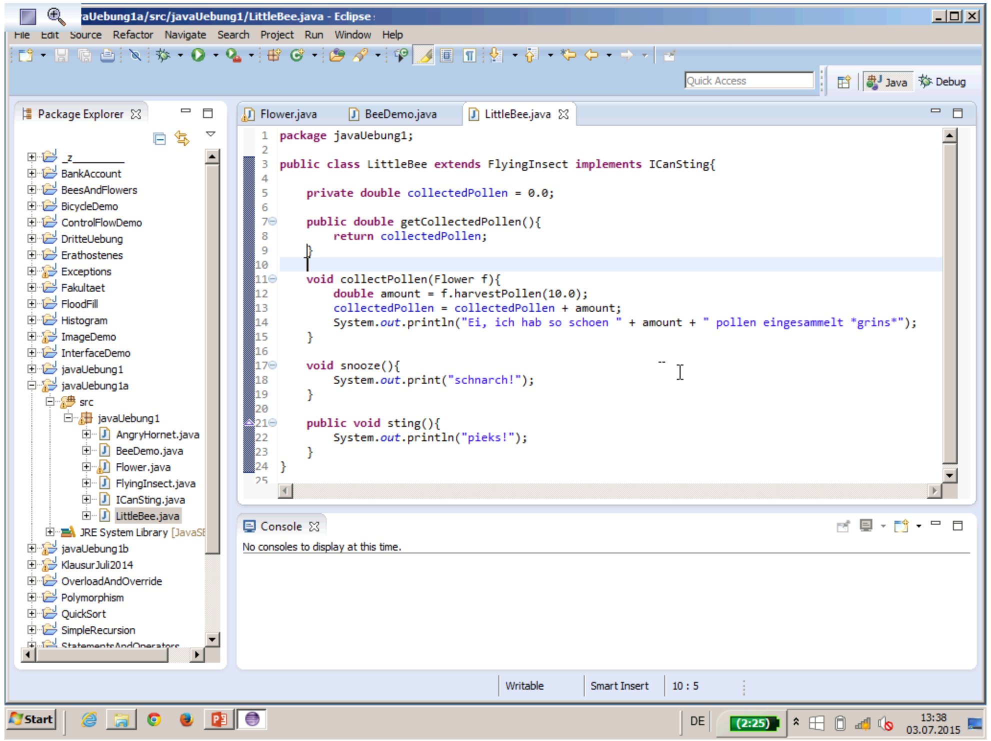
No consoles to display at this time.

Writable Smart Insert 1 : 1



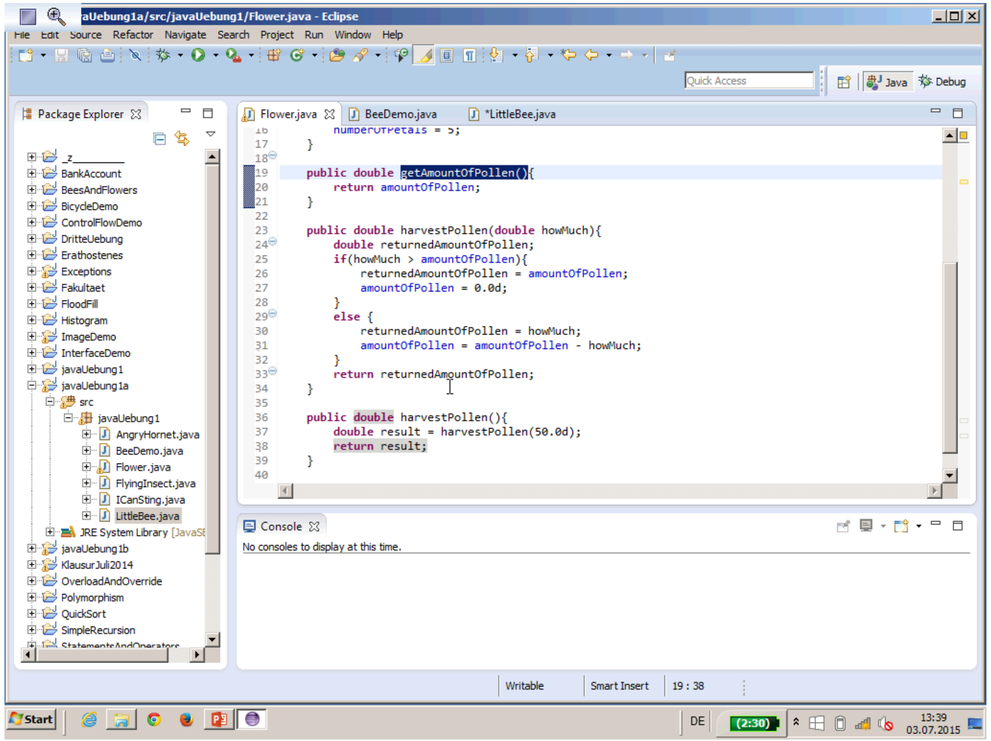
```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.collectedPollen);
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();

    }

}
```

No consoles to display at this time.



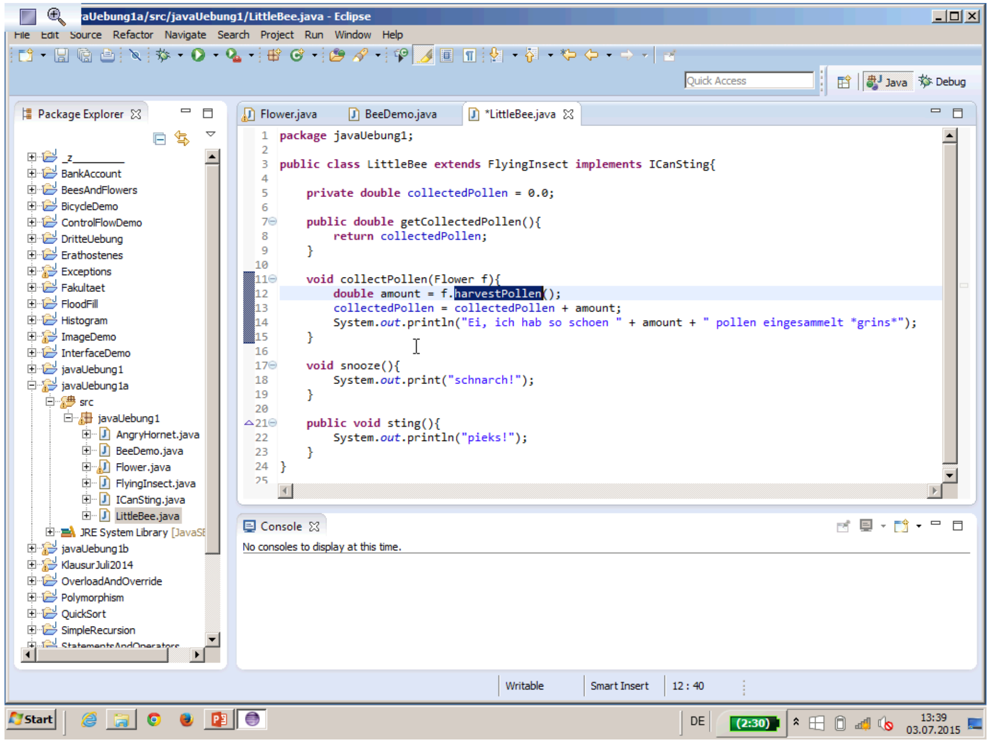
```java
package javaUebung1;

public class LittleBee extends FlyingInsect implements ICanSting{

    private double collectedPollen = 0.0;

    public double getCollectedPollen(){
        return collectedPollen;
    }

    void collectPollen(Flower f){
        double amount = f.harvestPollen(10.0);
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void snooze(){
        System.out.print("schnarch!");
    }

    public void sting(){
        System.out.println("pieks!");
    }
}
```

No consoles to display at this time.



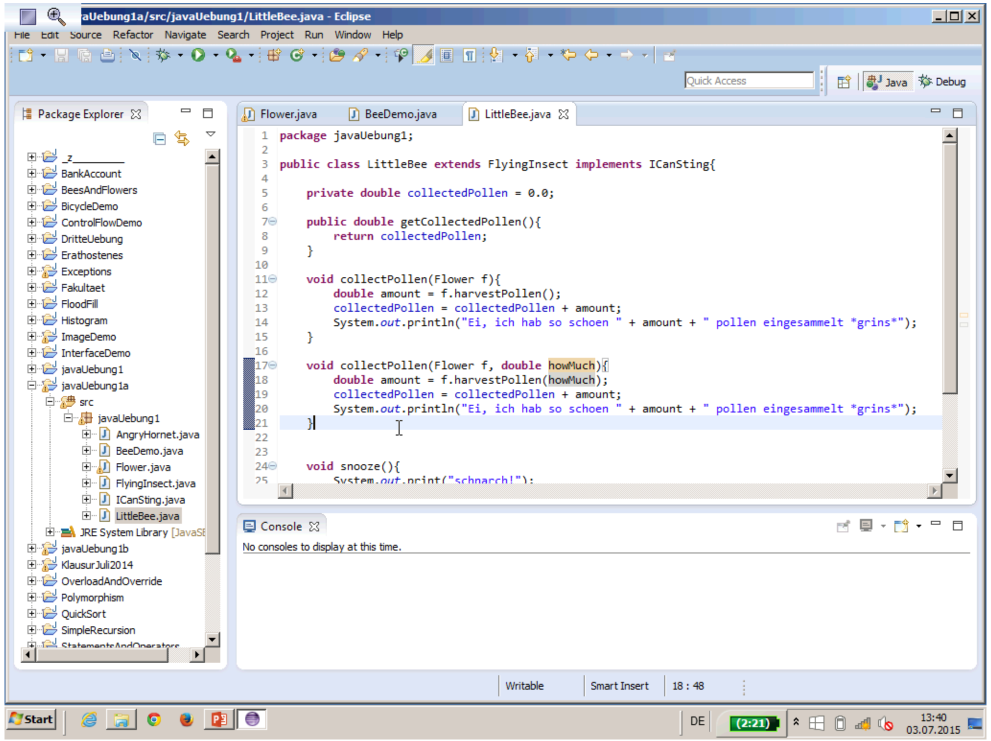
```java
        numberOfPetals = 5;
    }

    public double getAmountOfPollen(){
        return amountOfPollen;
    }

    public double harvestPollen(double howMuch){
        double returnedAmountOfPollen;
        if(howMuch > amountOfPollen){
            returnedAmountOfPollen = amountOfPollen;
            amountOfPollen = 0.0d;
        }
        else {
            returnedAmountOfPollen = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
        }
        return returnedAmountOfPollen;
    }

    public double harvestPollen(){
        double result = harvestPollen(50.0d);
        return result;
    }

```

No consoles to display at this time.



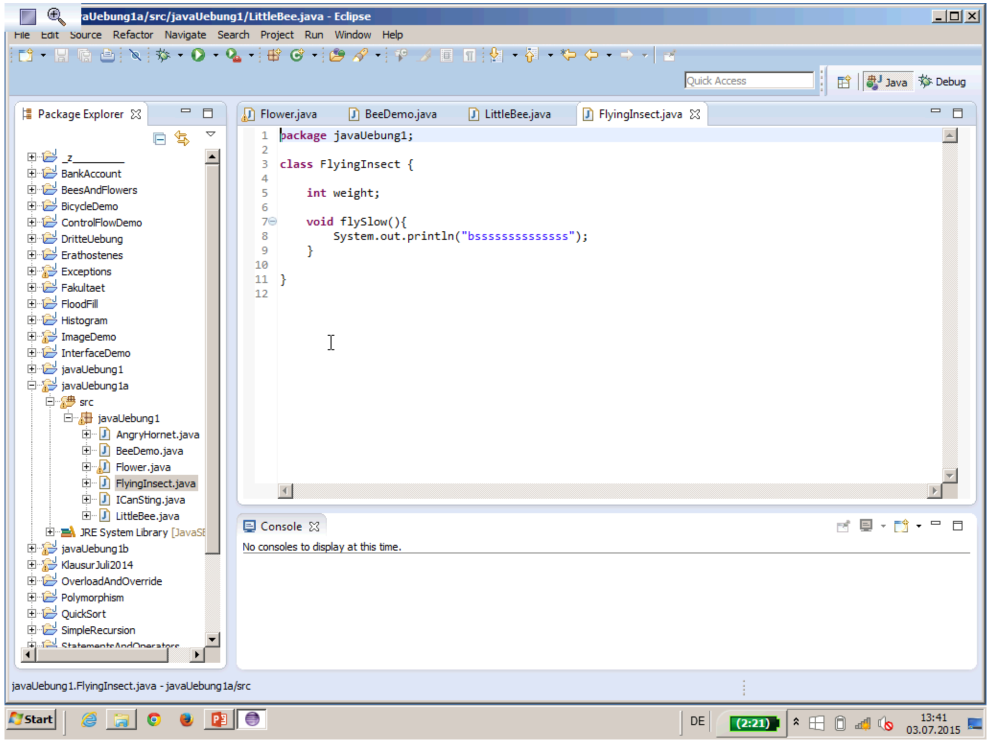
```java
package javaUebung1;

public class LittleBee extends FlyingInsect implements ICanSting{

    private double collectedPollen = 0.0;

    public double getCollectedPollen(){
        return collectedPollen;
    }

    void collectPollen(Flower f){
        double amount = f.harvestPollen();
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void snooze(){
        System.out.print("schnarch!");
    }

    public void sting(){
        System.out.println("pieks!");
    }
}
```

No consoles to display at this time.



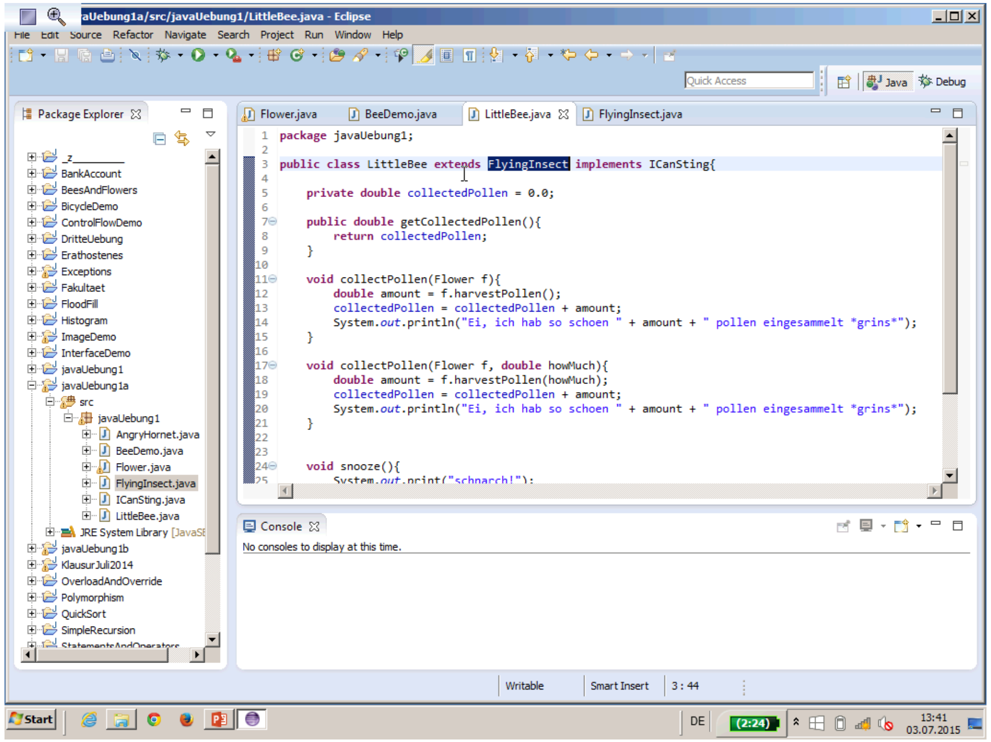
```java
package javaUebung1;

public class LittleBee extends FlyingInsect implements ICanSting{

    private double collectedPollen = 0.0;

    public double getCollectedPollen(){
        return collectedPollen;
    }

    void collectPollen(Flower f){
        double amount = f.harvestPollen();
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void collectPollen(Flower f, double howMuch){
        double amount = f.harvestPollen(howMuch);
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }


    void snooze(){
        System.out.print("schnarch!");
```

```
No consoles to display at this time.
```



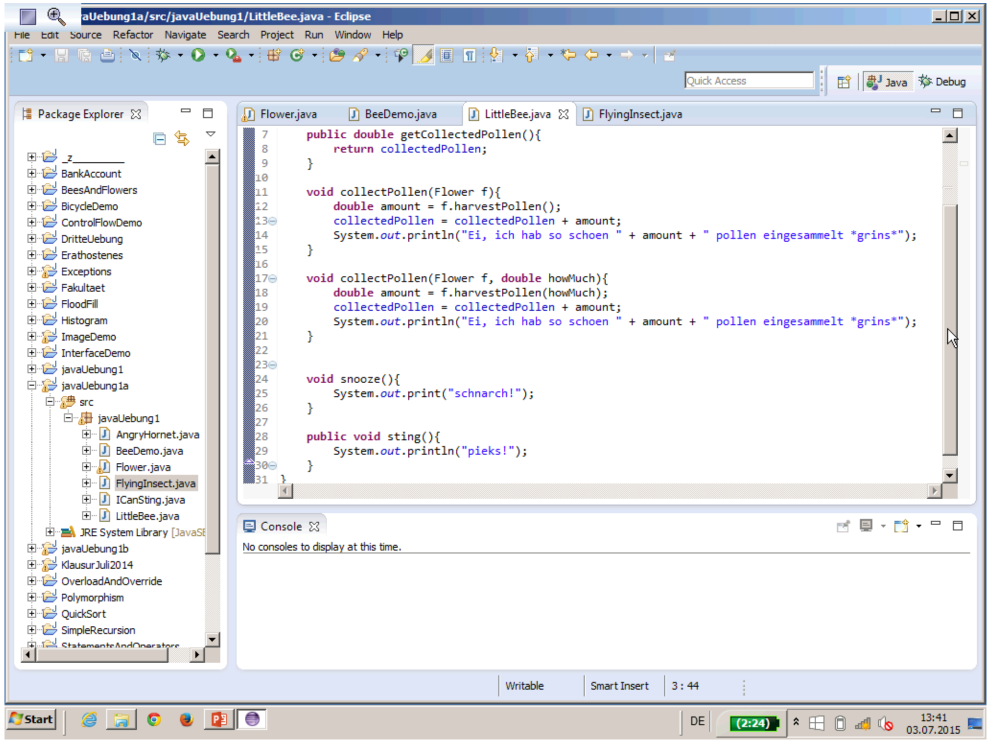
```java
package javaUebung1;

class FlyingInsect {

    int weight;

    void flySlow(){
        System.out.println("bsssssssssssss");
    }

}
```

```
No consoles to display at this time.
```



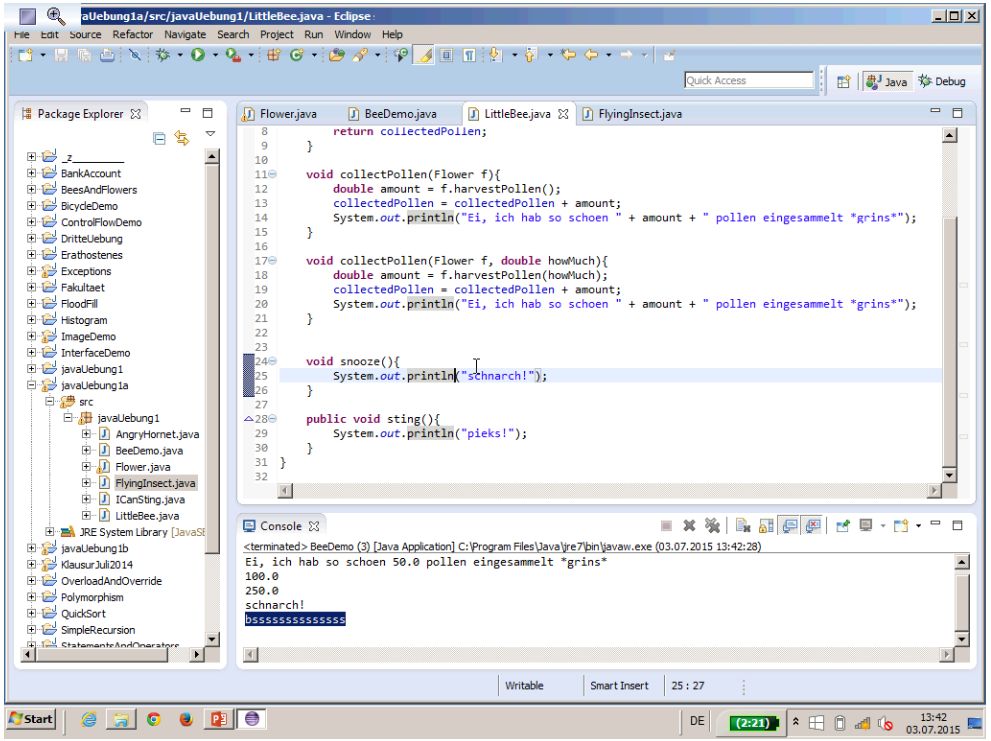
```java
package javaUebung1;

public class LittleBee extends FlyingInsect implements ICanSting{

    private double collectedPollen = 0.0;

    public double getCollectedPollen(){
        return collectedPollen;
    }

    void collectPollen(Flower f){
        double amount = f.harvestPollen();
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void collectPollen(Flower f, double howMuch){
        double amount = f.harvestPollen(howMuch);
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }


    void snooze(){
        System.out.print("schnarch!");
```

```
No consoles to display at this time.
```



```java
    public double getCollectedPollen(){
        return collectedPollen;
    }

    void collectPollen(Flower f){
        double amount = f.harvestPollen();
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void collectPollen(Flower f, double howMuch){
        double amount = f.harvestPollen(howMuch);
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }


    void snooze(){
        System.out.print("schnarch!");
    }

    public void sting(){
        System.out.println("pieks!");
    }
}
```

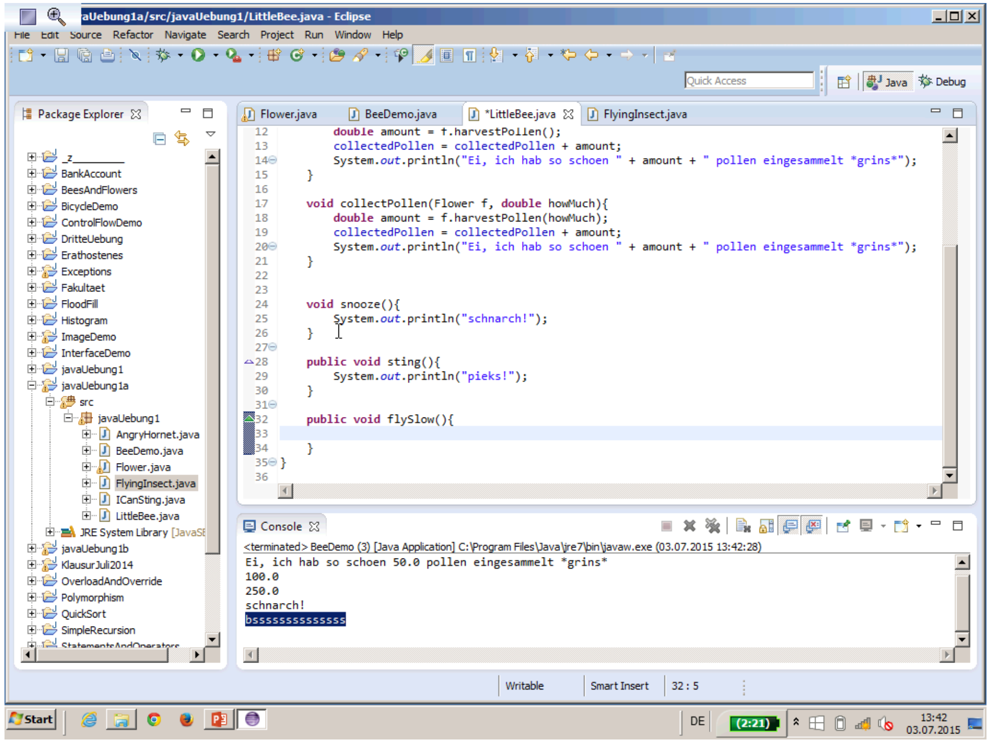
```
No consoles to display at this time.
```



aUebung1a/src/javaUebung1/LittleBee.java - Eclipse

```java
        return collectedPollen;
    }

    void collectPollen(Flower f){
        double amount = f.harvestPollen();
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void collectPollen(Flower f, double howMuch){
        double amount = f.harvestPollen(howMuch);
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }


    void snooze(){
        System.out.println("schnarch!");
    }

    public void sting(){
        System.out.println("pieks!");
    }
}
```

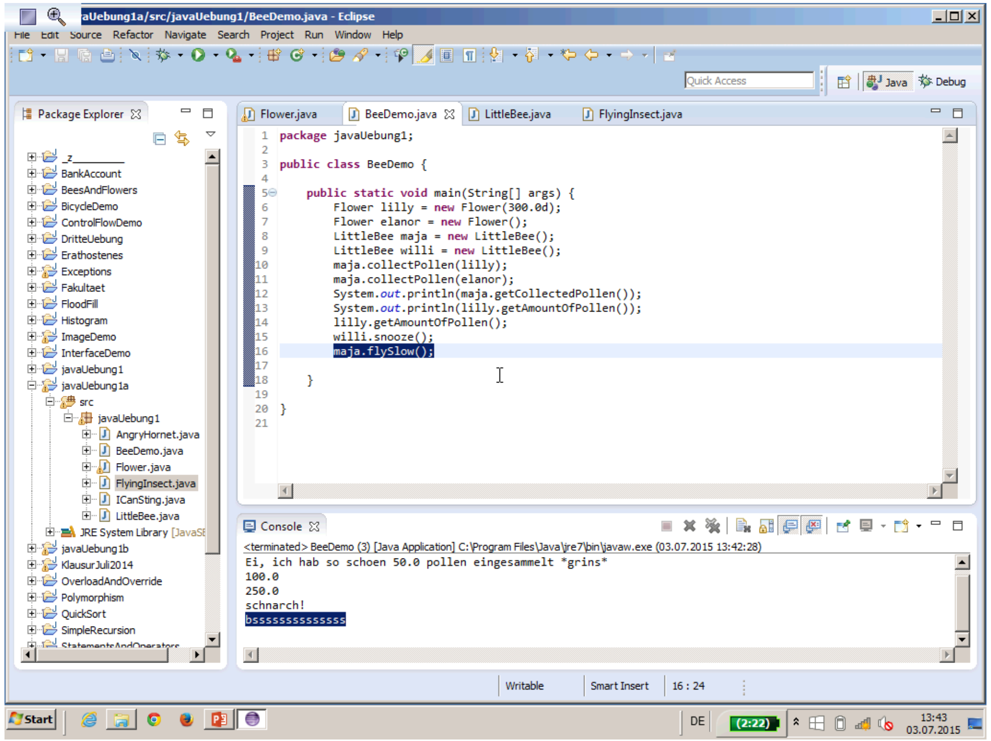
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:42:28)
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
bsssssssssssss
```



aUebung1a/src/javaUebung1/LittleBee.java - Eclipse

```java
        double amount = f.harvestPollen();
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }

    void collectPollen(Flower f, double howMuch){
        double amount = f.harvestPollen(howMuch);
        collectedPollen = collectedPollen + amount;
        System.out.println("Ei, ich hab so schoen " + amount + " pollen eingesammelt *grins*");
    }


    void snooze(){
        System.out.println("schnarch!");
    }

    public void sting(){
        System.out.println("pieks!");
    }

    public void flySlow(){

    }
}
```

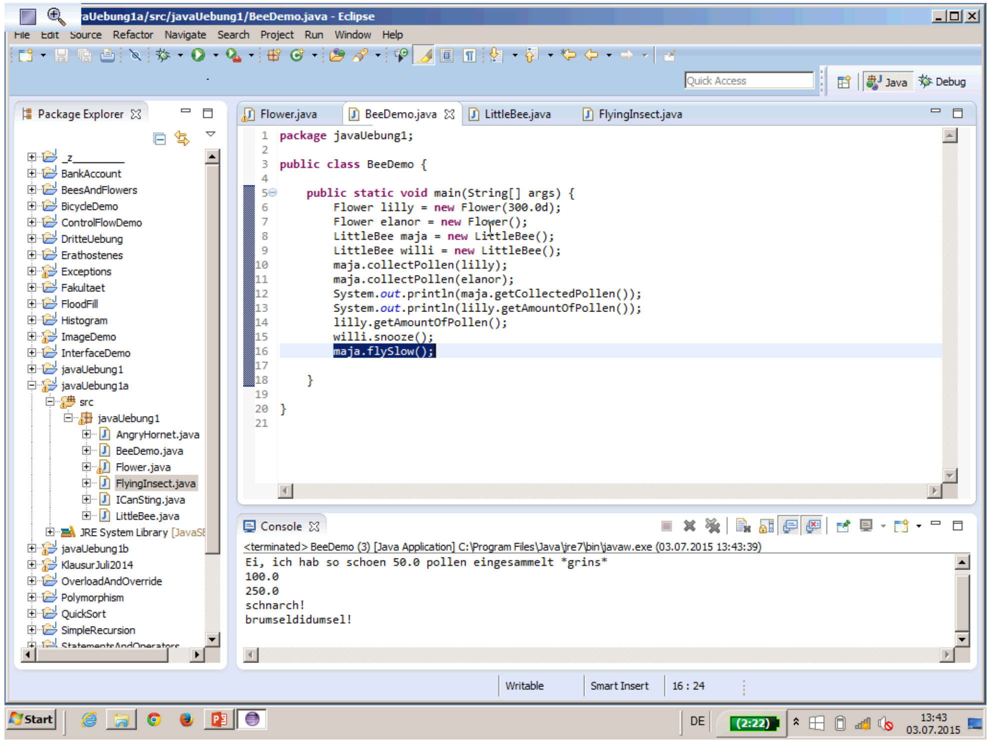
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:42:28)
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
bsssssssssssss
```



aUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();

    }

}
```

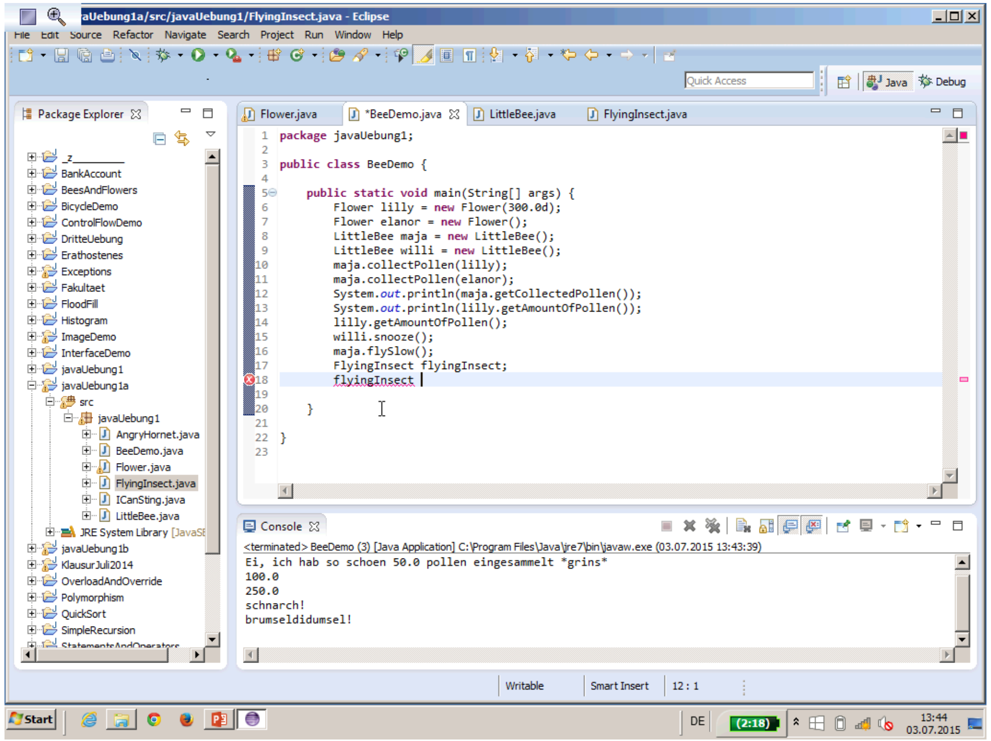
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:42:28)
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
bsssssssssssss
```



aUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();

    }

}
```

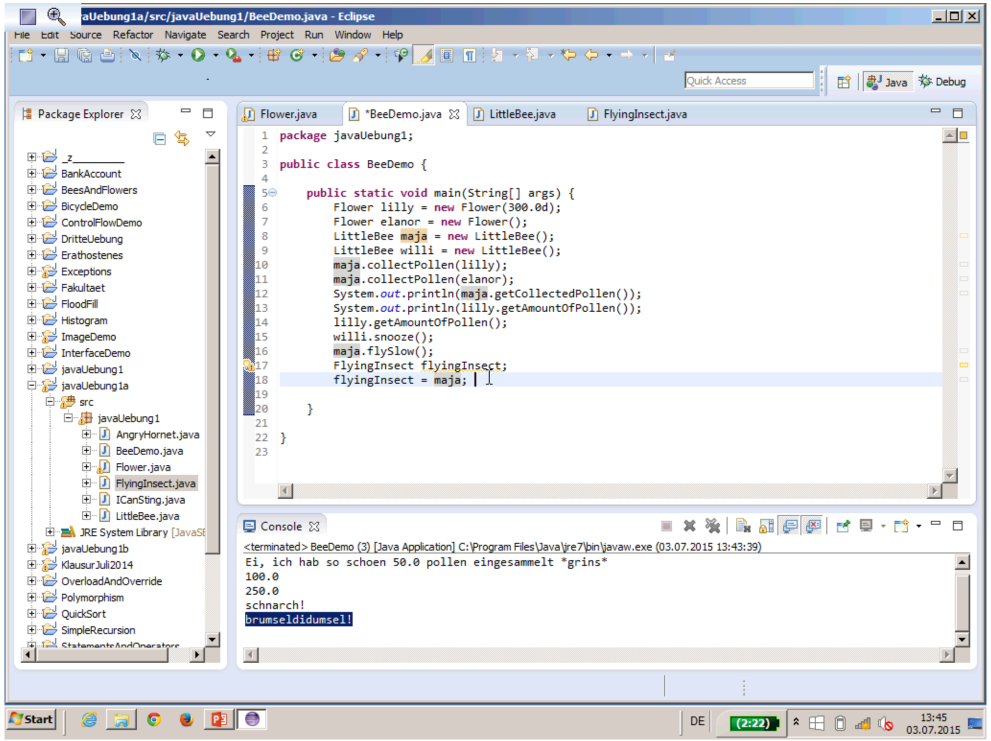
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:43:39)
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
brumseldidumsel!
```



aUebung1a/src/javaUebung1/FlyingInsect.java - Eclipse

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect 

    }

}
```

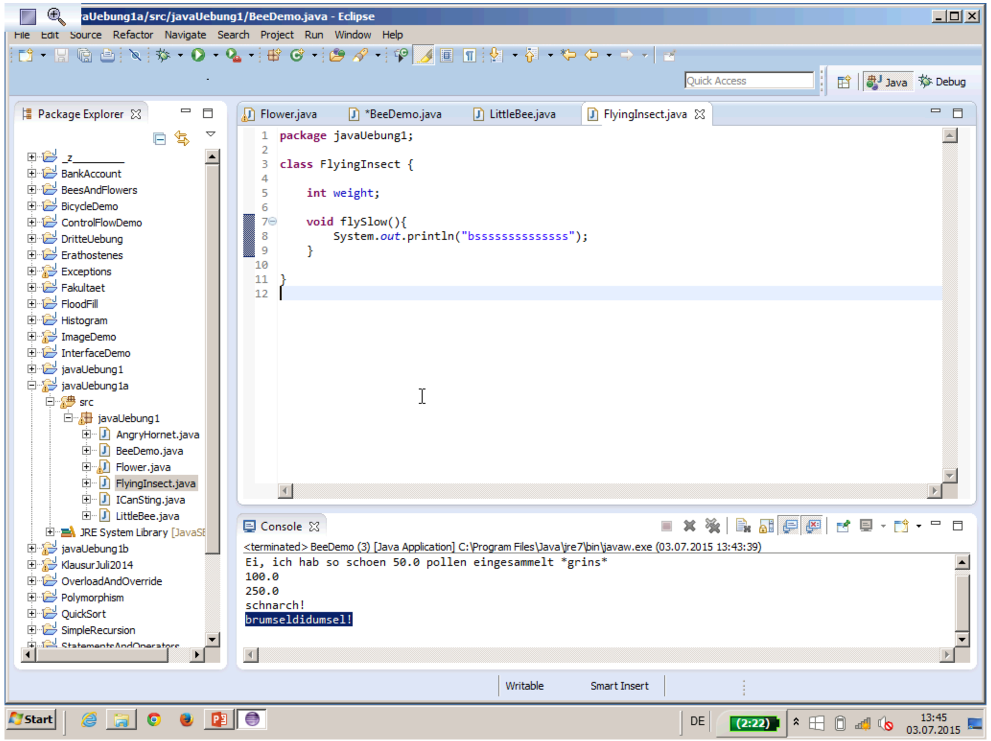
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:43:39)
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
brumseldidumsel!
```



aUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja; 

    }

}
```

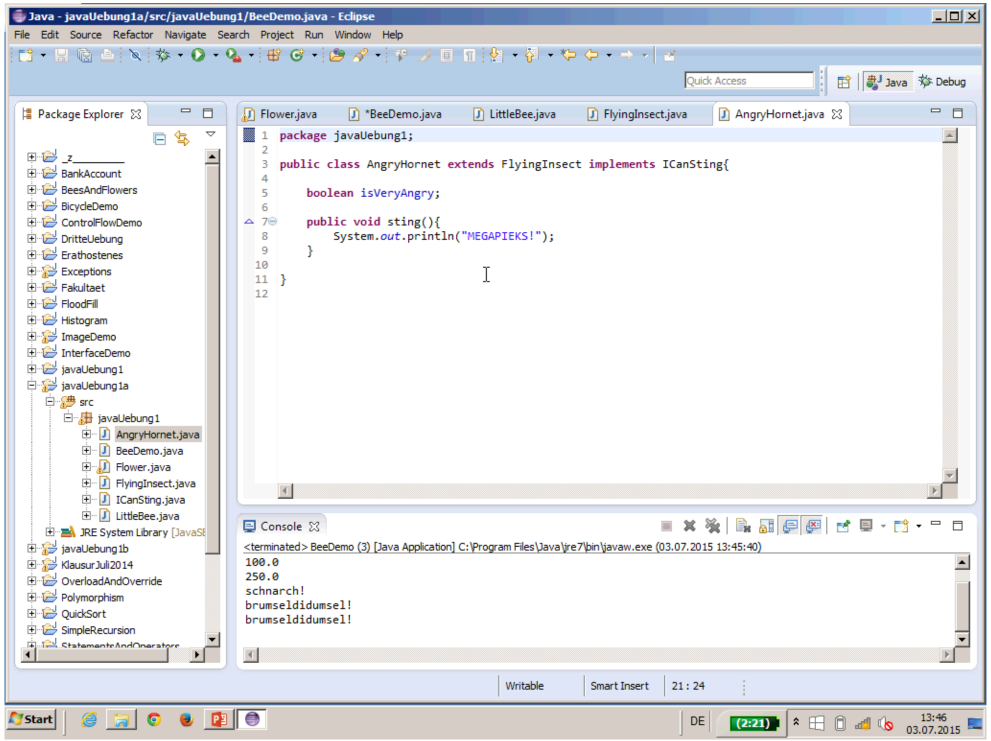
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:43:39)
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
brumseldidumsel!
```



aUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

```java
package javaUebung1;

class FlyingInsect {

    int weight;

    void flySlow(){
        System.out.println("bssssssssssssss");
    }

}

```

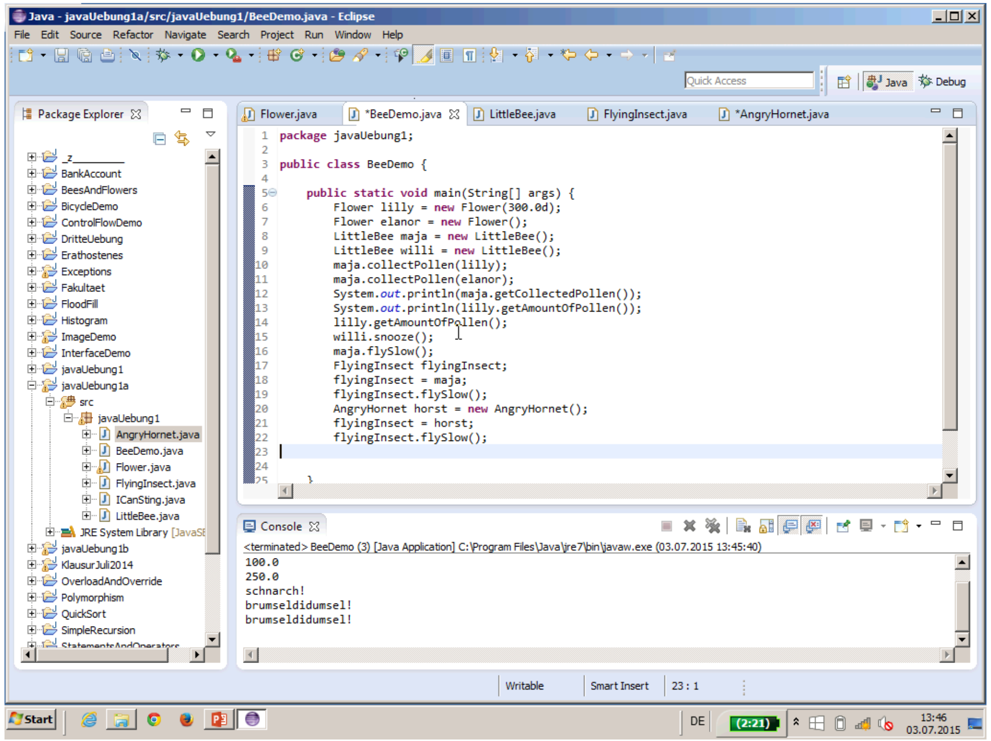
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:43:39)
Ei, ich hab so schoen 50.0 pollen eingesammelt *grins*
100.0
250.0
schnarch!
brumseldidumsel!
```



Java - javaUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

```java
package javaUebung1;

public class AngryHornet extends FlyingInsect implements ICanSting{

    boolean isVeryAngry;

    public void sting(){
        System.out.println("MEGAPIEKS!");
    }

}

```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:45:40)
100.0
250.0
schnarch!
brumseldidumsel!
brumseldidumsel!
```



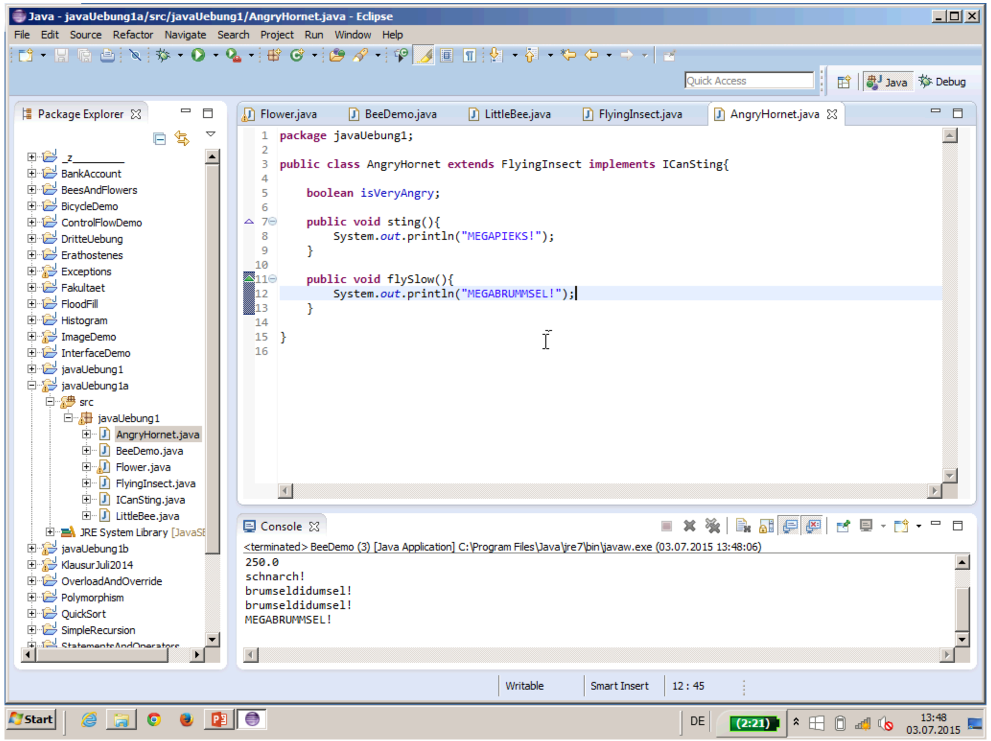
Java - javaUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

*BeeDemo.java

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        flyingInsect = horst;
        flyingInsect.flySlow();


    }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:45:40)
100.0
250.0
schnarch!
brumseldidumsel!
brumseldidumsel!
```



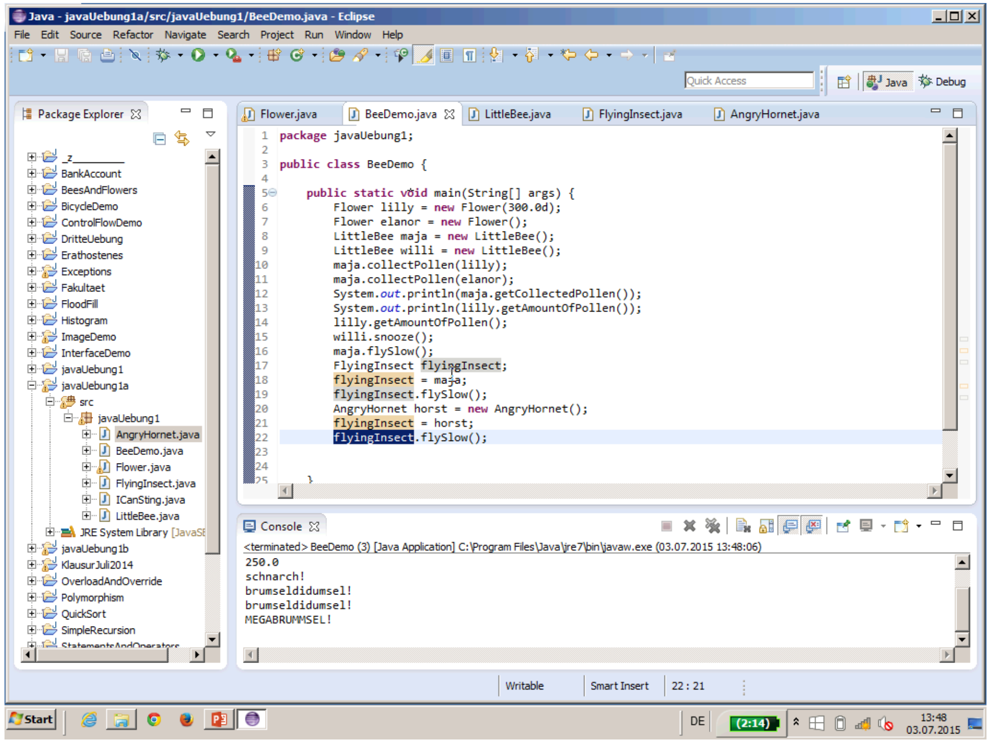
Java - javaUebung1a/src/javaUebung1/AngryHornet.java - Eclipse

AngryHornet.java

```java
package javaUebung1;

public class AngryHornet extends FlyingInsect implements ICanSting{

    boolean isVeryAngry;

    public void sting(){
        System.out.println("MEGAPIEKS!");
    }

    public void flySlow(){
        System.out.println("MEGABRUMMSEL!");
    }

}
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:48:06)
250.0
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGABRUMMSEL!
```



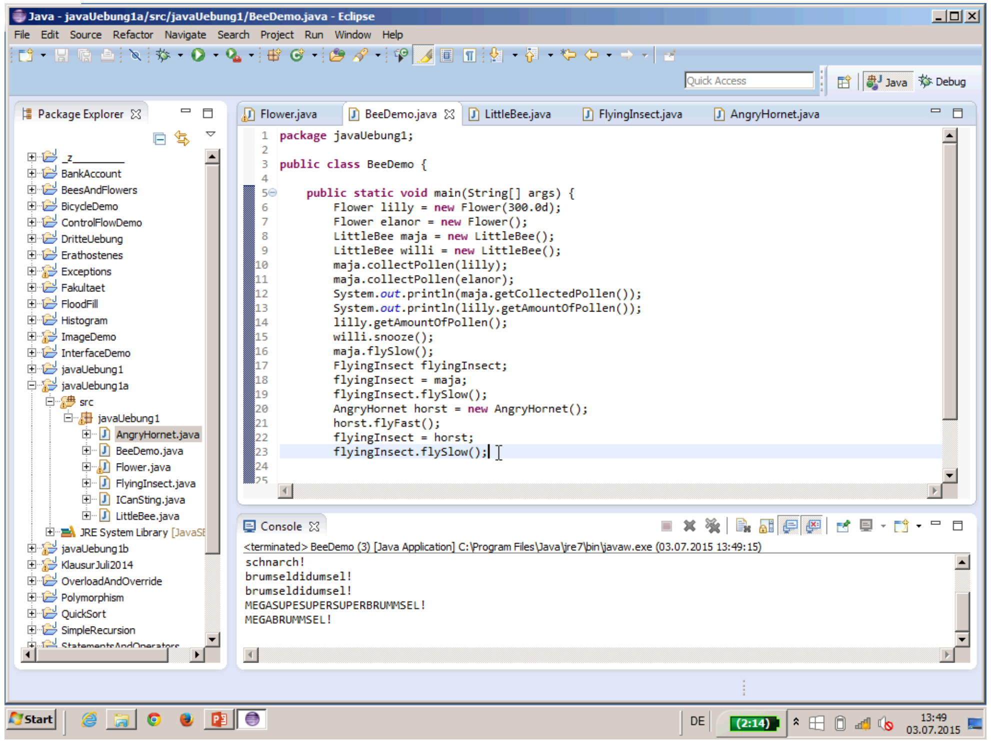
Java - javaUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

BeeDemo.java

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        flyingInsect = horst;
        flyingInsect.flySlow();


    }
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:48:06)
250.0
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGABRUMMSEL!
```



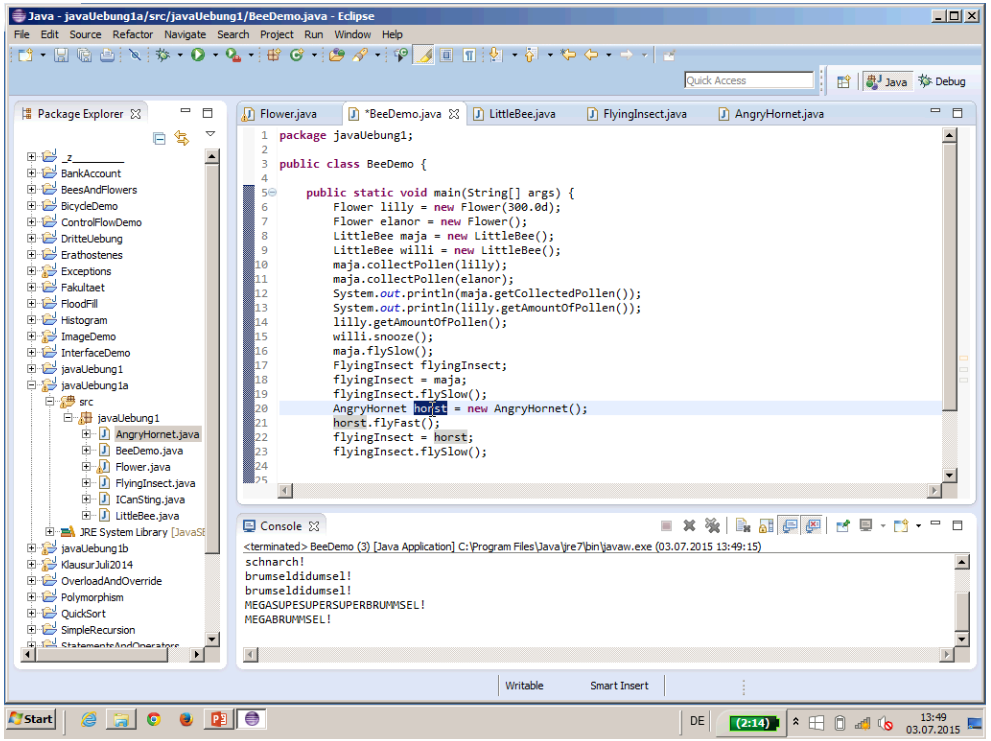
Java - javaUebung1a/src/javaUebung1/BeeDemo.java - Eclipse

BeeDemo.java

```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
```

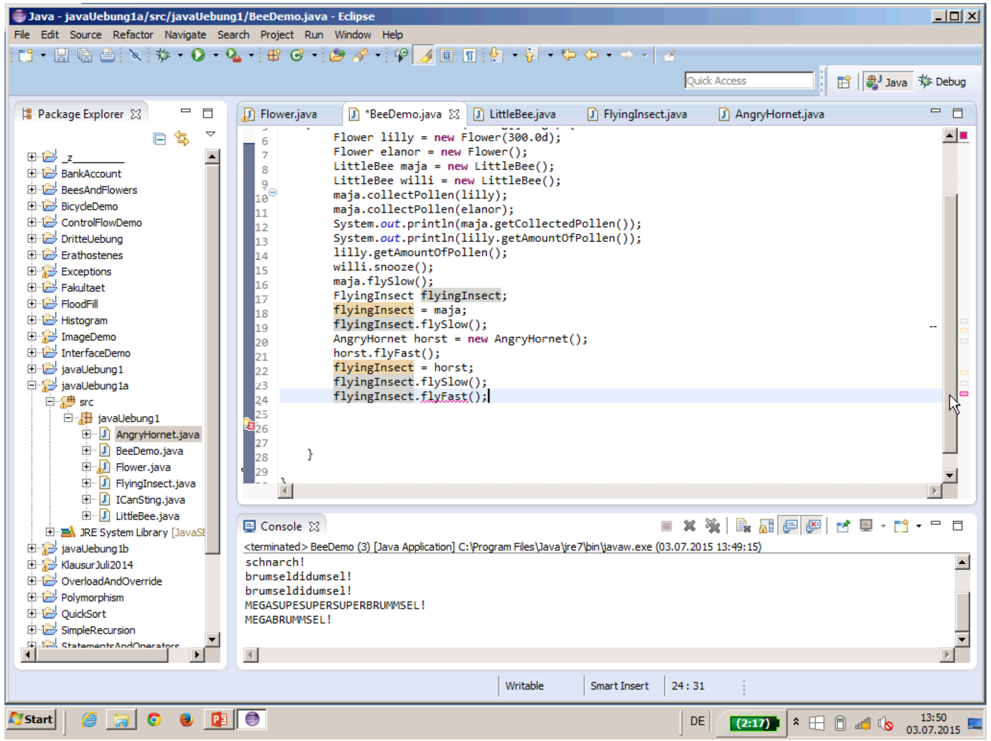
Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:49:15)
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
```



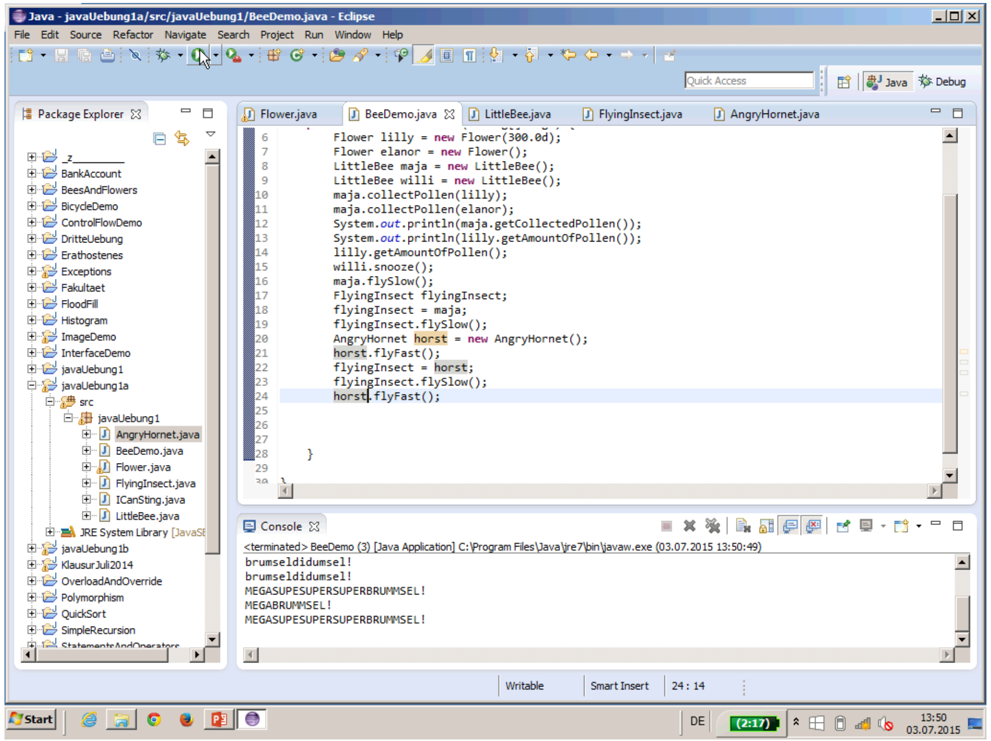
```java
package javaUebung1;

public class BeeDemo {

    public static void main(String[] args) {
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:49:15)
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
```



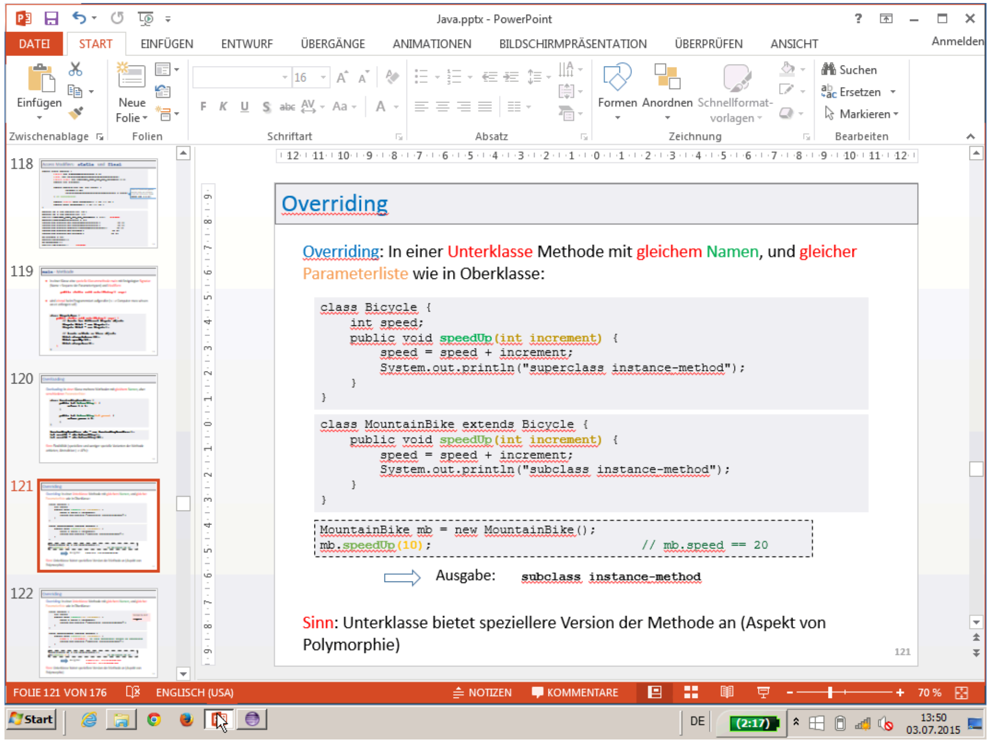
```java
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        flyingInsect.flyFast();



    }
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:49:15)
schnarch!
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
```



```java
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();



    }
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
```



Overriding

Overriding: In einer Unterklasse Methode mit gleichem Namen, und gleicher Parameterliste wie in Oberklase:

```java
class Bicycle {
    int speed;
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("superclass instance-method");
    }
}
```

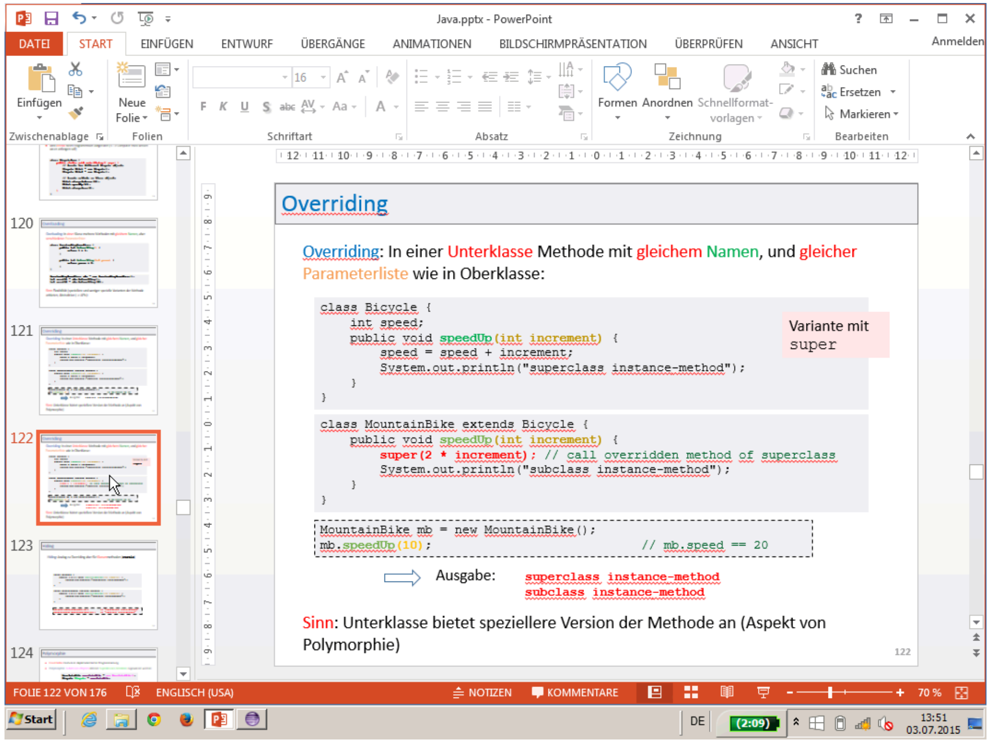
```java
class MountainBike extends Bicycle {
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("subclass instance-method");
    }
}
```

```java
MountainBike mb = new MountainBike();
mb.speedUp(10);                          // mb.speed == 20
```

⇨ Ausgabe: `subclass instance-method`

Sinn: Unterklasse bietet speziellere Version der Methode an (Aspekt von Polymorphie)

121


Java.pptx - PowerPoint
DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHIRMPRÄSENTATION ÜBERPRÜFEN ANSICHT
FOLIE 122 VON 176
ENGLISCH (USA)
70 %
13:51
03.07.2015

Overriding

Overriding: In einer Unterklasse Methode mit gleichem Namen, und gleicher Parameterliste wie in Oberklasse:

```
class Bicycle {
    int speed;
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("superclass instance-method");
    }
}
```

Variante mit `super`

```
class MountainBike extends Bicycle {
    public void speedUp(int increment) {
        super(2 * increment); // call overridden method of superclass
        System.out.println("subclass instance-method");
    }
}
```

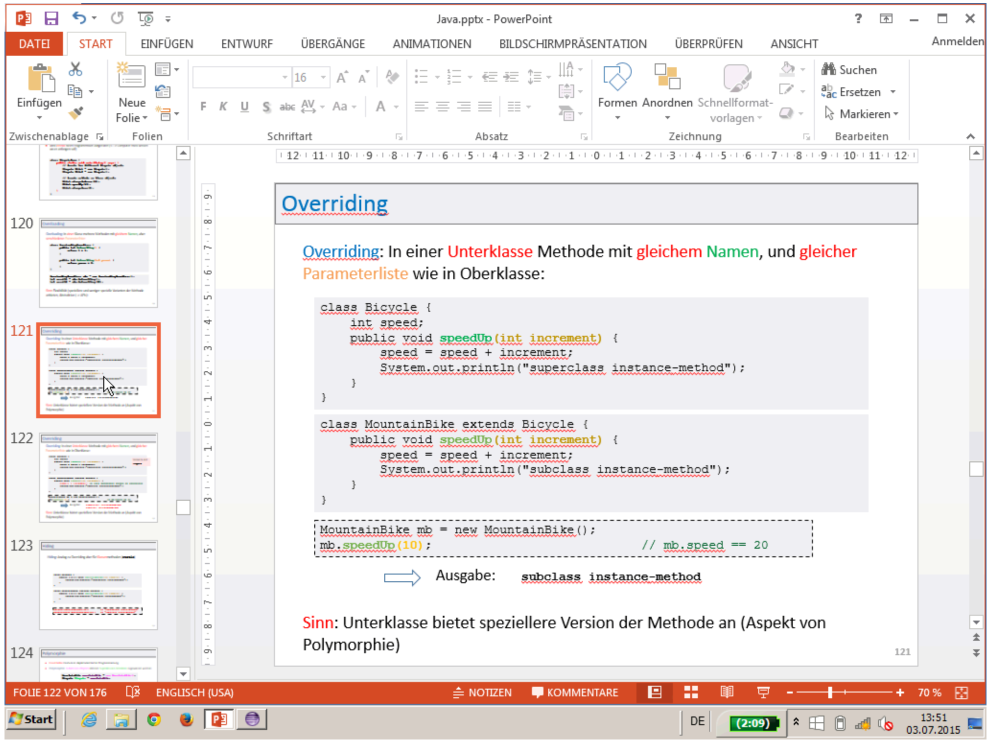
```
MountainBike mb = new MountainBike();
mb.speedUp(10);                         // mb.speed == 20
```

Ausgabe:

```
superclass instance-method
subclass instance-method
```

Sinn: Unterklasse bietet speziellere Version der Methode an (Aspekt von Polymorphie)

122



Overriding

Overriding: In einer Unterklasse Methode mit gleichem Namen, und gleicher Parameterliste wie in Oberklasse:

```
class Bicycle {
    int speed;
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("superclass instance-method");
    }
}
```

```
class MountainBike extends Bicycle {
    public void speedUp(int increment) {
        speed = speed + increment;
        System.out.println("subclass instance-method");
    }
}
```

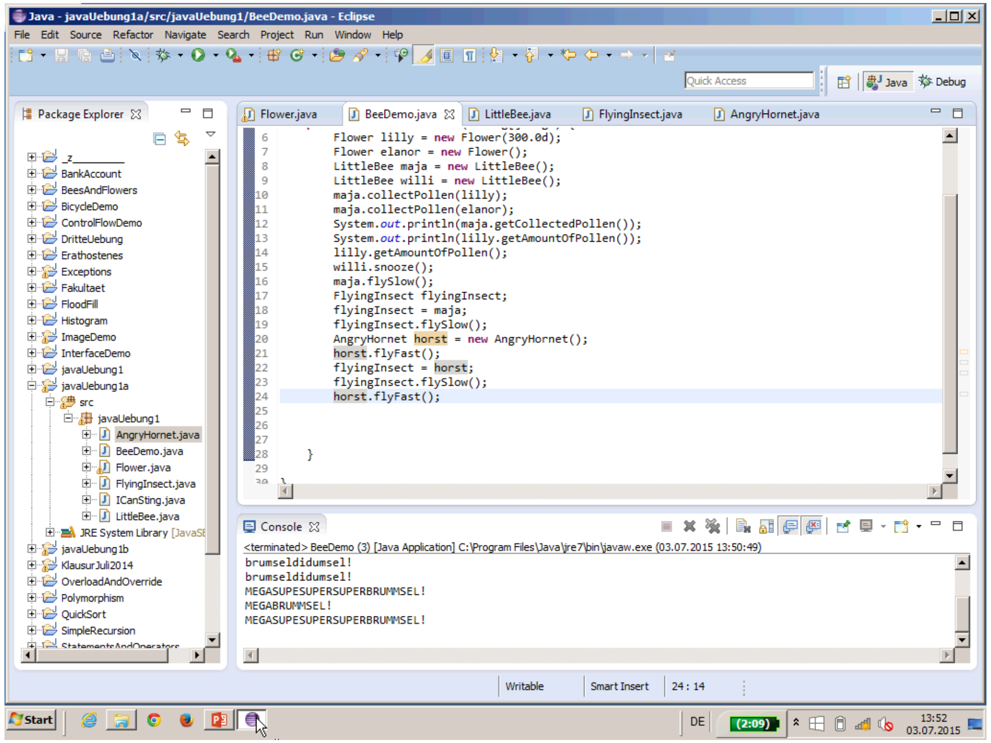
```
MountainBike mb = new MountainBike();
mb.speedUp(10);                         // mb.speed == 20
```

Ausgabe:

```
subclass instance-method
```

Sinn: Unterklasse bietet speziellere Version der Methode an (Aspekt von Polymorphie)

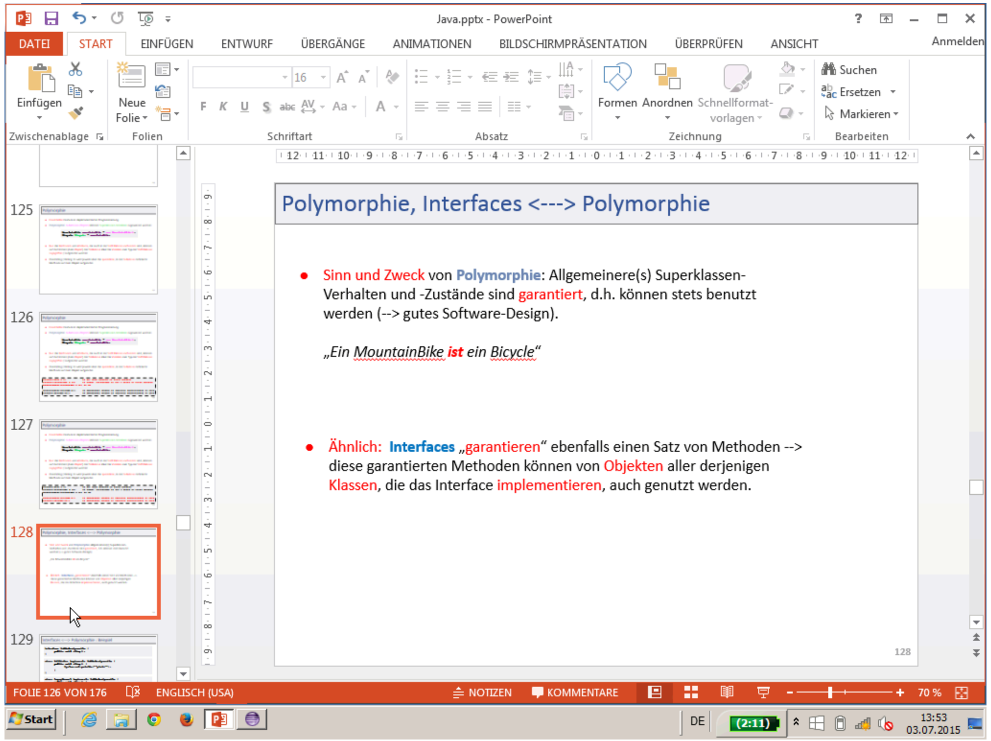
121



```
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();



    }

}
```

Console

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
```



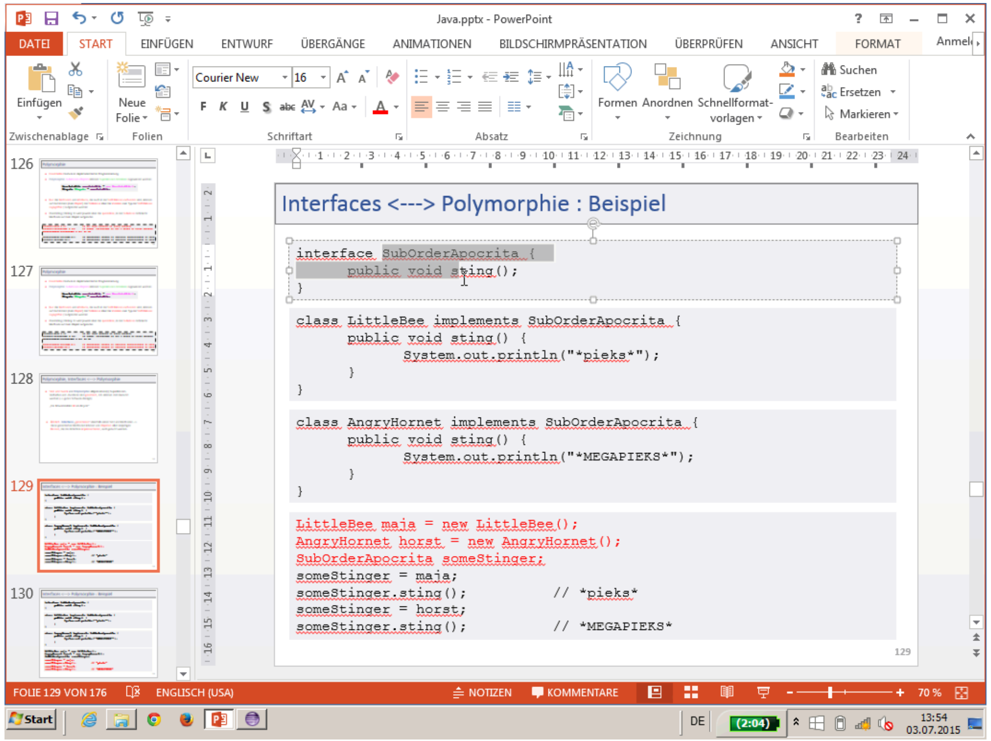
Polymorphie, Interfaces <---> Polymorphie

- Sinn und Zweck von Polymorphie: Allgemeinere(s) Superklassen-Verhalten und -Zustände sind garantiert, d.h. können stets benutzt werden (--> gutes Software-Design).

  „Ein MountainBike ***ist*** *ein Bicycle“*

- Ähnlich: **Interfaces** „garantieren“ ebenfalls einen Satz von Methoden --> diese garantierten Methoden können von Objekten aller derjenigen Klassen, die das Interface implementieren, auch genutzt werden.

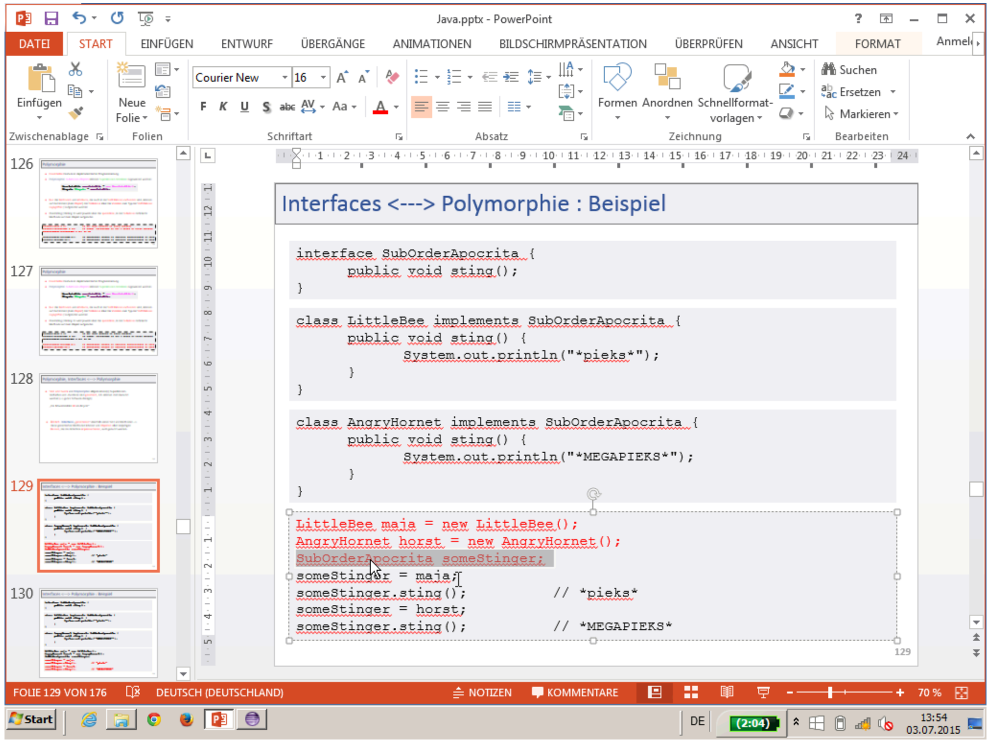
128


Java.pptx - PowerPoint
DATEI START EINFÜGEN ENTWURF ÜBERGÄNGE ANIMATIONEN BILDSCHIRMPRÄSENTATION ÜBERPRÜFEN ANSICHT FORMAT
Courier New 16
FOLIE 129 VON 176 ENGLISCH (USA) NOTIZEN KOMMENTARE 70 %
13:54 03.07.2015

Interfaces <---> Polymorphie : Beispiel

```
interface SubOrderApocrita {
        public void sting();
}

class LittleBee implements SubOrderApocrita {
        public void sting() {
                System.out.println("*pieks*");
        }
}

class AngryHornet implements SubOrderApocrita {
        public void sting() {
                System.out.println("*MEGAPIEKS*");
        }
}

LittleBee maja = new LittleBee();
AngryHornet horst = new AngryHornet();
SubOrderApocrita someStinger;
someStinger = maja;
someStinger.sting();            // *pieks*
someStinger = horst;
someStinger.sting();            // *MEGAPIEKS*
```

129


Java.pptx - PowerPoint
FOLIE 129 VON 176 DEUTSCH (DEUTSCHLAND) NOTIZEN KOMMENTARE 70 %
13:54 03.07.2015

Interfaces <---> Polymorphie : Beispiel

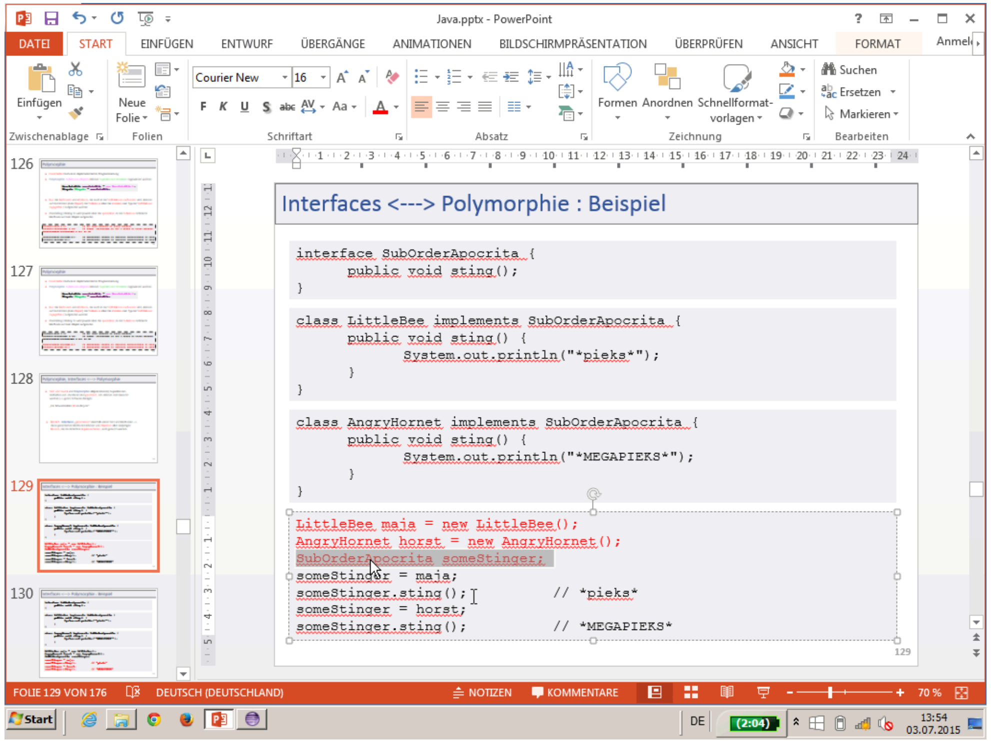
interface SubOrderApocrita {
        public void sting();
}

class LittleBee implements SubOrderApocrita {
        public void sting() {
                System.out.println("*pieks*");
        }
}

class AngryHornet implements SubOrderApocrita {
        public void sting() {
                System.out.println("*MEGAPIEKS*");
        }
}

LittleBee maja = new LittleBee();
AngryHornet horst = new AngryHornet();
SubOrderApocrita someStinger;
someStinger = maja;
someStinger.sting();   // *pieks*
someStinger = horst;
someStinger.sting();   // *MEGAPIEKS*

129


Java.pptx - PowerPoint
FOLIE 129 VON 176 DEUTSCH (DEUTSCHLAND) NOTIZEN KOMMENTARE 70 %
13:54 03.07.2015

Interfaces <---> Polymorphie : Beispiel

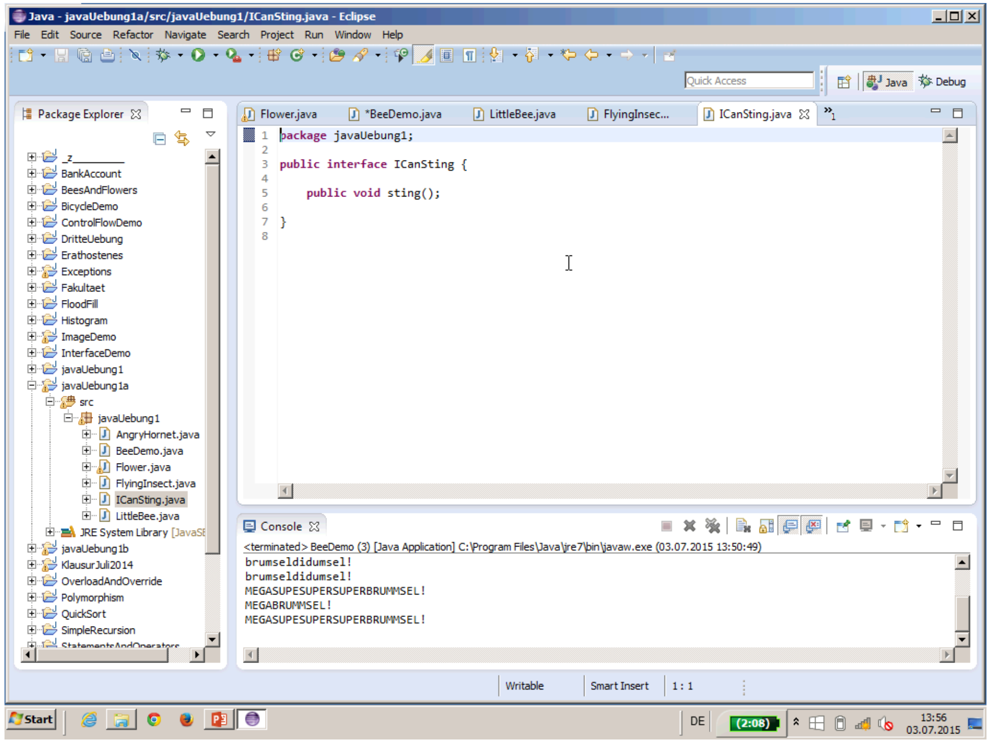
interface SubOrderApocrita {
        public void sting();
}

class LittleBee implements SubOrderApocrita {
        public void sting() {
                System.out.println("*pieks*");
        }
}

class AngryHornet implements SubOrderApocrita {
        public void sting() {
                System.out.println("*MEGAPIEKS*");
        }
}

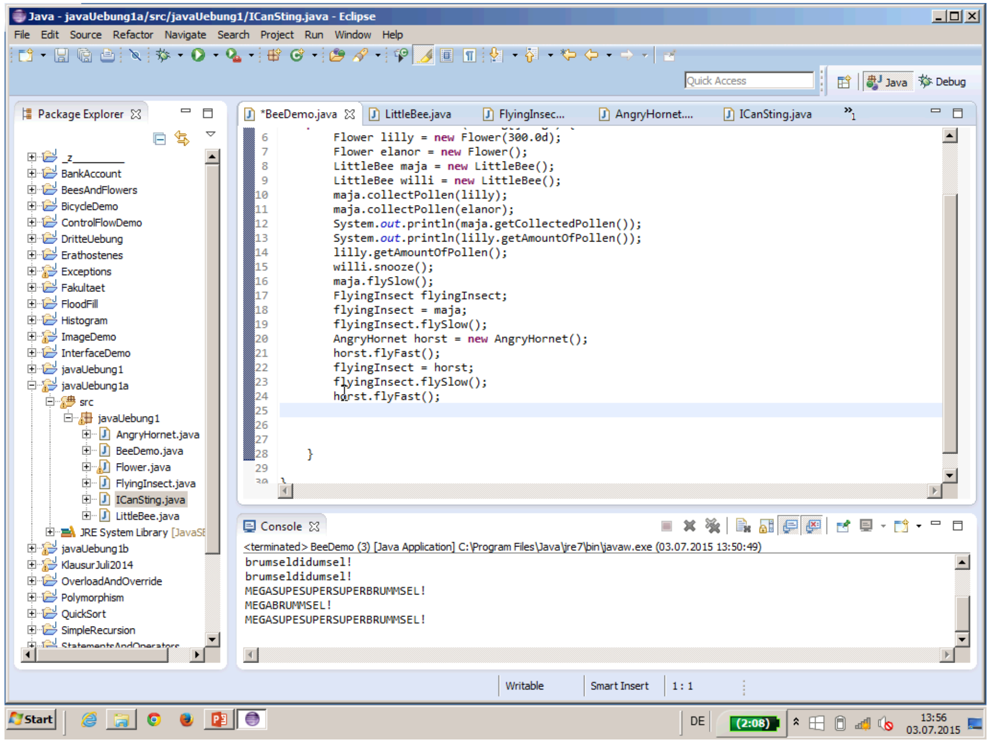
LittleBee maja = new LittleBee();
AngryHornet horst = new AngryHornet();
SubOrderApocrita someStinger;
someStinger = maja;
someStinger.sting();   // *pieks*
someStinger = horst;
someStinger.sting();   // *MEGAPIEKS*

129



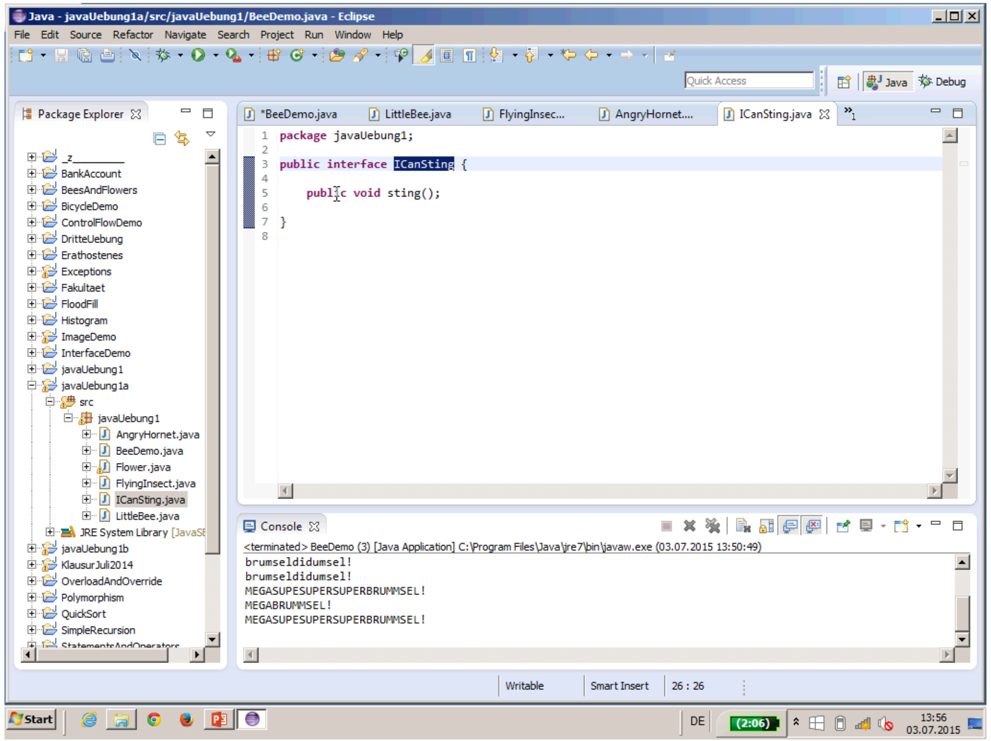
```
package javaUebung1;

public interface ICanSting {

    public void sting();

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
```



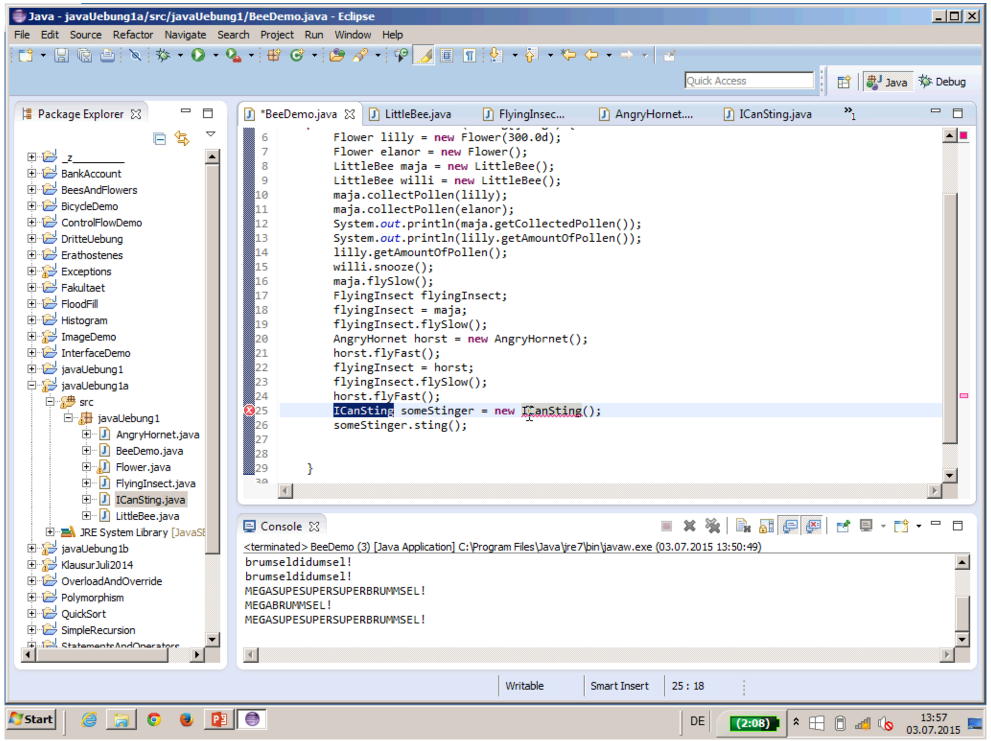
```
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();



    }
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
```



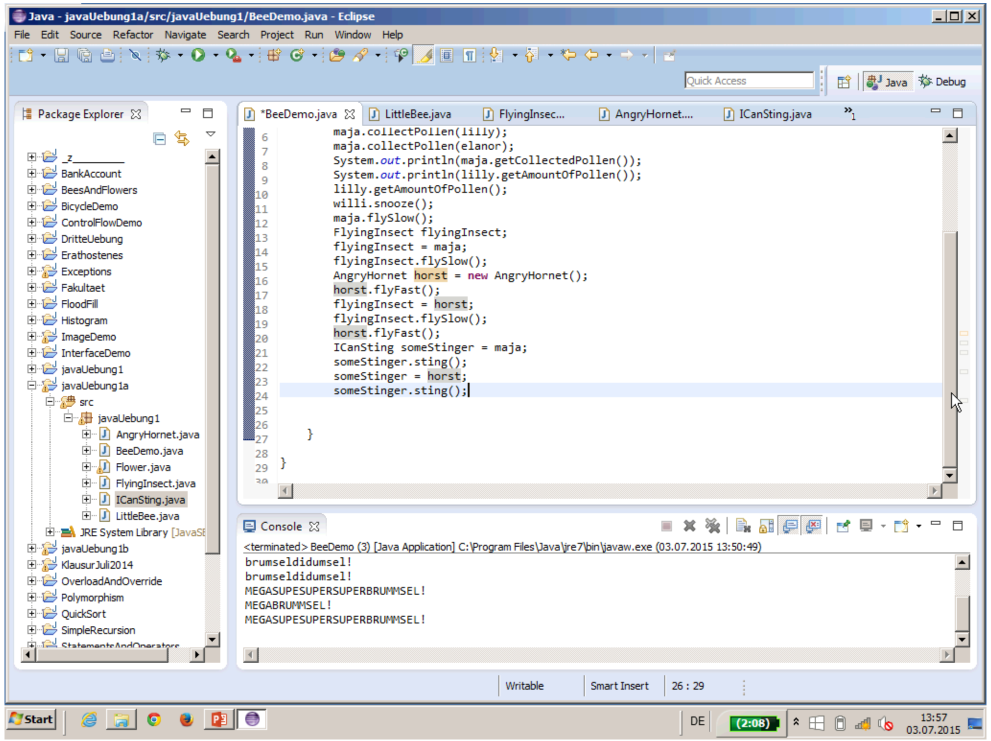
```
package javaUebung1;

public interface ICanSting {

    public void sting();

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
```



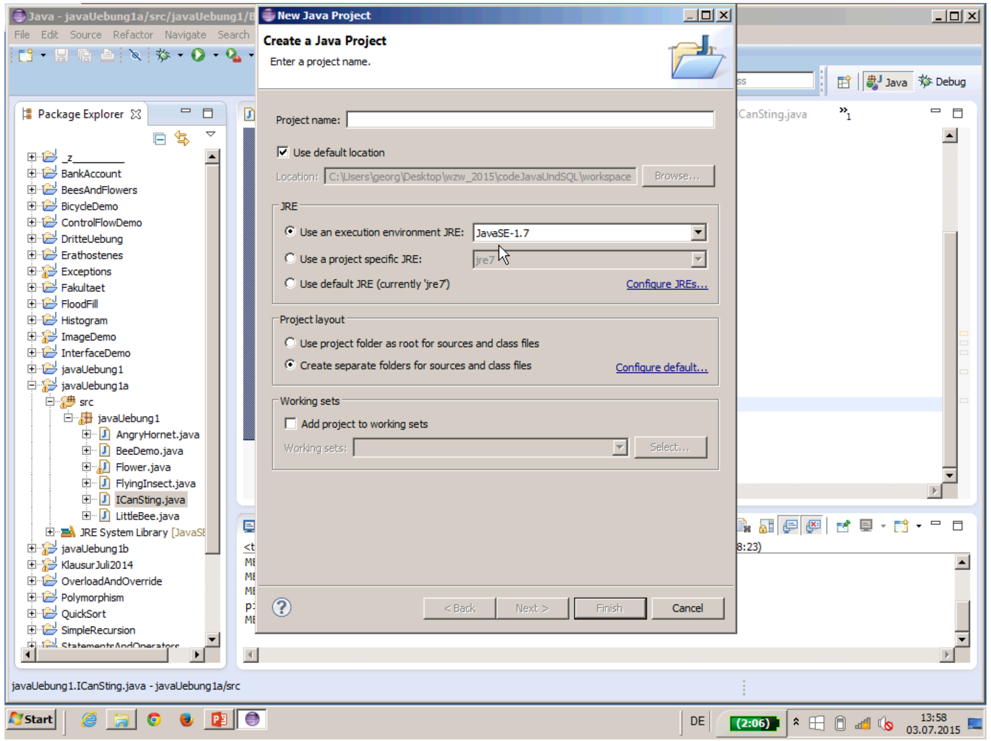
```
        Flower lilly = new Flower(300.0d);
        Flower elanor = new Flower();
        LittleBee maja = new LittleBee();
        LittleBee willi = new LittleBee();
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();
        ICanSting someStinger = new ICanSting();
        someStinger.sting();


    }
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
```



```
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();
        ICanSting someStinger = maja;
        someStinger.sting();
        someStinger = horst;
        someStinger.sting();


    }

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:50:49)
brumseldidumsel!
brumseldidumsel!
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
```


Java - javaUebung1a/src/javaUebung1/B...
New Java Project
Package Explorer
javaUebung1.ICanSting.java - javaUebung1a/src
13:58
03.07.2015

Create a Java Project

Enter a project name.

Project name:

☑ Use default location

Location: C:\Users\georg\Desktop\wzw_2015\codeJavaUndSQL\workspace

JRE

● Use an execution environment JRE: JavaSE-1.7

○ Use a project specific JRE: jre7

○ Use default JRE (currently 'jre7')

Project layout

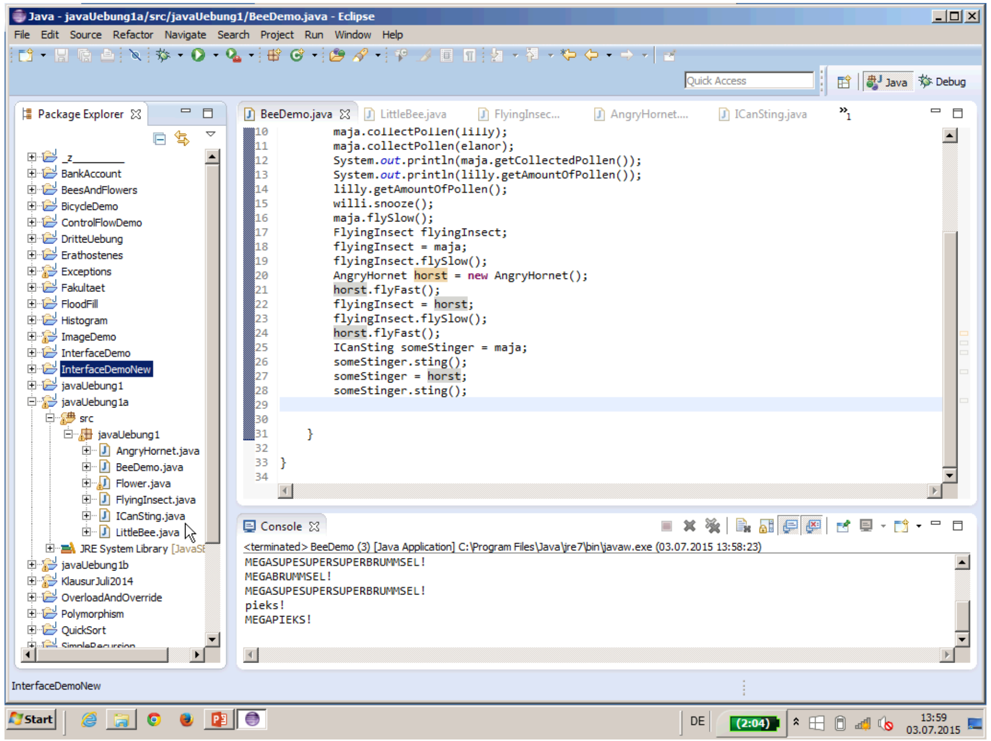
○ Use project folder as root for sources and class files

● Create separate folders for sources and class files

Working sets

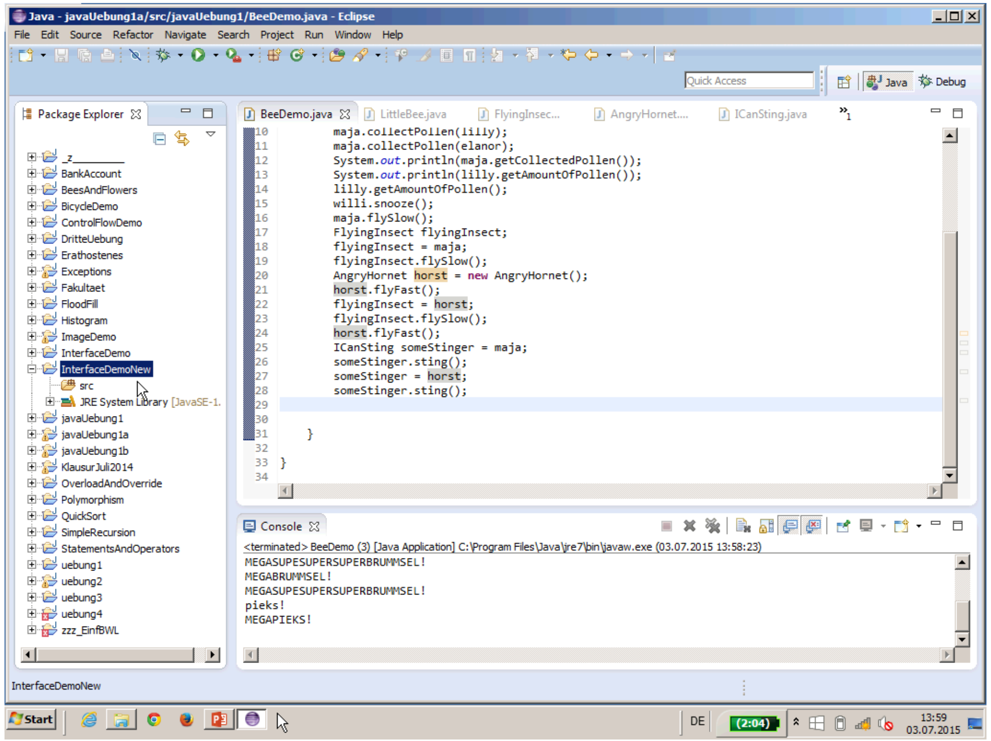
□ Add project to working sets

Working sets:



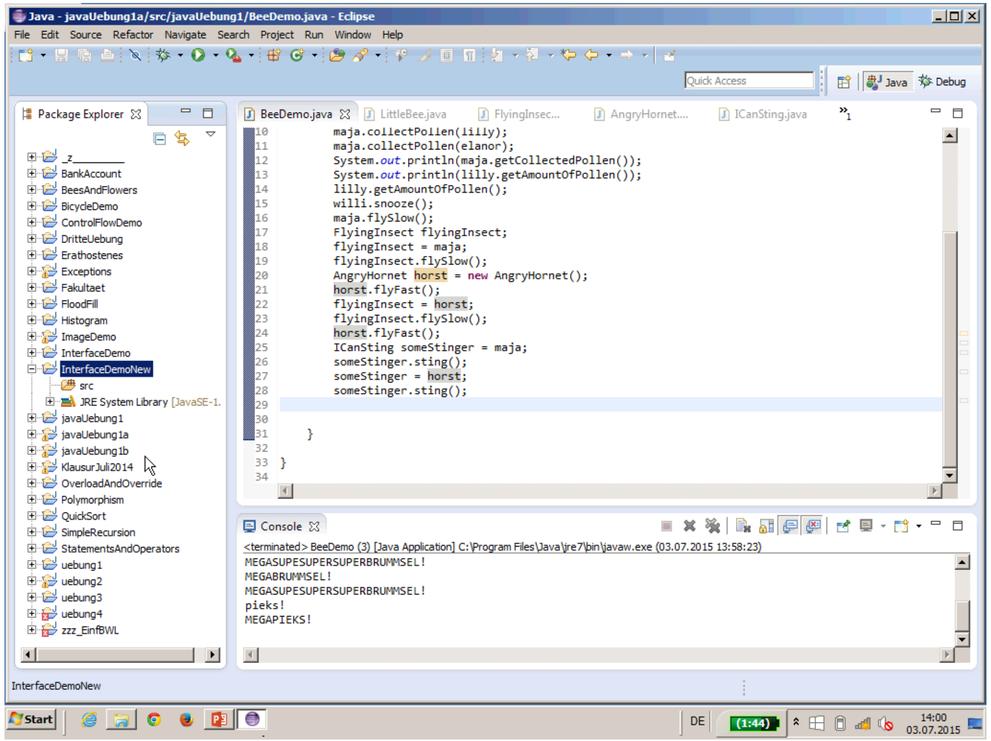
```
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();
        ICanSting someStinger = maja;
        someStinger.sting();
        someStinger = horst;
        someStinger.sting();


    }

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```


Java - javaUebung1a/src/javaUebung1/BeeDemo.java - Eclipse
Package Explorer
InterfaceDemoNew
src
JRE System Library [JavaSE-1.
InterfaceDemoNew
13:59
03.07.2015

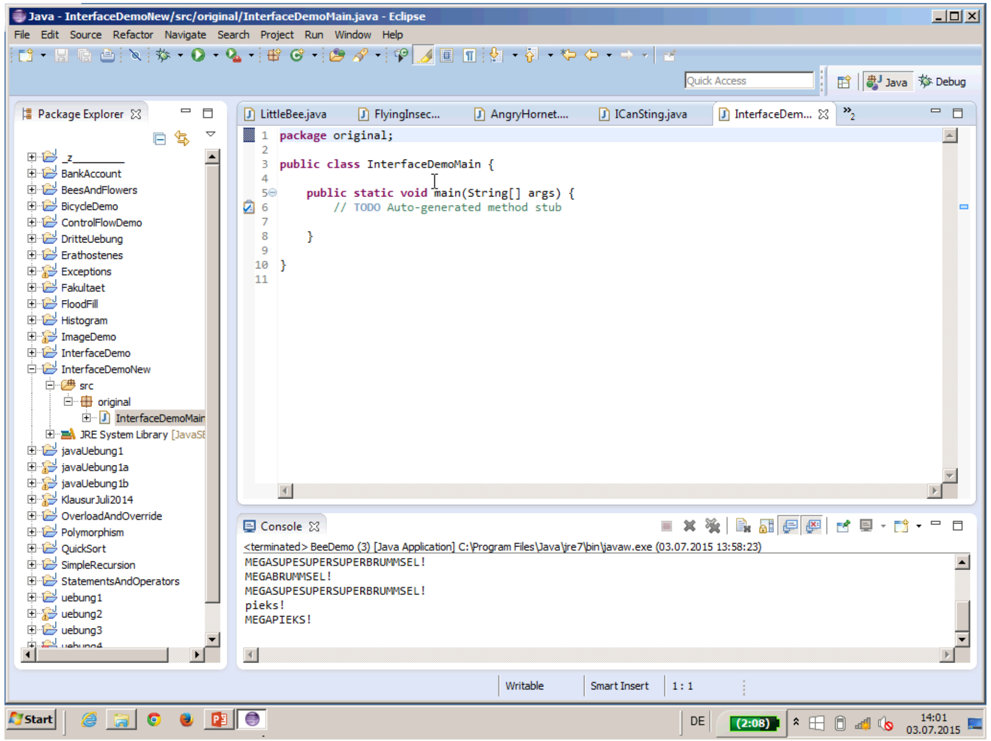
```
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();
        ICanSting someStinger = maja;
        someStinger.sting();
        someStinger = horst;
        someStinger.sting();


    }

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```


Java - javaUebung1a/src/javaUebung1/BeeDemo.java - Eclipse
Package Explorer
InterfaceDemoNew
src
JRE System Library [JavaSE-1.
InterfaceDemoNew
14:00
03.07.2015

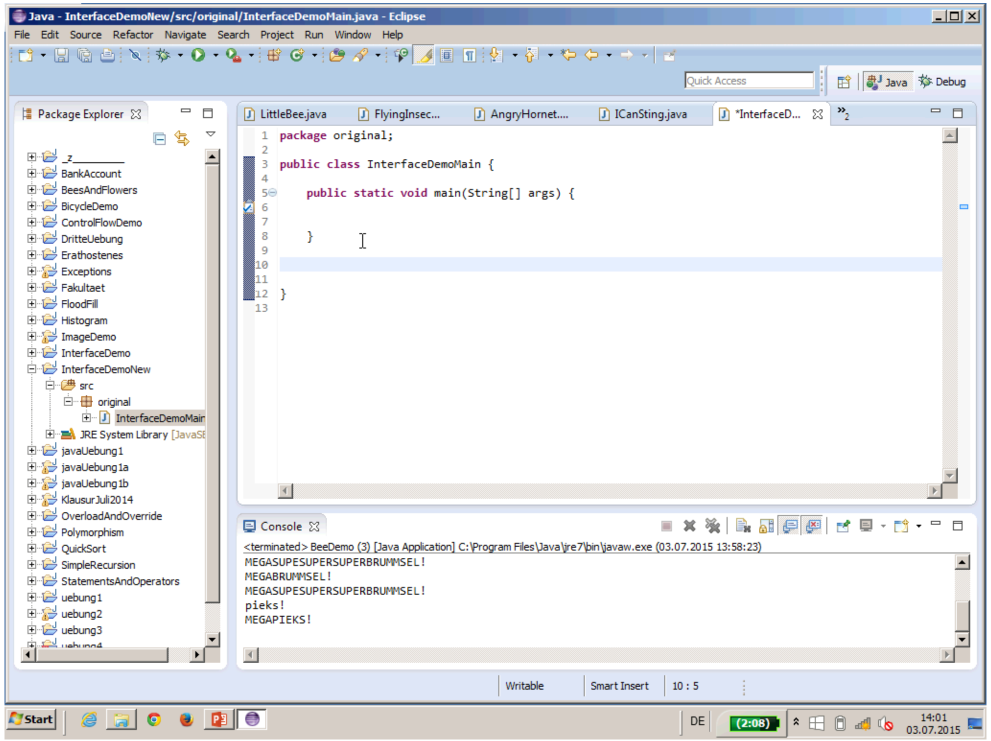
```
        maja.collectPollen(lilly);
        maja.collectPollen(elanor);
        System.out.println(maja.getCollectedPollen());
        System.out.println(lilly.getAmountOfPollen());
        lilly.getAmountOfPollen();
        willi.snooze();
        maja.flySlow();
        FlyingInsect flyingInsect;
        flyingInsect = maja;
        flyingInsect.flySlow();
        AngryHornet horst = new AngryHornet();
        horst.flyFast();
        flyingInsect = horst;
        flyingInsect.flySlow();
        horst.flyFast();
        ICanSting someStinger = maja;
        someStinger.sting();
        someStinger = horst;
        someStinger.sting();


    }

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



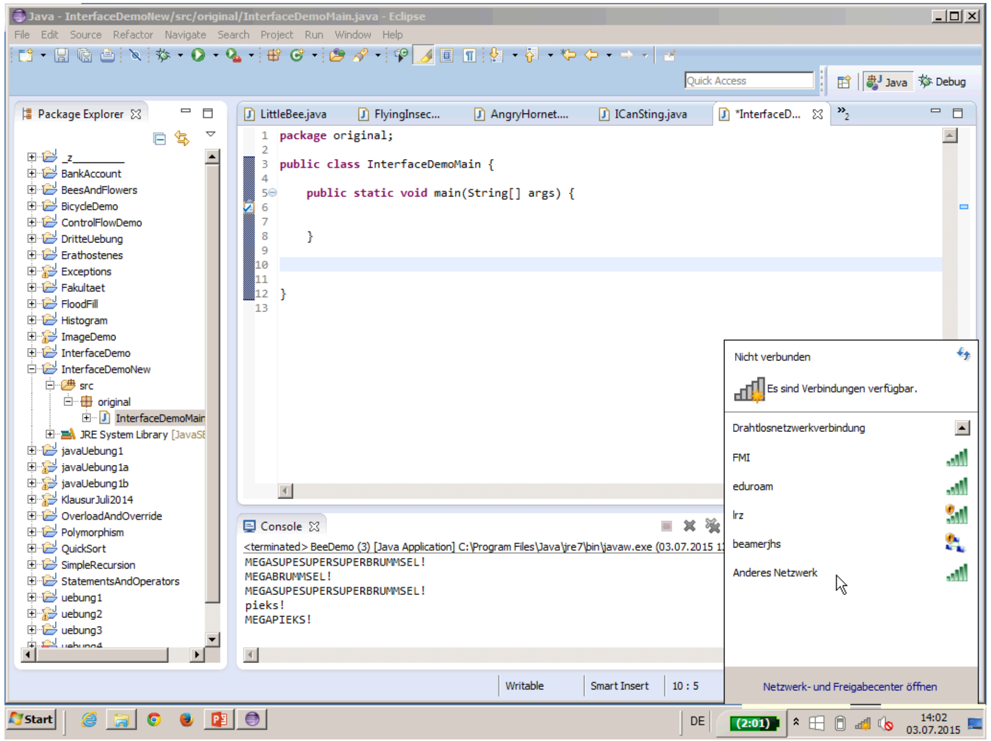
```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {

    }


}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {

    }


}
```

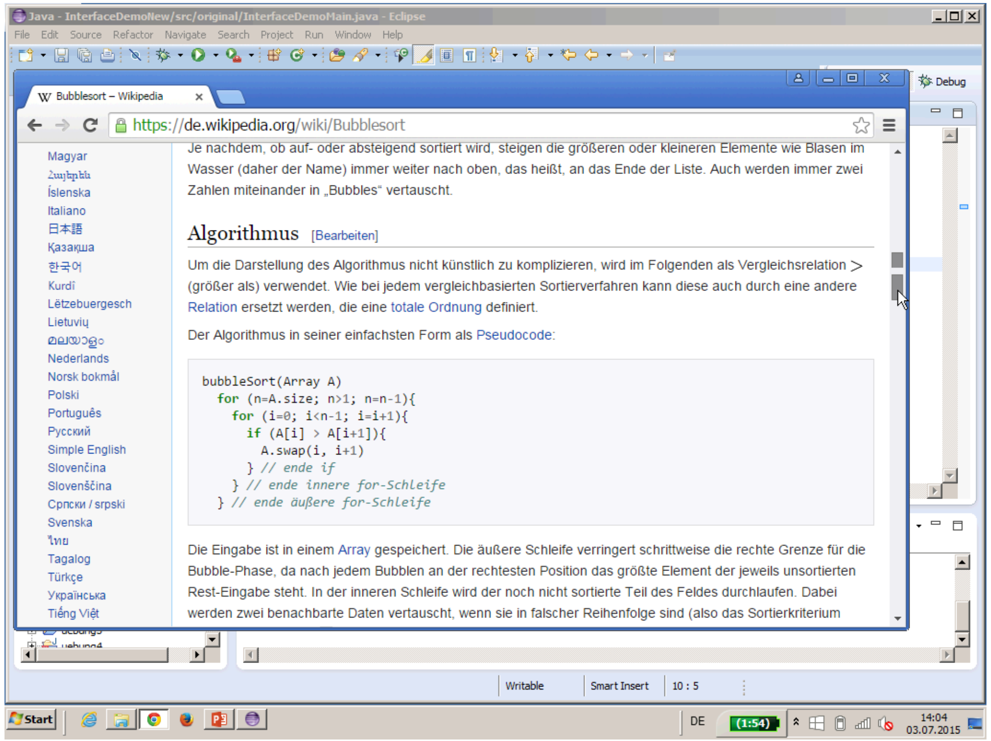
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 1
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

Nicht verbunden

Es sind Verbindungen verfügbar.

Drahtlosnetzwerkverbindung
FMI
eduroam
lrz
beamerjhs
Anderes Netzwerk

Netzwerk- und Freigabecenter öffnen



Je nachdem, ob auf- oder absteigend sortiert wird, steigen die größeren oder kleineren Elemente wie Blasen im Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

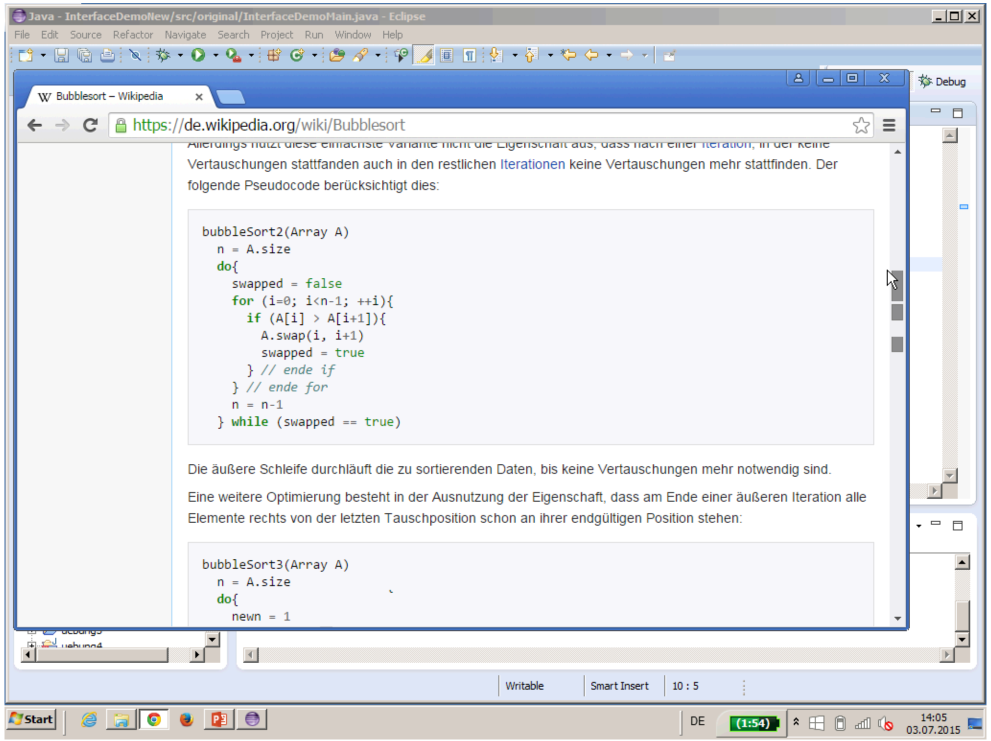
Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation > (größer als) verwendet. Wie bei jedem vergleichbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

```
bubbleSort(Array A)
  for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
      } // ende if
    } // ende innere for-Schleife
  } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubblen an der rechtesten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium


Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Bubblesort – Wikipedia
https://de.wikipedia.org/wiki/Bubblesort
Writable
Smart Insert
10 : 5
Start
DE
(1:54)
14:05
03.07.2015

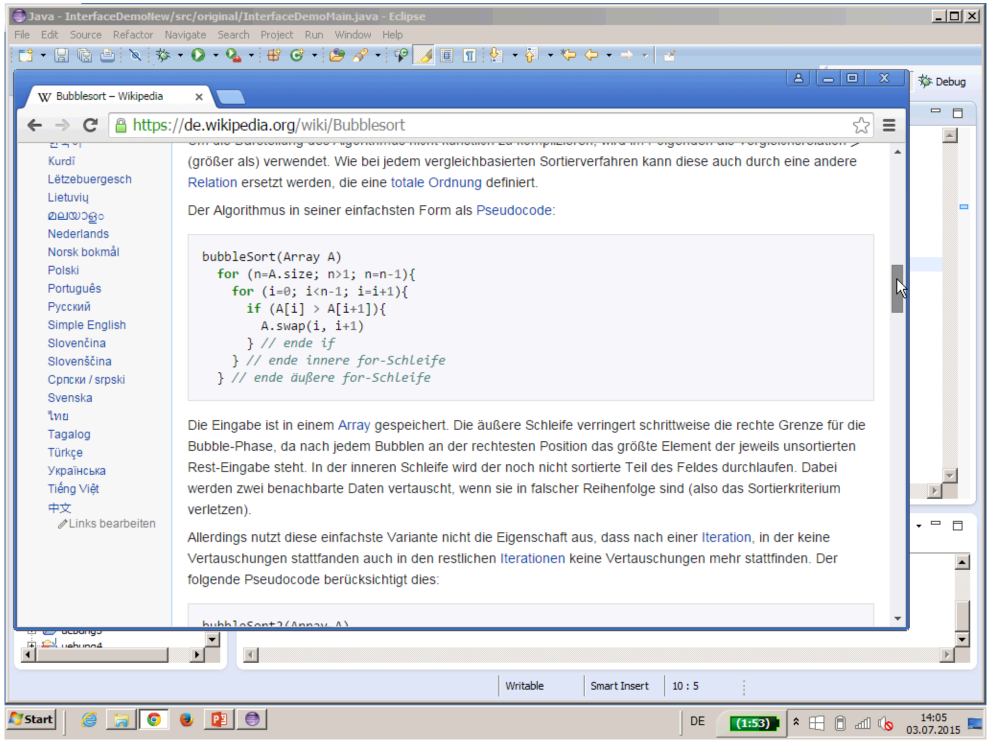
Allerdings nutzt diese einfachste Variante nicht die Eigenschaft aus, dass nach einer Iteration, in der keine Vertauschungen stattfanden auch in den restlichen Iterationen keine Vertauschungen mehr stattfinden. Der folgende Pseudocode berücksichtigt dies:

```
bubbleSort2(Array A)
  n = A.size
  do{
    swapped = false
    for (i=0; i<n-1; ++i){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
        swapped = true
      } // ende if
    } // ende for
    n = n-1
  } while (swapped == true)
```

Die äußere Schleife durchläuft die zu sortierenden Daten, bis keine Vertauschungen mehr notwendig sind.

Eine weitere Optimierung besteht in der Ausnutzung der Eigenschaft, dass am Ende einer äußeren Iteration alle Elemente rechts von der letzten Tauschposition schon an ihrer endgültigen Position stehen:

```
bubbleSort3(Array A)
  n = A.size
  do{
    newn = 1
```



Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation > (größer als) verwendet. Wie bei jedem vergleichsbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

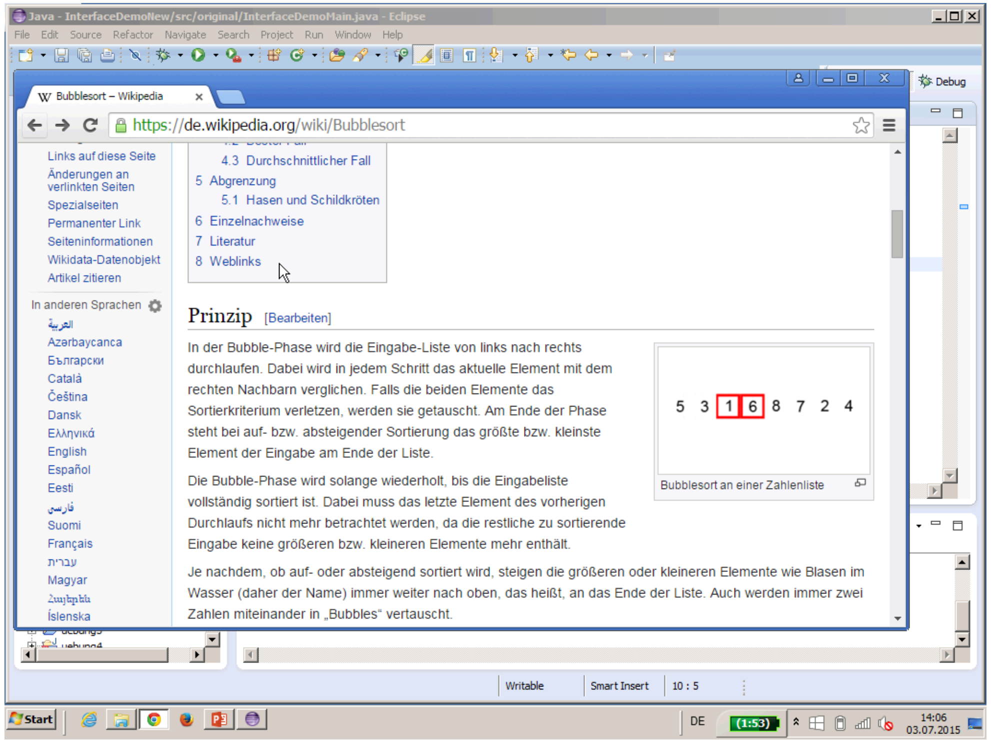
Der Algorithmus in seiner einfachsten Form als Pseudocode:

```
bubbleSort(Array A)
  for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
      } // ende if
    } // ende innere for-Schleife
  } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubblen an der rechtesten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium verletzen).

Allerdings nutzt diese einfachste Variante nicht die Eigenschaft aus, dass nach einer Iteration, in der keine Vertauschungen stattfanden auch in den restlichen Iterationen keine Vertauschungen mehr stattfinden. Der folgende Pseudocode berücksichtigt dies:

```
bubbleSort2(Array A)
```



4.2 Bester Fall
4.3 Durchschnittlicher Fall
5 Abgrenzung
5.1 Hasen und Schildkröten
6 Einzelnachweise
7 Literatur
8 Weblinks

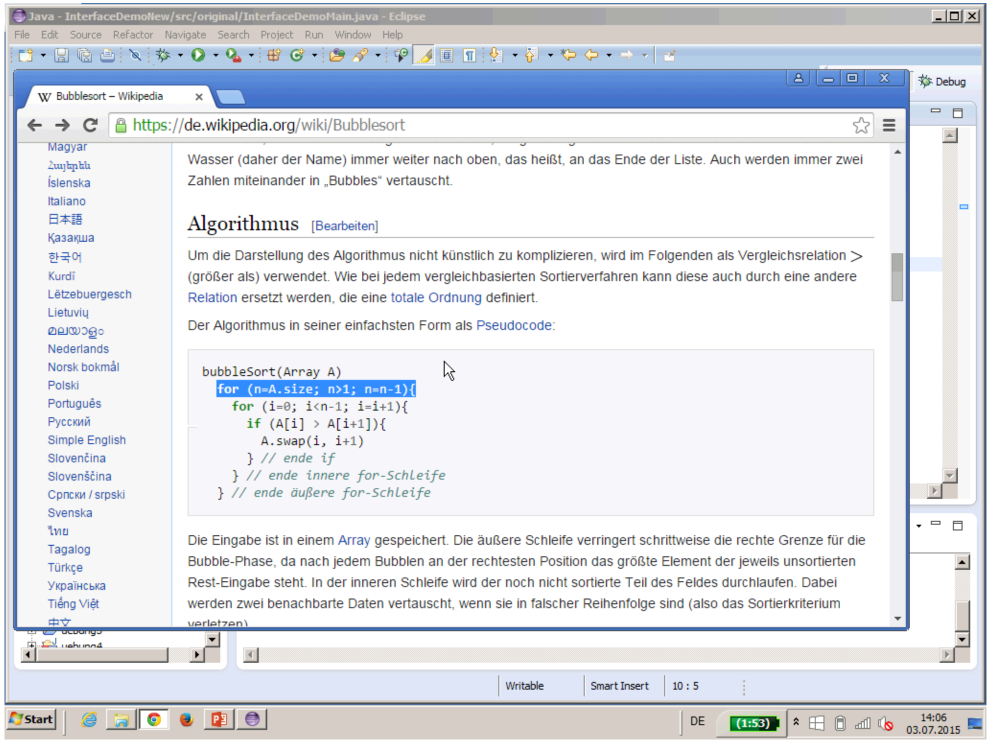
Prinzip [Bearbeiten]

In der Bubble-Phase wird die Eingabe-Liste von links nach rechts durchlaufen. Dabei wird in jedem Schritt das aktuelle Element mit dem rechten Nachbarn verglichen. Falls die beiden Elemente das Sortierkriterium verletzen, werden sie getauscht. Am Ende der Phase steht bei auf- bzw. absteigender Sortierung das größte bzw. kleinste Element der Eingabe am Ende der Liste.

Bubblesort an einer Zahlenliste

Die Bubble-Phase wird solange wiederholt, bis die Eingabeliste vollständig sortiert ist. Dabei muss das letzte Element des vorherigen Durchlaufs nicht mehr betrachtet werden, da die restliche zu sortierende Eingabe keine größeren bzw. kleineren Elemente mehr enthält.

Je nachdem, ob auf- oder absteigend sortiert wird, steigen die größeren oder kleineren Elemente wie Blasen im Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.



Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

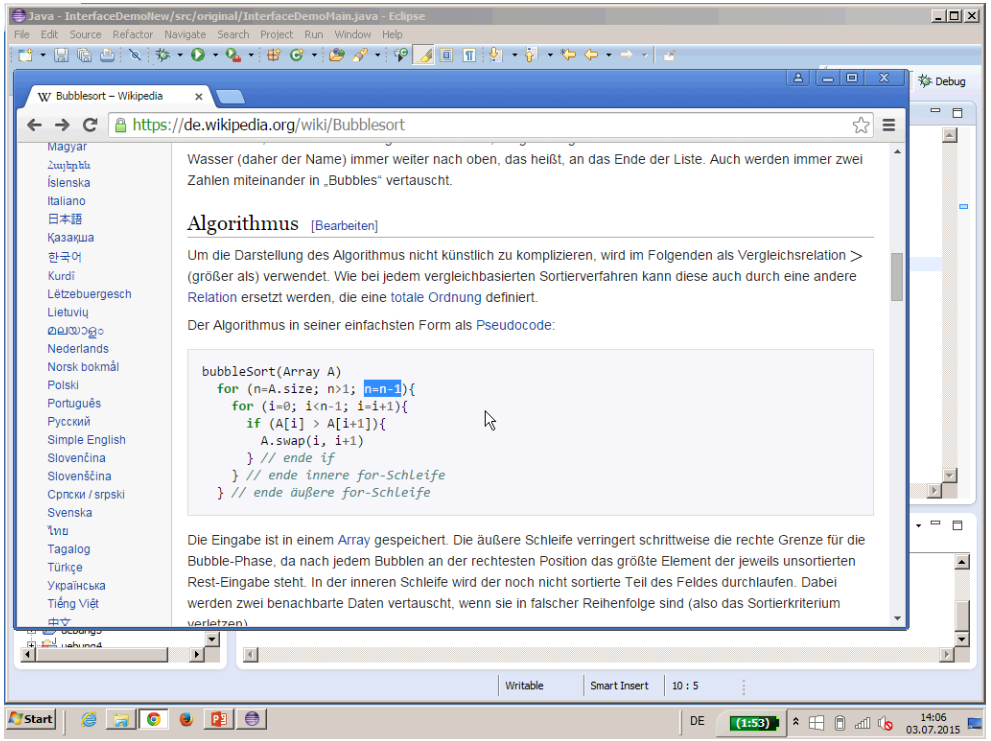
Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation > (größer als) verwendet. Wie bei jedem vergleichsbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

```
bubbleSort(Array A)
  for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
      } // ende if
    } // ende innere for-Schleife
  } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubblen an der rechtesten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium verletzen).



Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

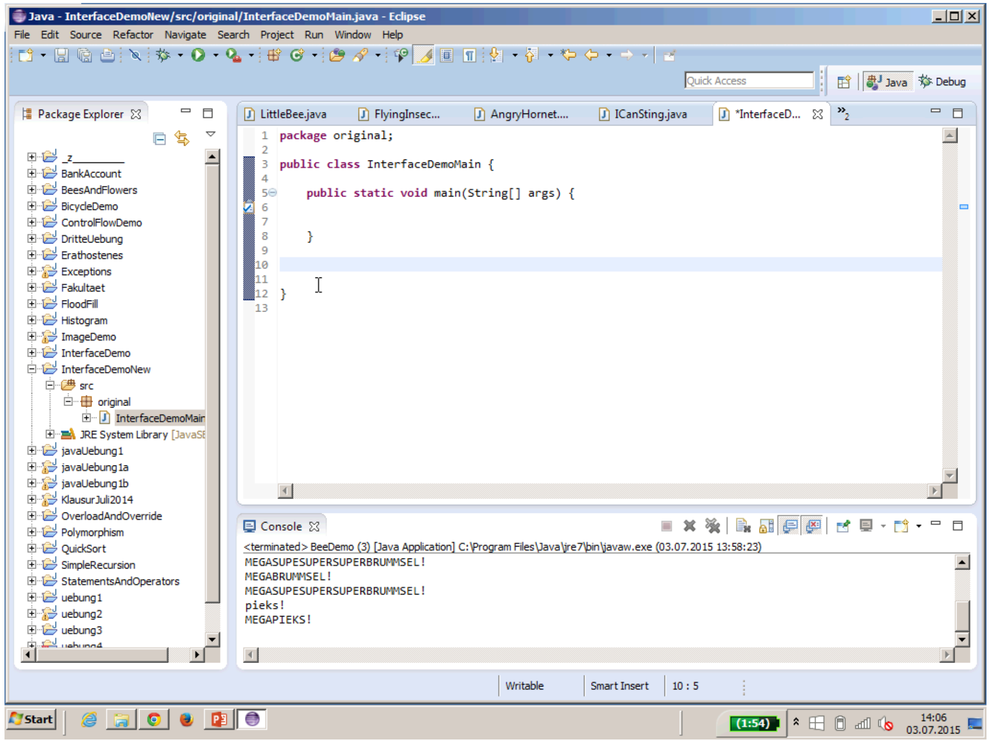
Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation > (größer als) verwendet. Wie bei jedem vergleichbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

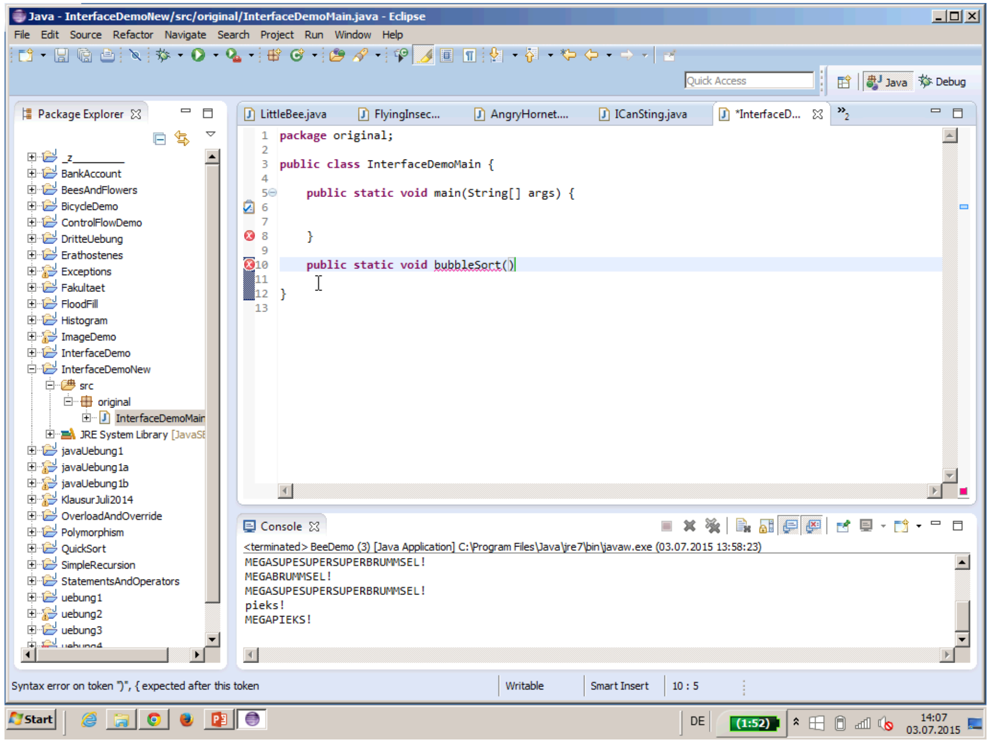
```
bubbleSort(Array A)
  for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
      } // ende if
    } // ende innere for-Schleife
  } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubblen an der rechtesten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium verletzen)



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }


}
```

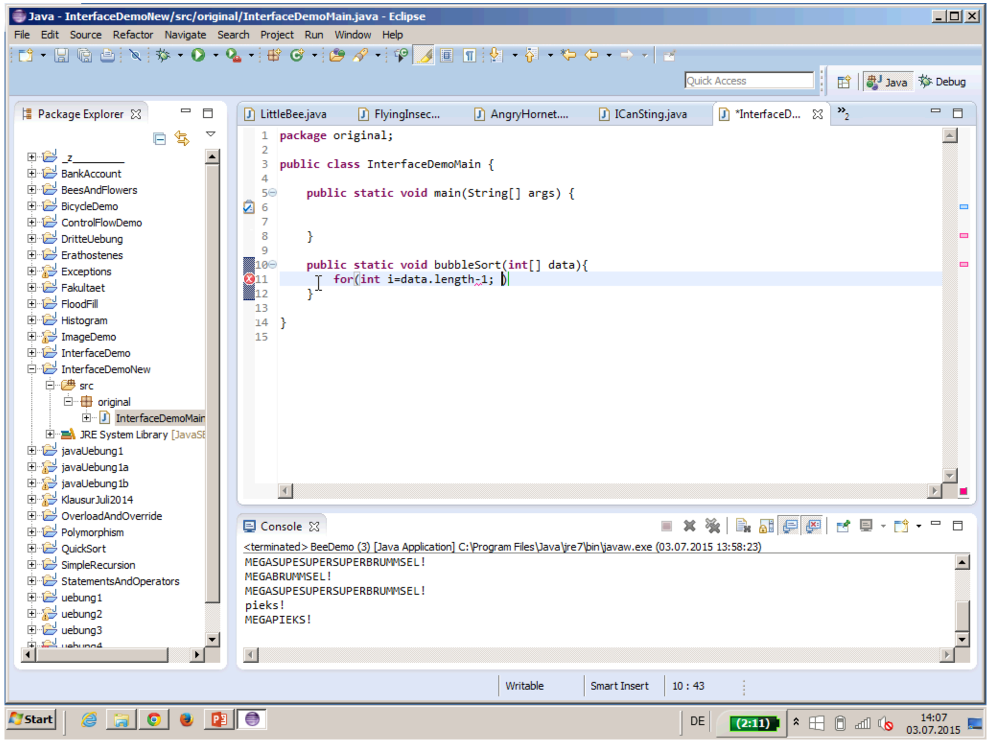
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort()

}
```

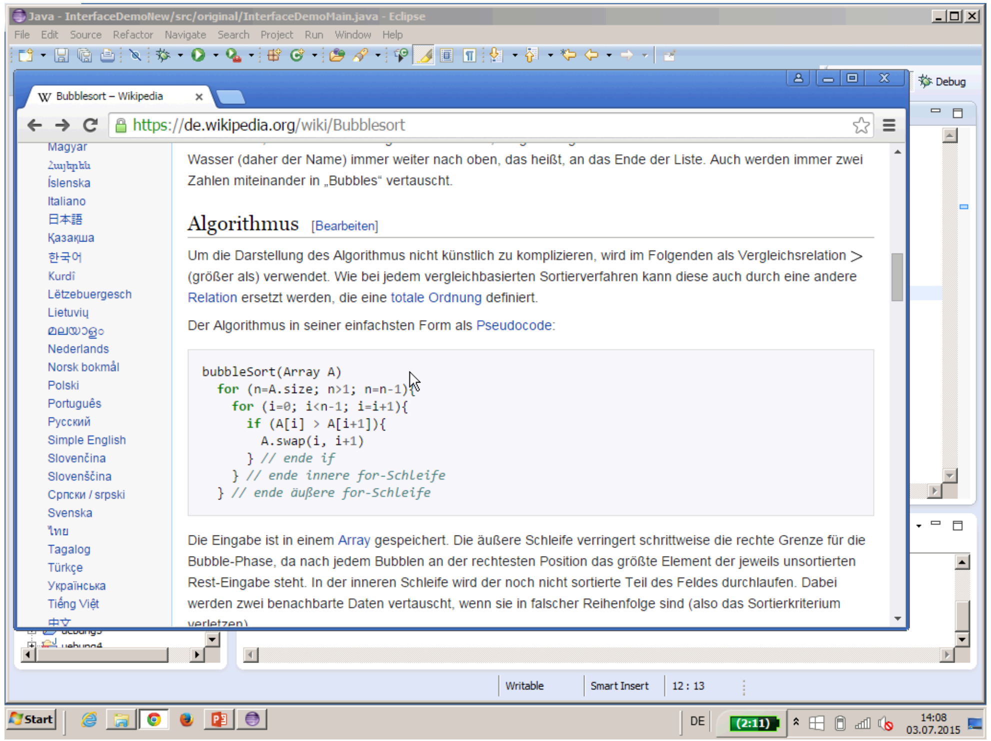
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

Syntax error on token ")", { expected after this token



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; )
    }

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

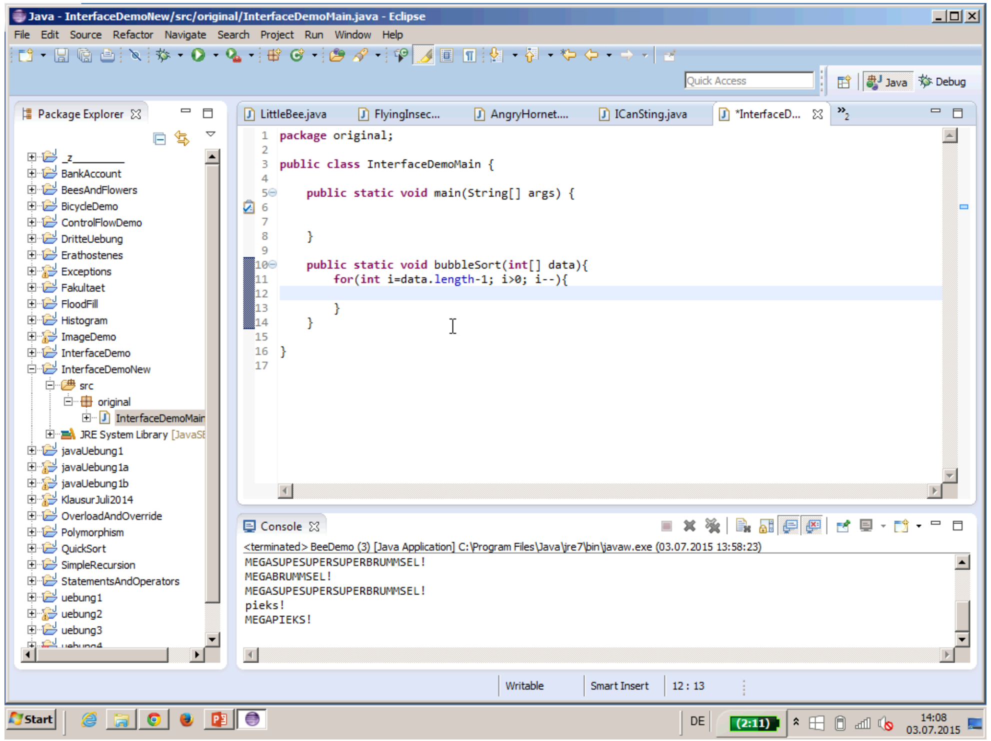
Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation > (größer als) verwendet. Wie bei jedem vergleichsbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

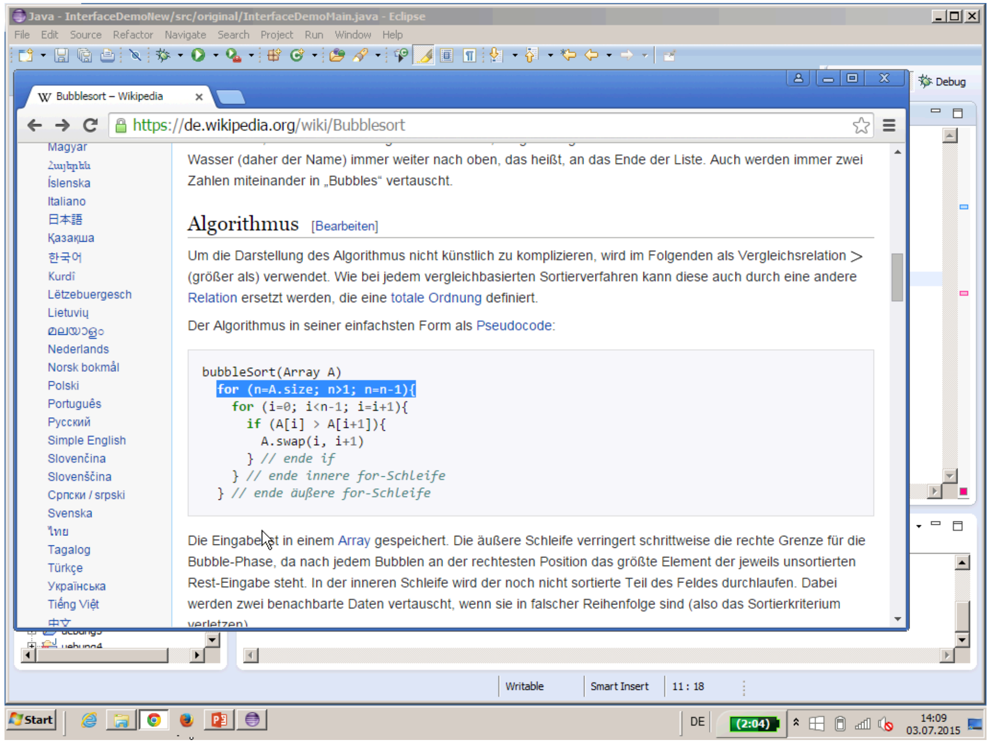
```
bubbleSort(Array A)
  for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
      } // ende if
    } // ende innere for-Schleife
  } // ende äußere for-Schleife
```

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubblen an der rechtesten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium verletzen)



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){

        }
    }

}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```


Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
Bubblesort – Wikipedia
https://de.wikipedia.org/wiki/Bubblesort
Debug
Writable
Smart Insert
11 : 18
Start
DE
14:09
03.07.2015

Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

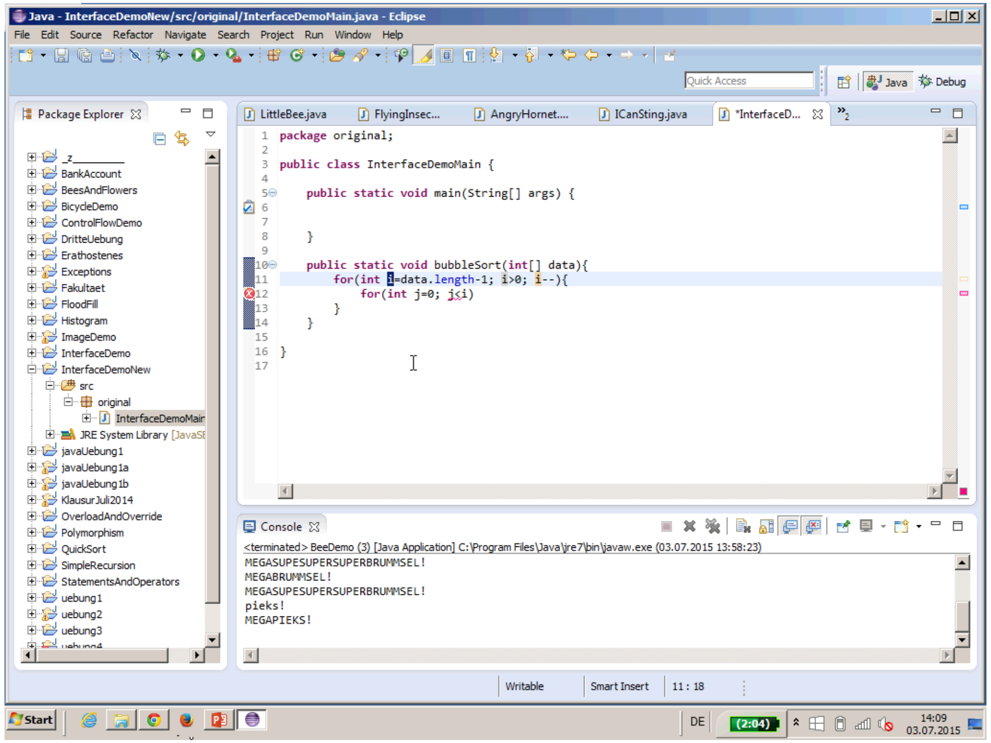
Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation > (größer als) verwendet. Wie bei jedem vergleichsbasierten Sortierverfahren kann diese auch durch eine andere Relation ersetzt werden, die eine totale Ordnung definiert.

Der Algorithmus in seiner einfachsten Form als Pseudocode:

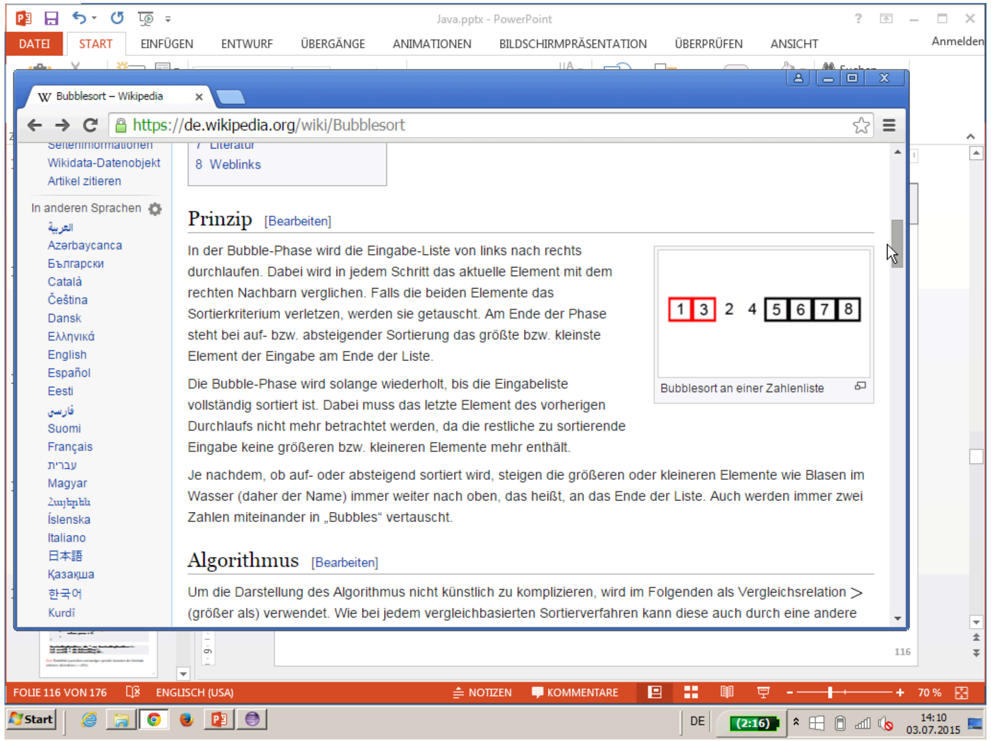
bubbleSort(Array A)
  for (n=A.size; n>1; n=n-1){
    for (i=0; i<n-1; i=i+1){
      if (A[i] > A[i+1]){
        A.swap(i, i+1)
      } // ende if
    } // ende innere for-Schleife
  } // ende äußere for-Schleife

Die Eingabe ist in einem Array gespeichert. Die äußere Schleife verringert schrittweise die rechte Grenze für die Bubble-Phase, da nach jedem Bubblen an der rechtesten Position das größte Element der jeweils unsortierten Rest-Eingabe steht. In der inneren Schleife wird der noch nicht sortierte Teil des Feldes durchlaufen. Dabei werden zwei benachbarte Daten vertauscht, wenn sie in falscher Reihenfolge sind (also das Sortierkriterium verletzen)



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i)
        }
    }

}
```

<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!



Prinzip [Bearbeiten]

In der Bubble-Phase wird die Eingabe-Liste von links nach rechts durchlaufen. Dabei wird in jedem Schritt das aktuelle Element mit dem rechten Nachbarn verglichen. Falls die beiden Elemente das Sortierkriterium verletzen, werden sie getauscht. Am Ende der Phase steht bei auf- bzw. absteigender Sortierung das größte bzw. kleinste Element der Eingabe am Ende der Liste.

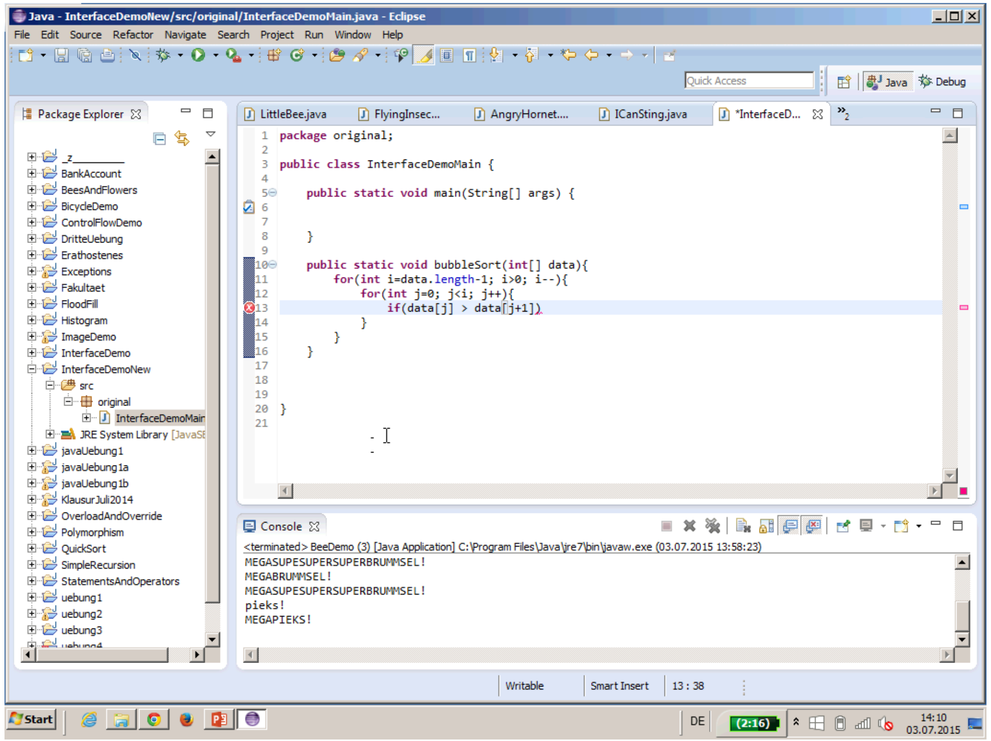
Bubblesort an einer Zahlenliste

Die Bubble-Phase wird solange wiederholt, bis die Eingabeliste vollständig sortiert ist. Dabei muss das letzte Element des vorherigen Durchlaufs nicht mehr betrachtet werden, da die restliche zu sortierende Eingabe keine größeren bzw. kleineren Elemente mehr enthält.

Je nachdem, ob auf- oder absteigend sortiert wird, steigen die größeren oder kleineren Elemente wie Blasen im Wasser (daher der Name) immer weiter nach oben, das heißt, an das Ende der Liste. Auch werden immer zwei Zahlen miteinander in „Bubbles“ vertauscht.

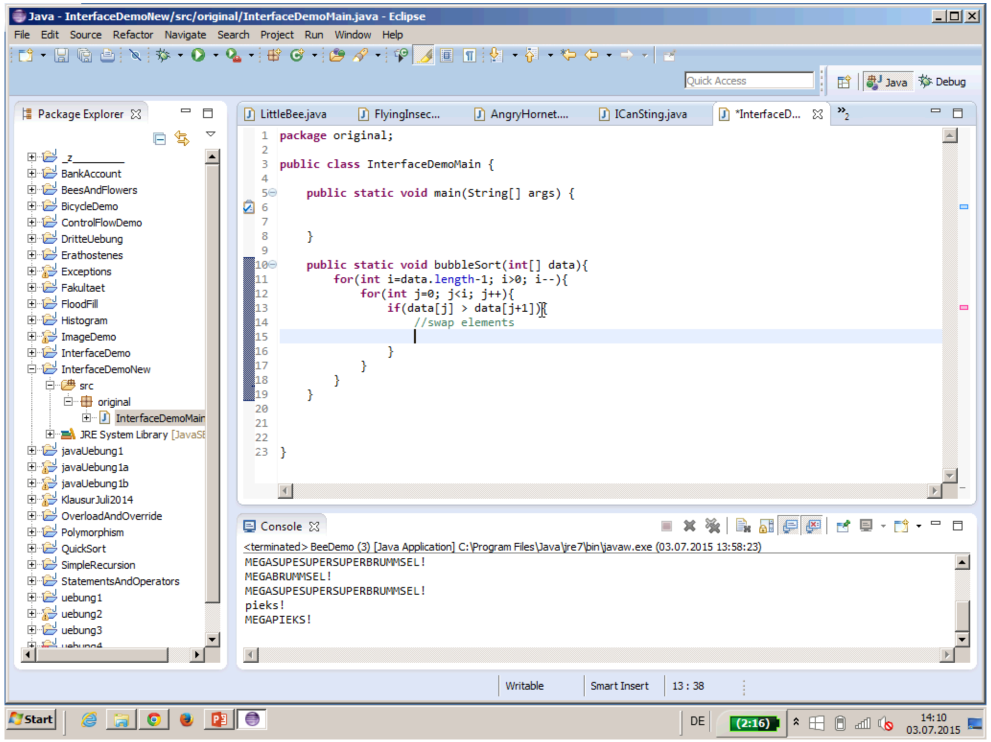
Algorithmus [Bearbeiten]

Um die Darstellung des Algorithmus nicht künstlich zu komplizieren, wird im Folgenden als Vergleichsrelation > (größer als) verwendet. Wie bei jedem vergleichbasierten Sortierverfahren kann diese auch durch eine andere



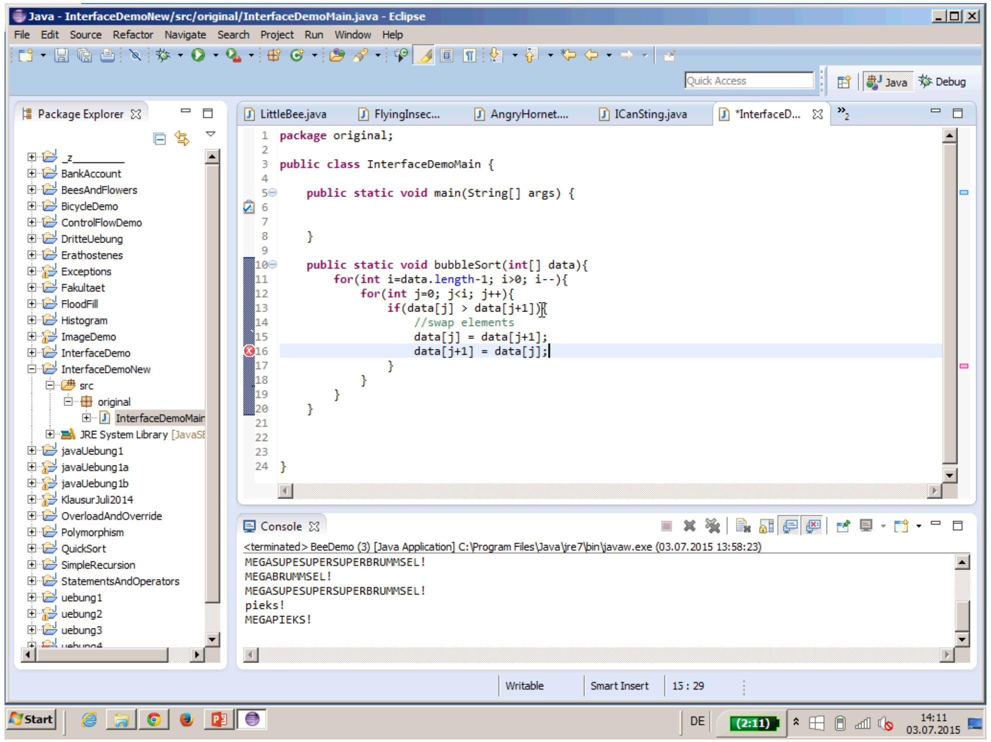
```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1])
            }
        }
    }



}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



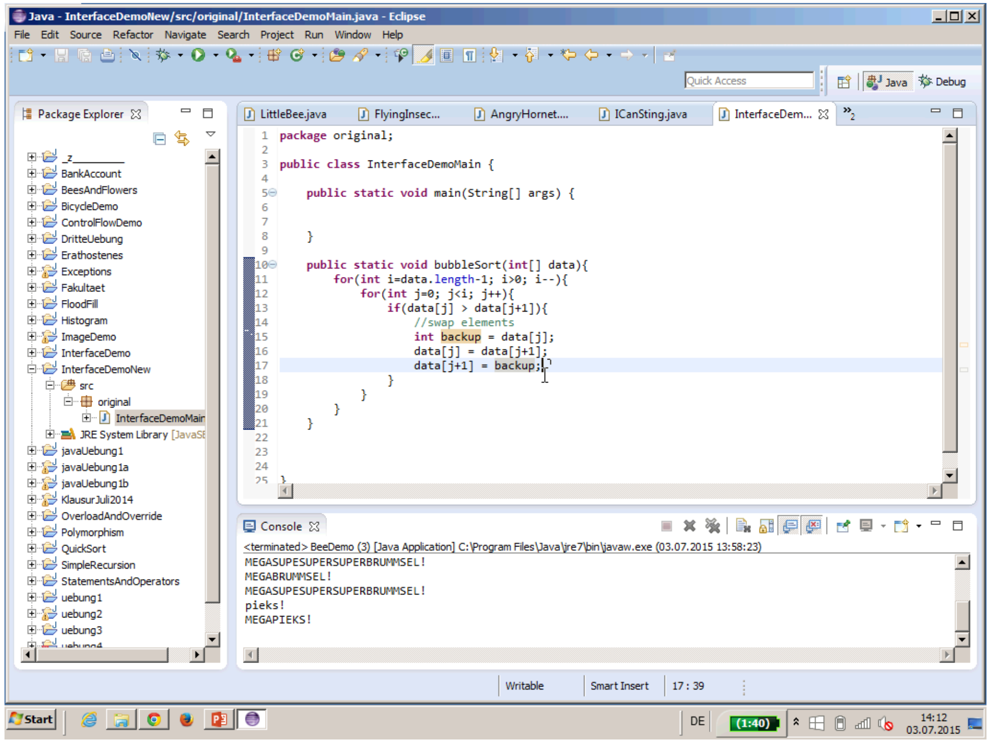
```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements

                }
            }
        }
    }



}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    data[j] = data[j+1];
                    data[j+1] = data[j];
                }
            }
        }
    }



}
```

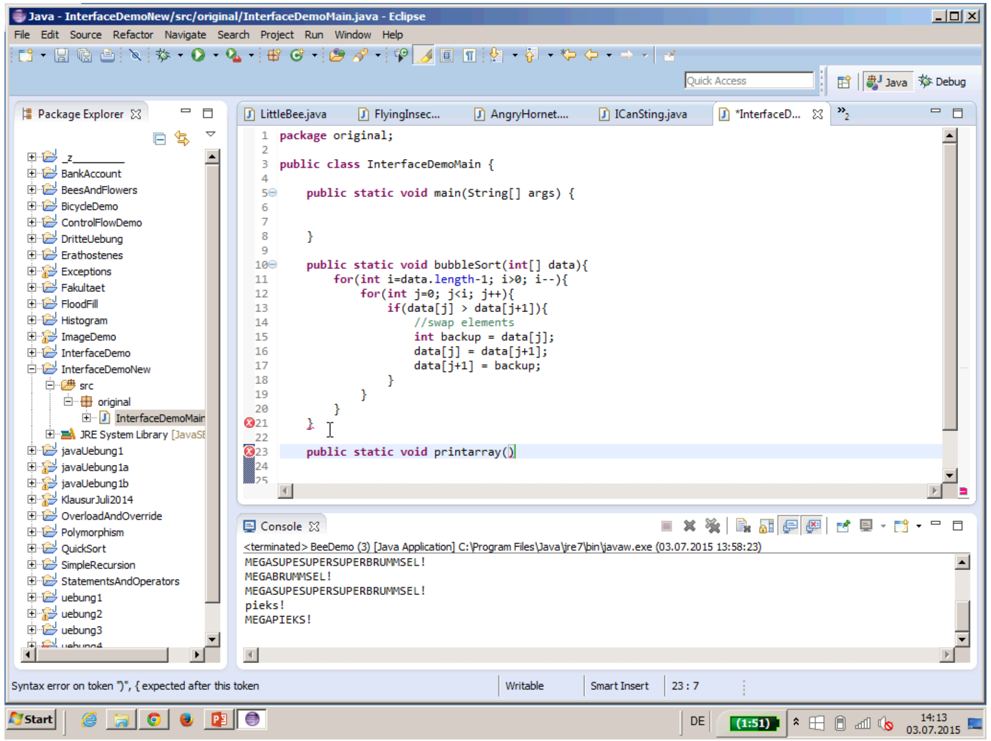
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }



}
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

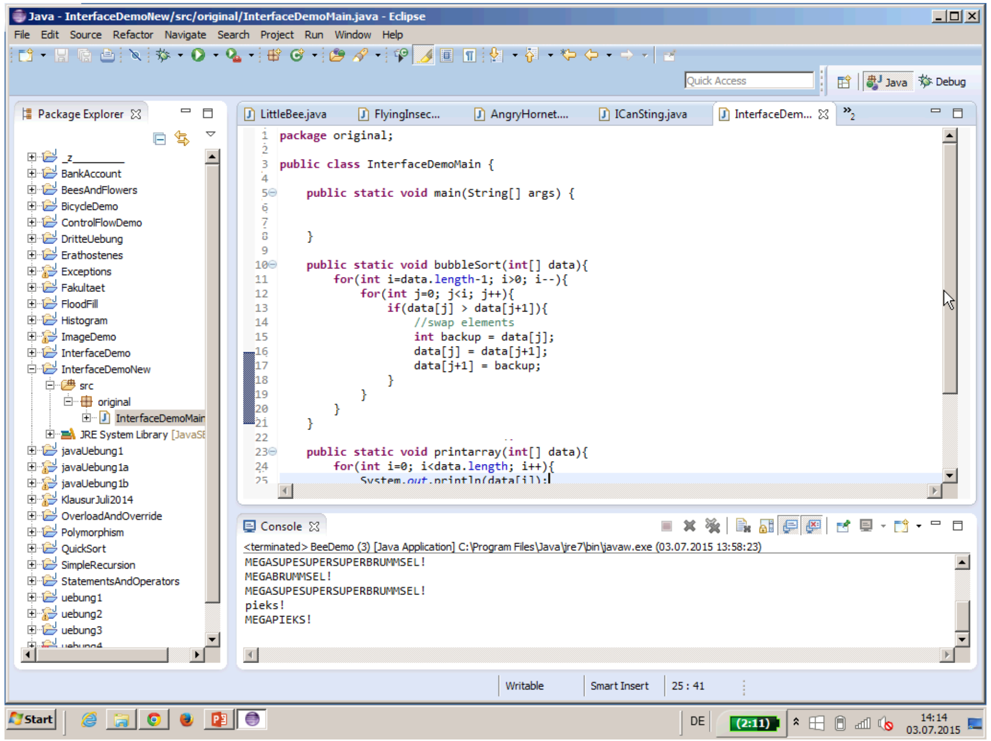
Writable | Smart Insert | 17 : 39



```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printarray()
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

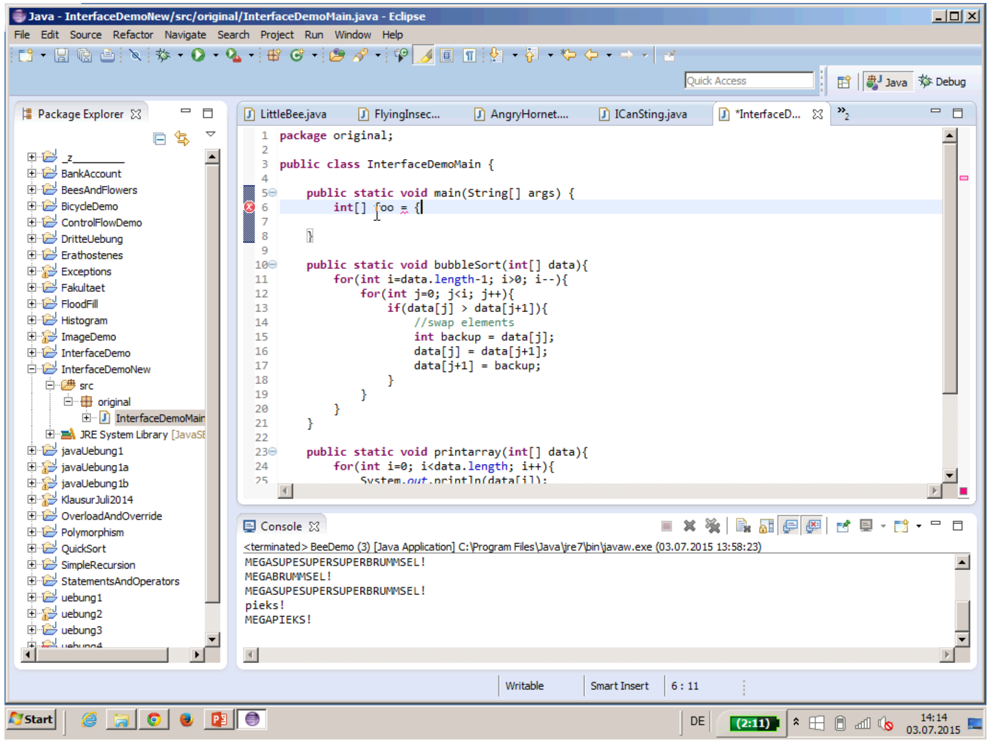
Syntax error on token ")", { expected after this token | Writable | Smart Insert | 23 : 7



```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printarray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

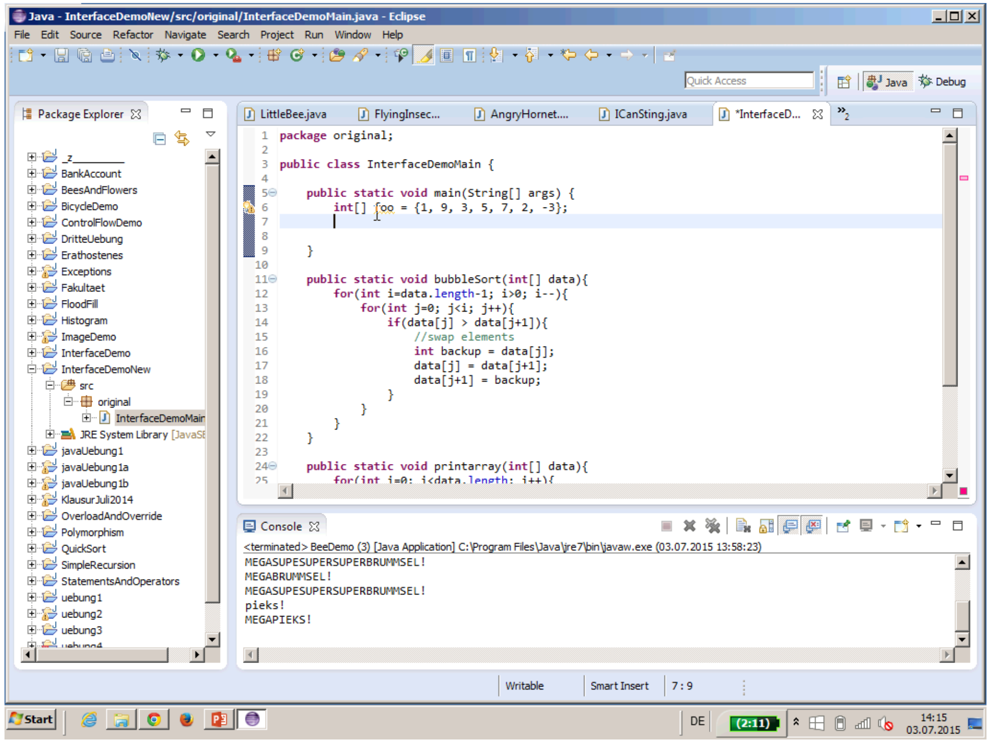
Writable | Smart Insert | 25 : 41



```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printarray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
```

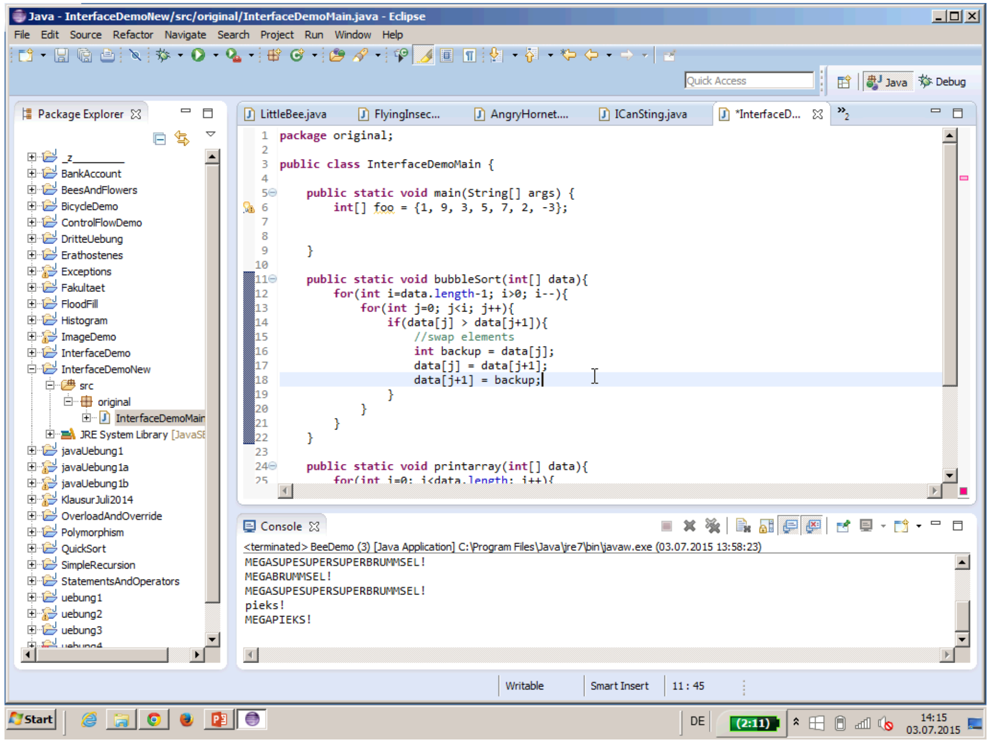
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```

Writable | Smart Insert | 6 : 11



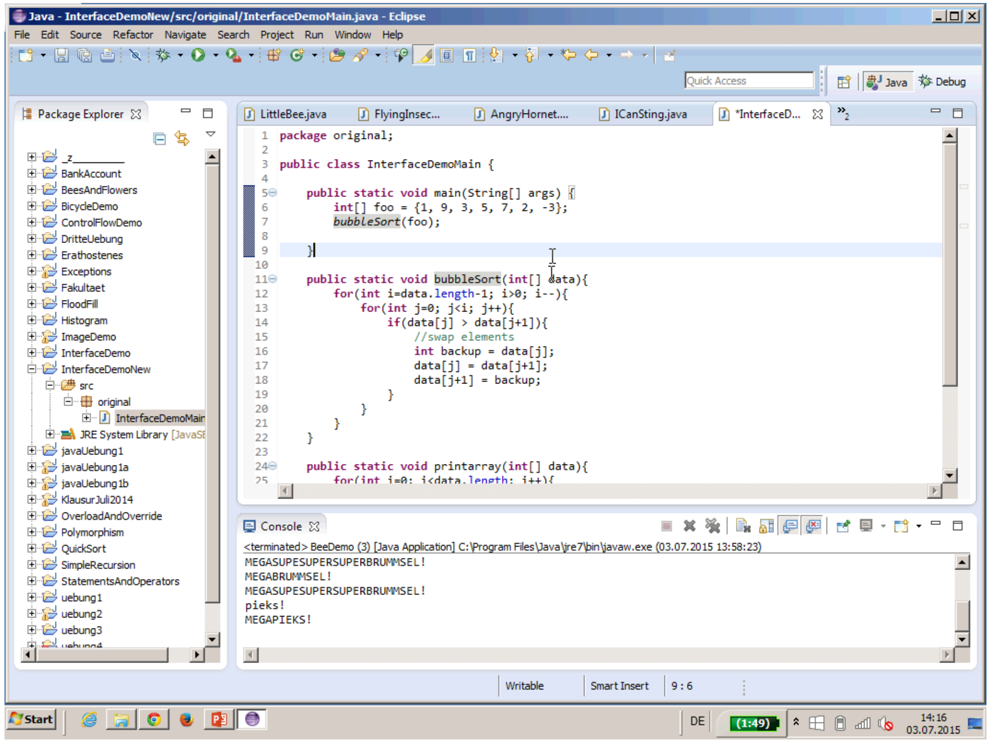
```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printarray(int[] data){
        for(int i=0; i<data.length; i++){
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



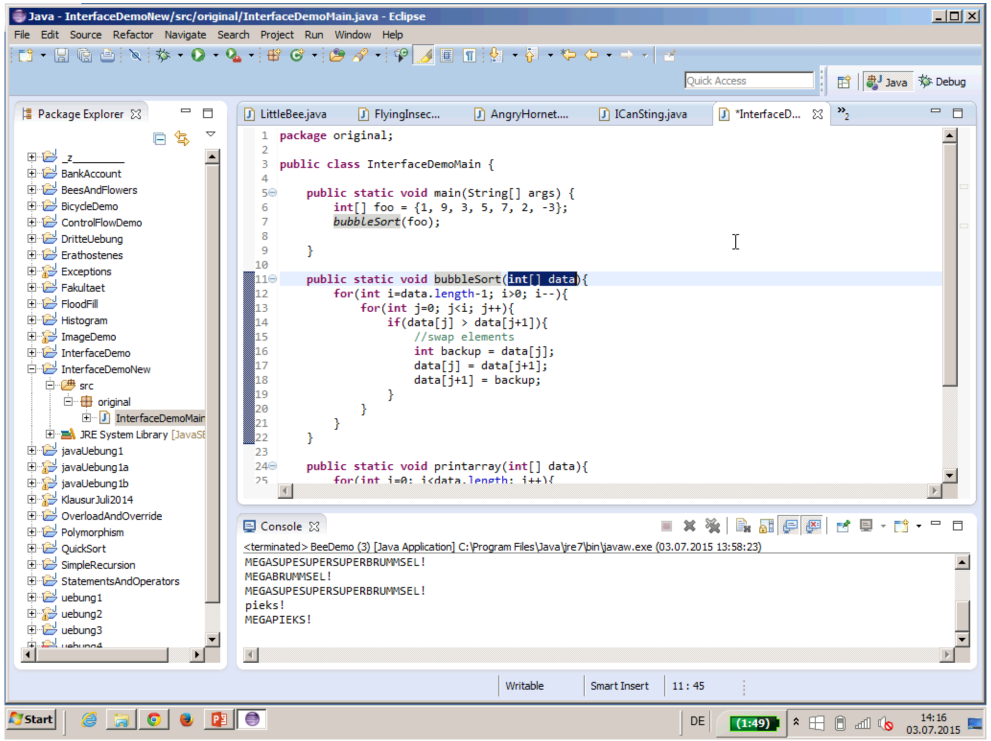
```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};


    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printarray(int[] data){
        for(int i=0; i<data.length; i++){
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



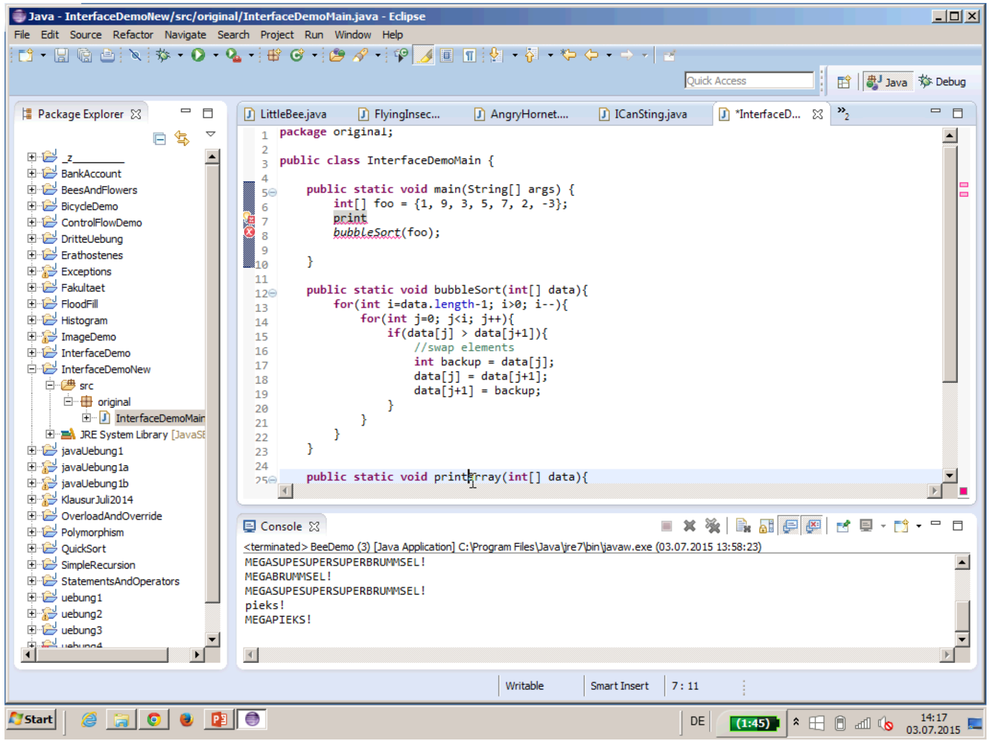
```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printarray(int[] data){
        for(int i=0; i<data.length; i++){
```

```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



```java
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printarray(int[] data){
        for(int i=0; i<data.length; i++){
```

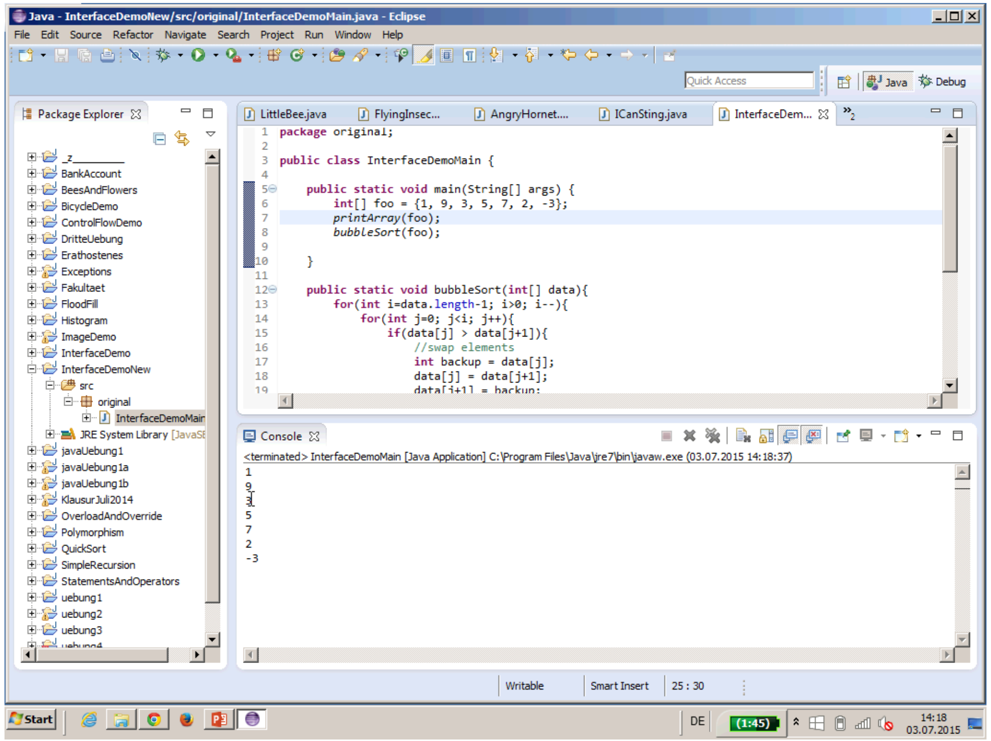
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
*InterfaceD...
Writable
Smart Insert
7 : 11
14:17
03.07.2015

```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        print
        bubbleSort(foo);

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(int[] data){
```

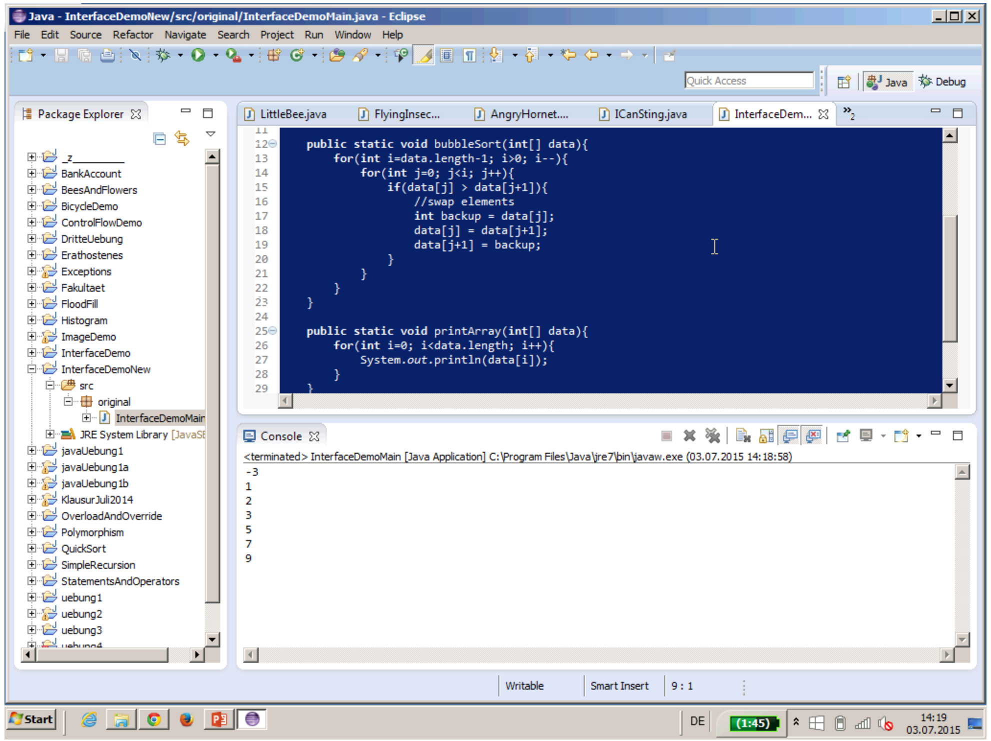
```
<terminated> BeeDemo (3) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 13:58:23)
MEGASUPESUPERSUPERBRUMMSEL!
MEGABRUMMSEL!
MEGASUPESUPERSUPERBRUMMSEL!
pieks!
MEGAPIEKS!
```



Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
InterfaceDem...
Writable
Smart Insert
25 : 30
14:18
03.07.2015

```
package original;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        printArray(foo);
        bubbleSort(foo);

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
```

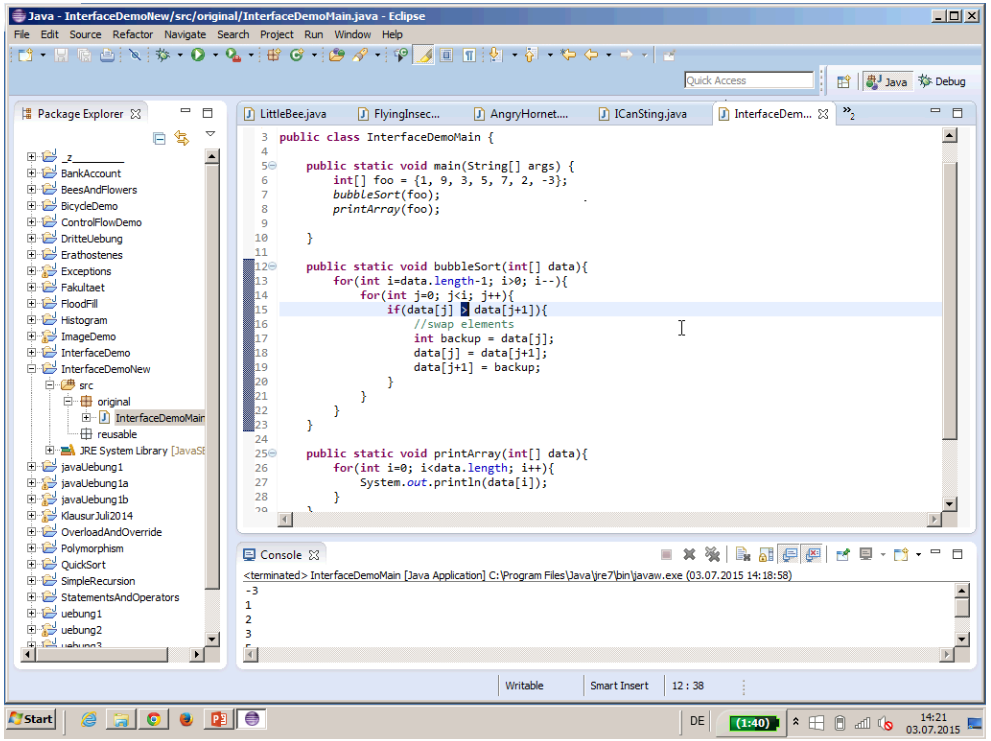
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:37)
1
9
3
5
7
2
-3
```



Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
InterfaceDem...
Writable
Smart Insert
9 : 1
14:19
03.07.2015

```
    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
        }
    }
```

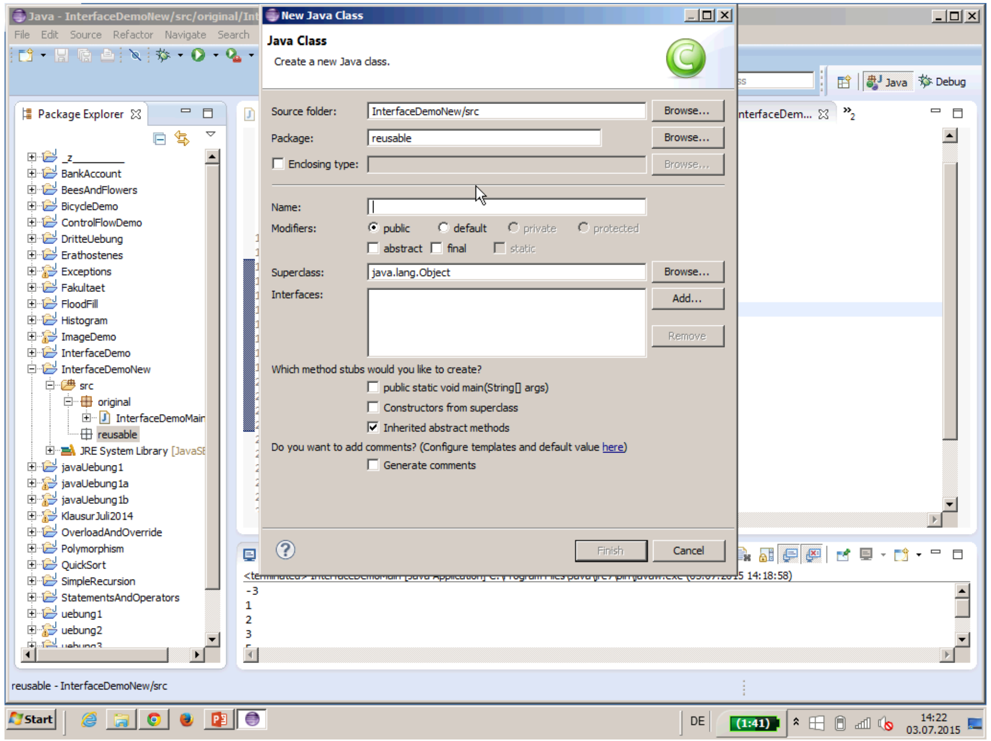
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
5
7
9
```



Java - InterfaceDemoNew/src/original/InterfaceDemoMain.java - Eclipse
InterfaceDem...
Writable
Smart Insert
12 : 38
14:21
03.07.2015

```
public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
        }
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



New Java Class

Java Class

Create a new Java class.

Source folder: InterfaceDemoNew/src

Package: reusable

□ Enclosing type:

Name:

Modifiers: ● public ○ default ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

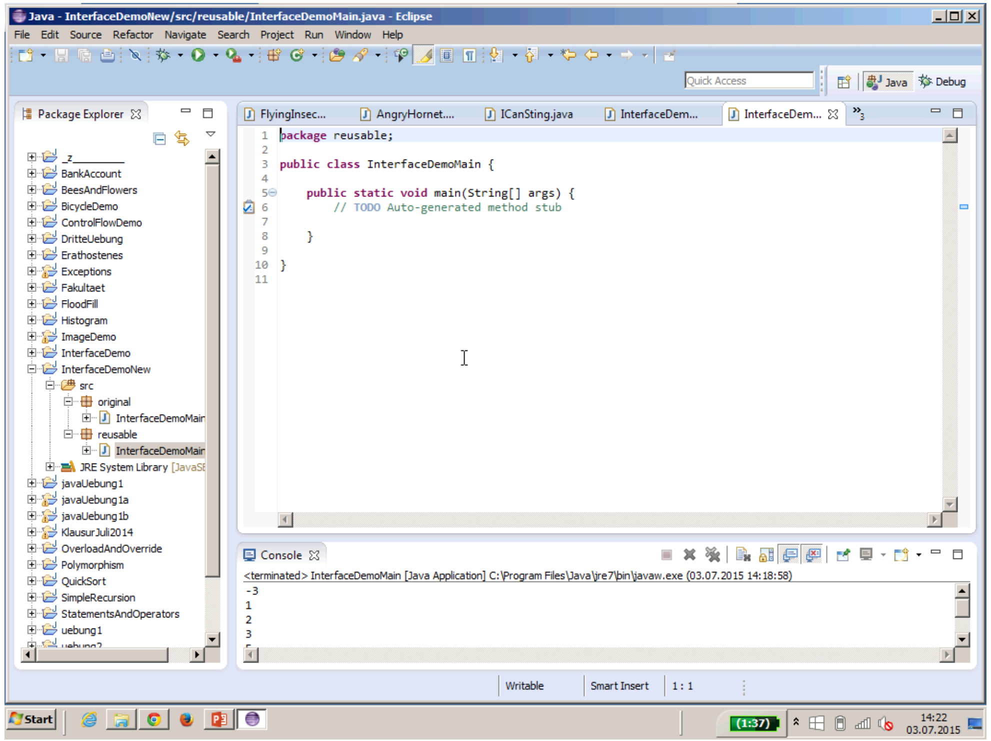
□ public static void main(String[] args)

□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

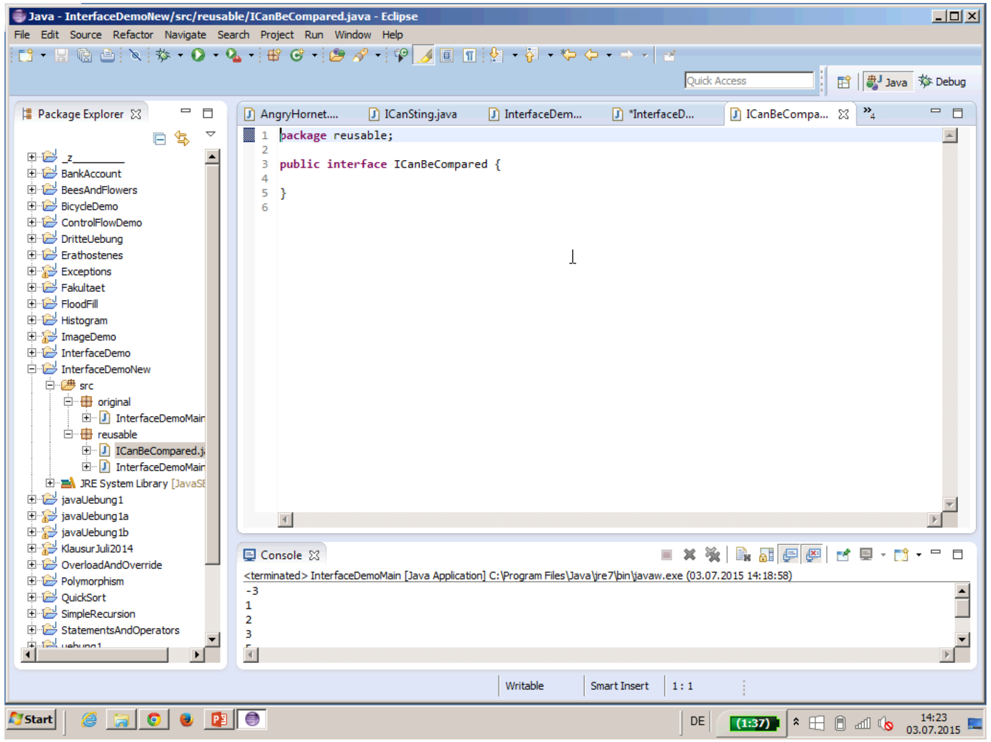
□ Generate comments



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}
```

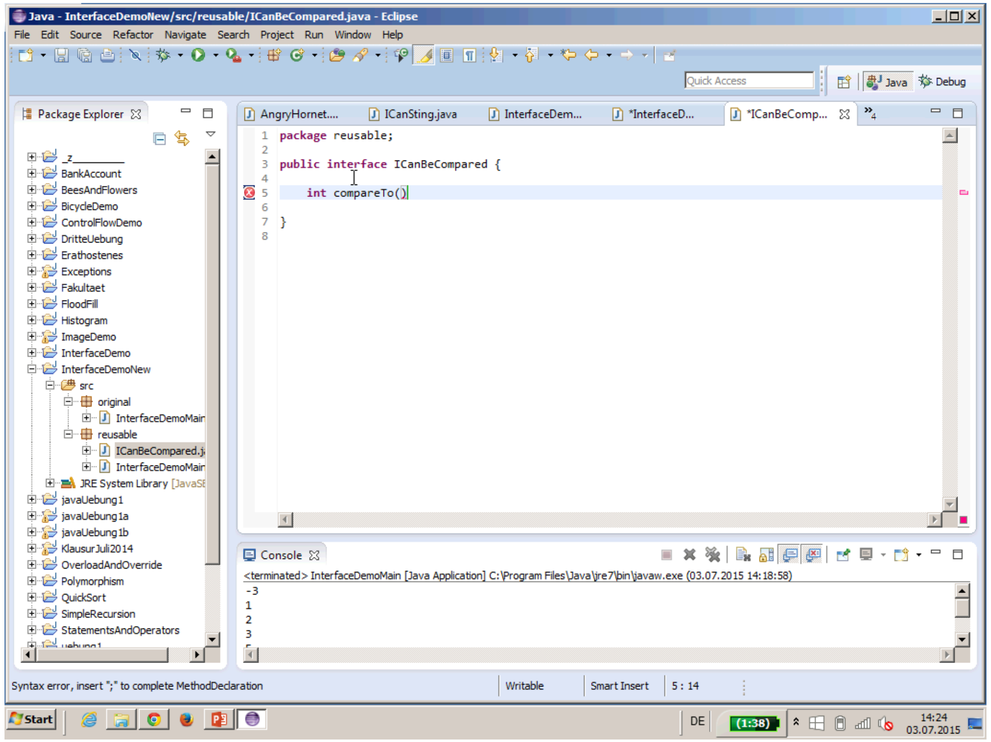
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```
package reusable;

public interface ICanBeCompared {

}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

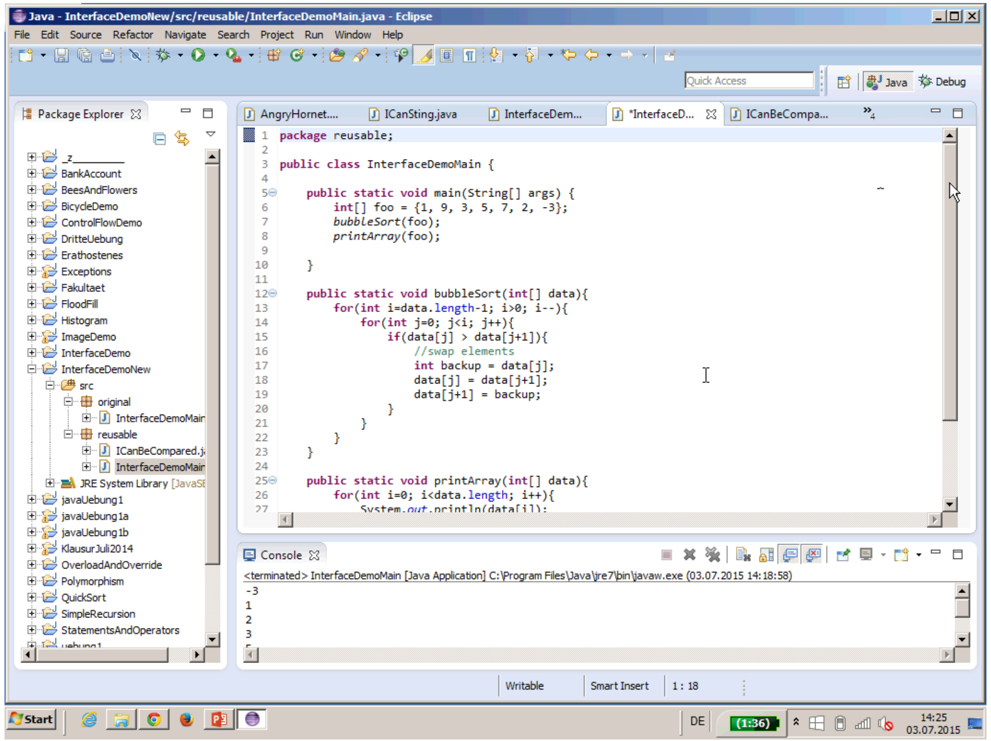


Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```
package reusable;

public interface ICanBeCompared {

    int compareTo()

}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Syntax error, insert ";" to complete MethodDeclaration

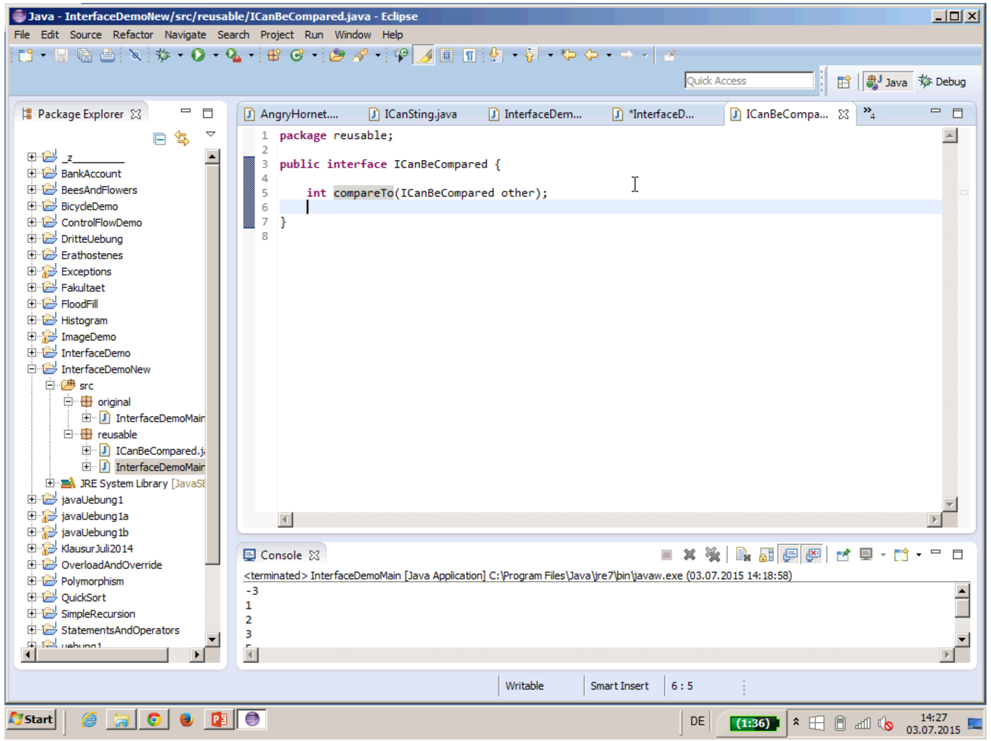


Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(int[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 1 : 18

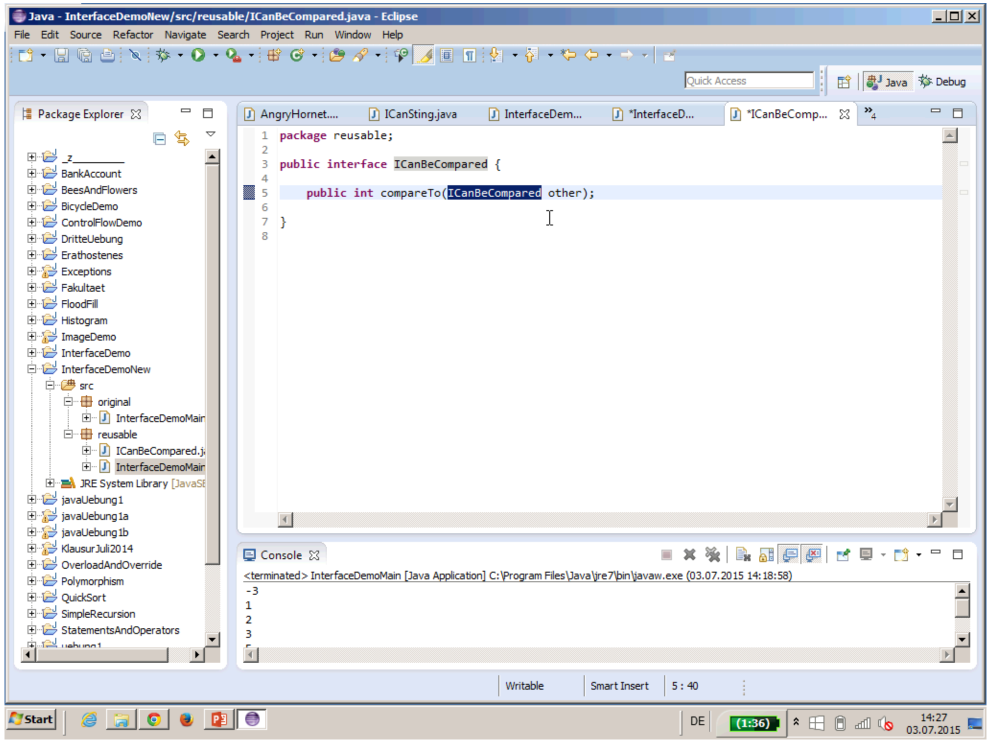


Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```java
package reusable;

public interface ICanBeCompared {

    int compareTo(ICanBeCompared other);

}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 6 : 5

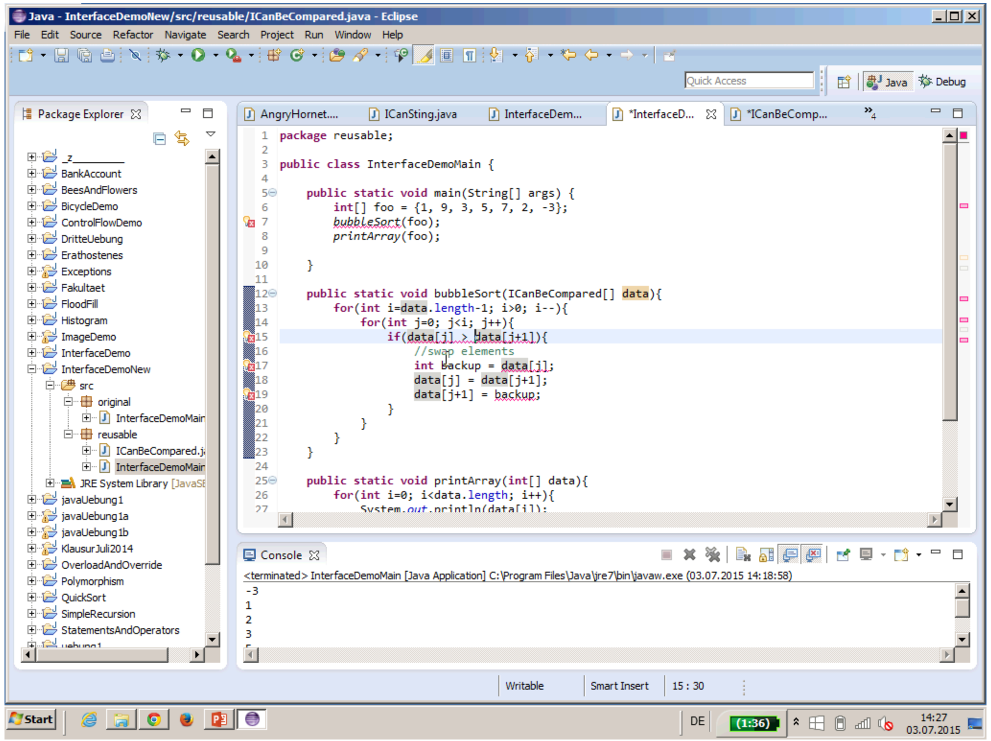


Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```java
package reusable;

public interface ICanBeCompared {

    public int compareTo(ICanBeCompared other);

}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable Smart Insert 5 : 40

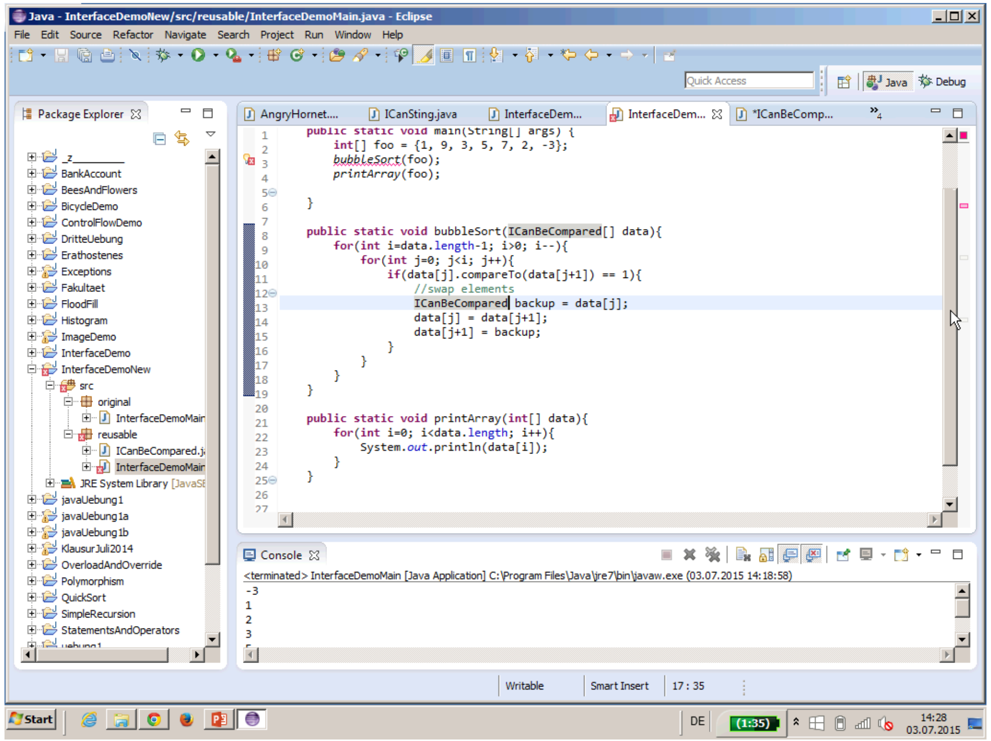


Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j] > data[j+1]){
                    //swap elements
                    int backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

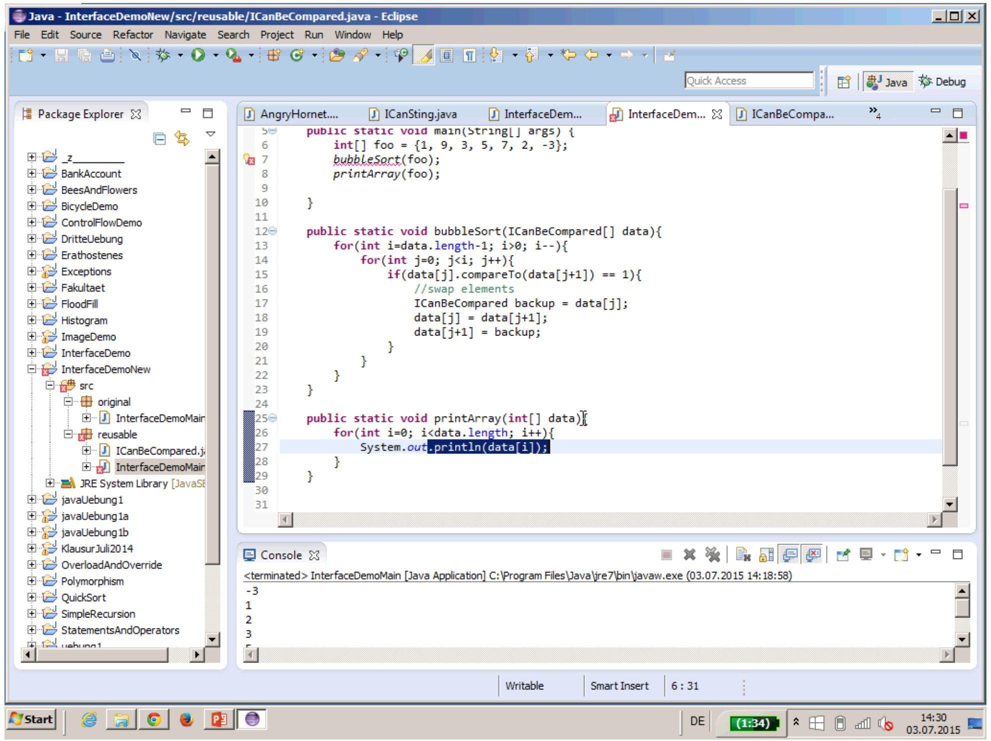
Writable Smart Insert 15 : 30



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```java
    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
        }
    }
```

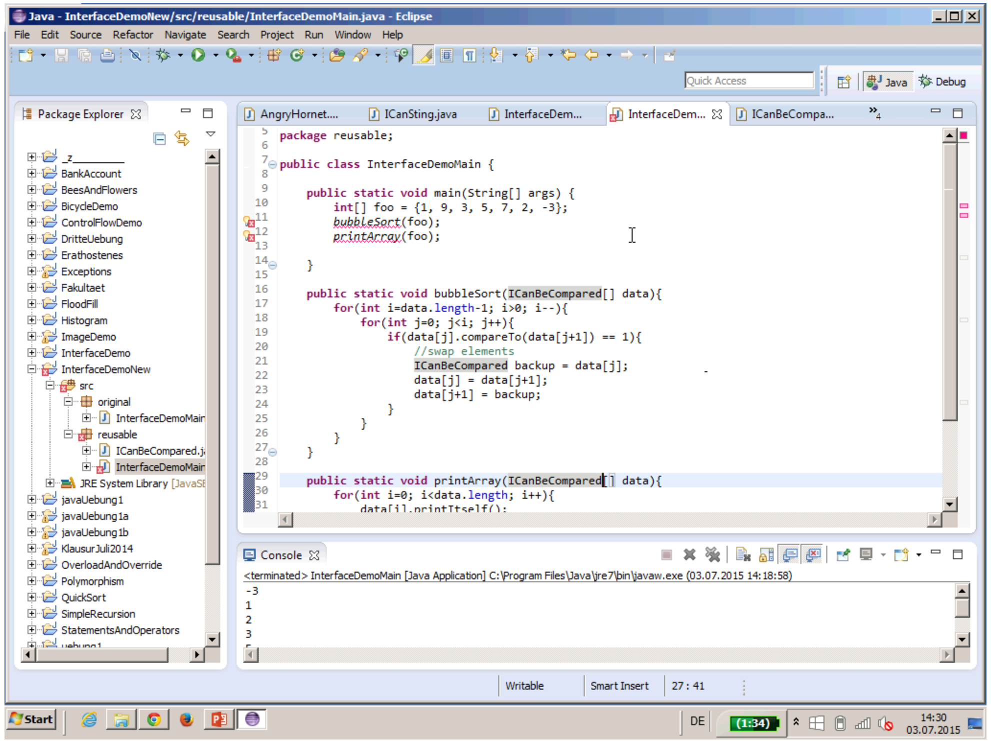
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/ICanBeCompared.java - Eclipse

```java
    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(int[] data){
        for(int i=0; i<data.length; i++){
            System.out.println(data[i]);
        }
    }
```

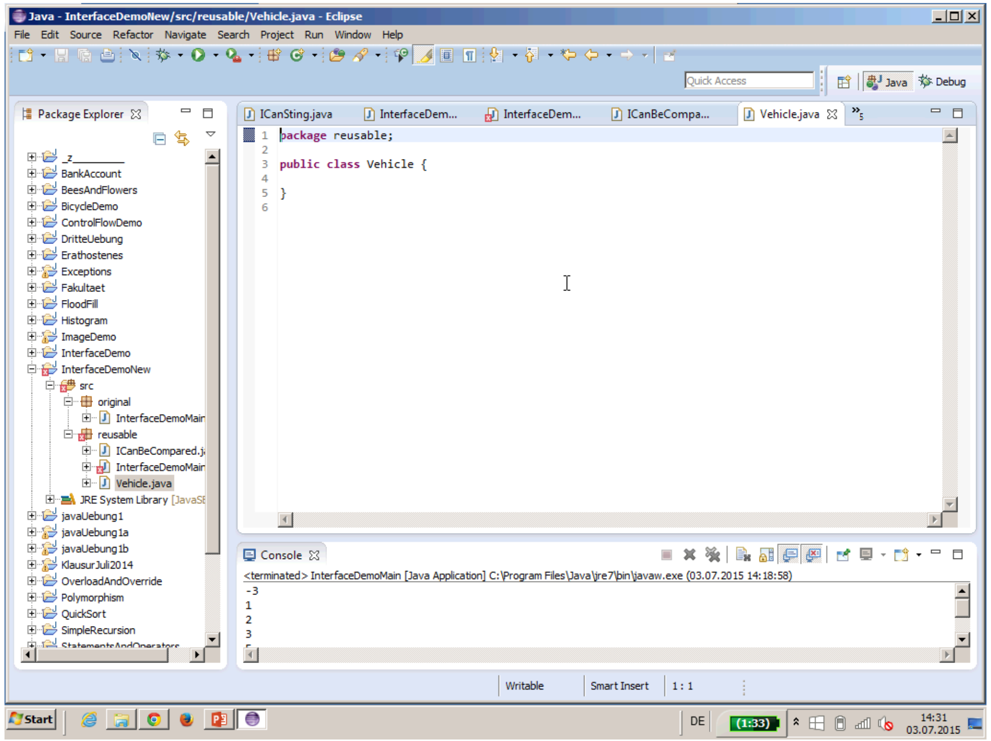
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        int[] foo = {1, 9, 3, 5, 7, 2, -3};
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(ICanBeCompared[] data){
        for(int i=0; i<data.length; i++){
            data[i].printItself();
```

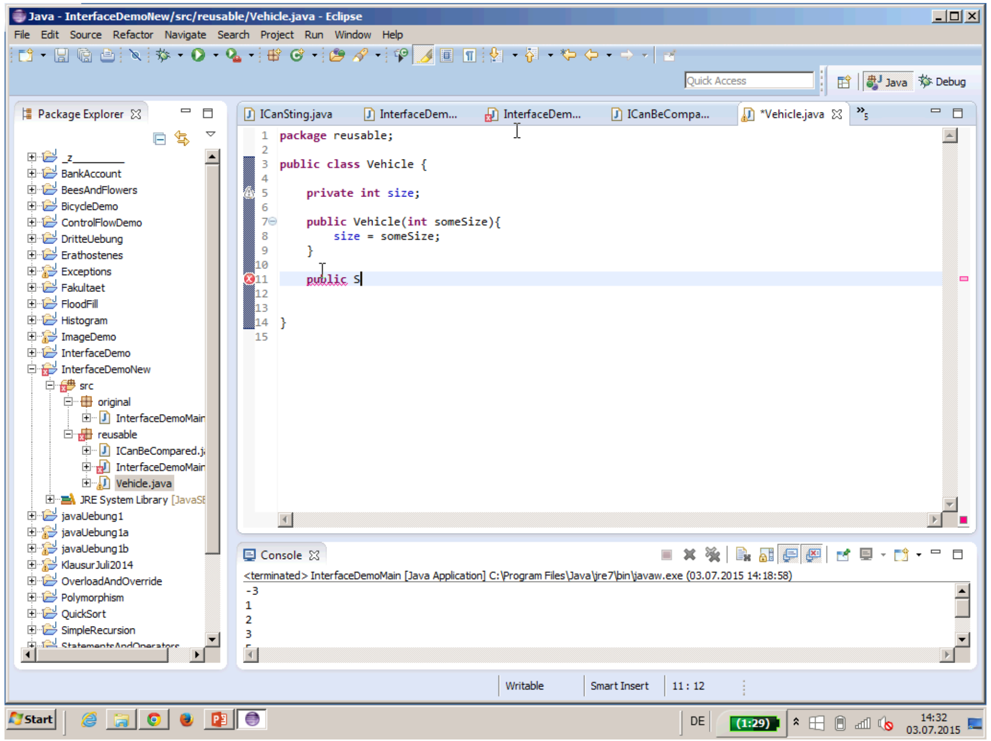
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```java
package reusable;

public class Vehicle {

}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

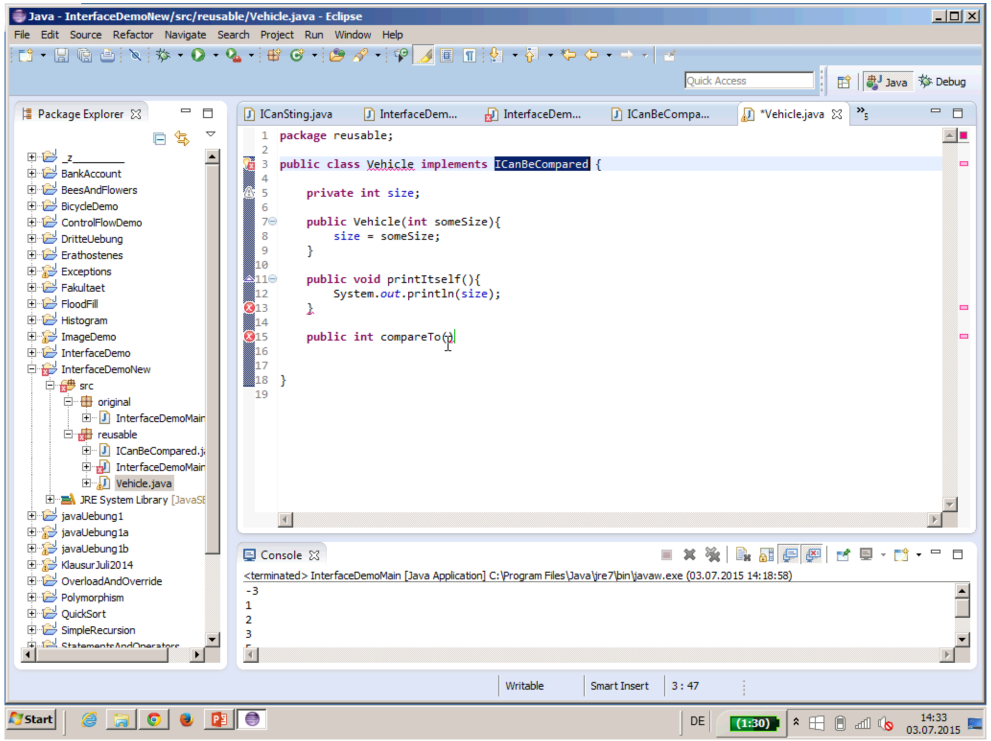


Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```java
package reusable;

public class Vehicle {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public S

}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 11 : 12

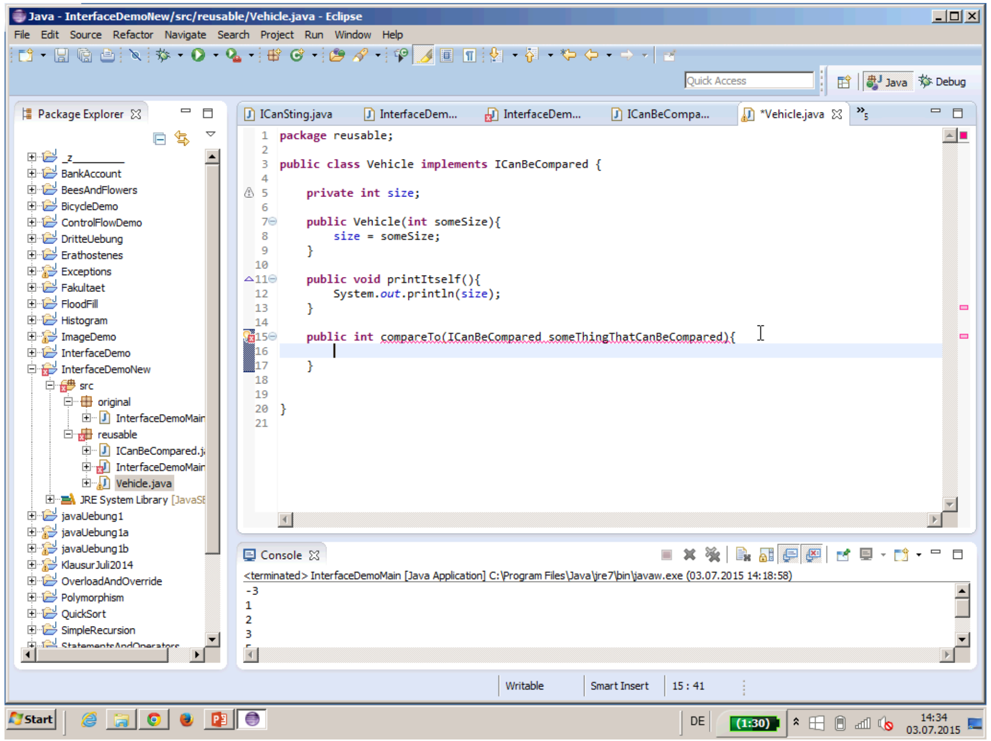


Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo()


}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 3 : 47

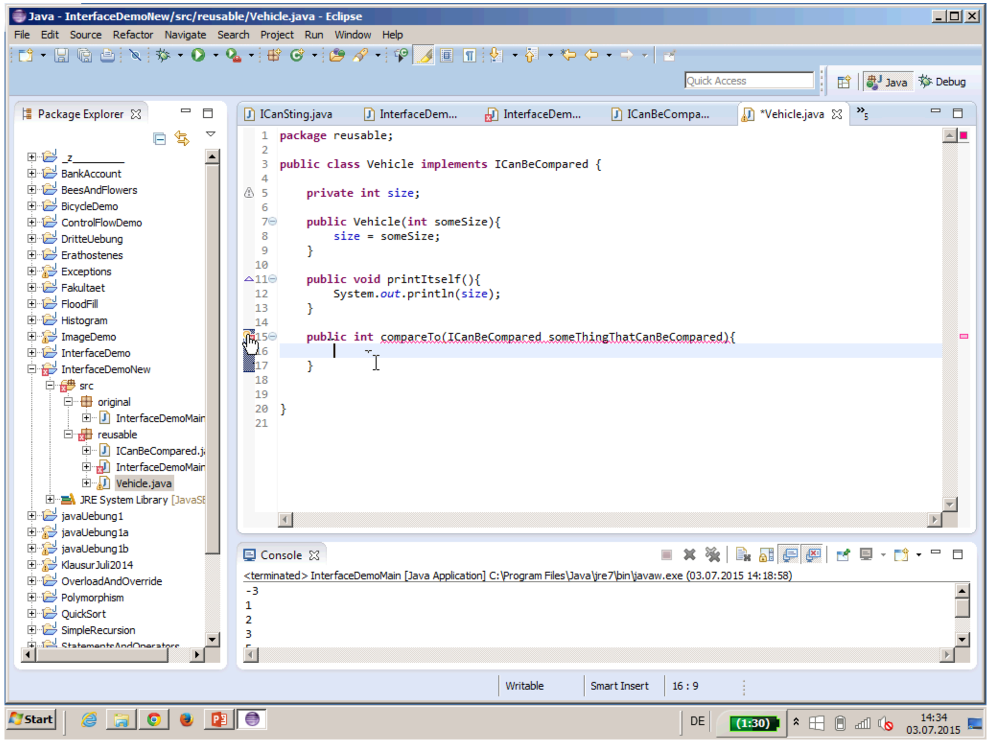


Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        
    }



}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 15 : 41

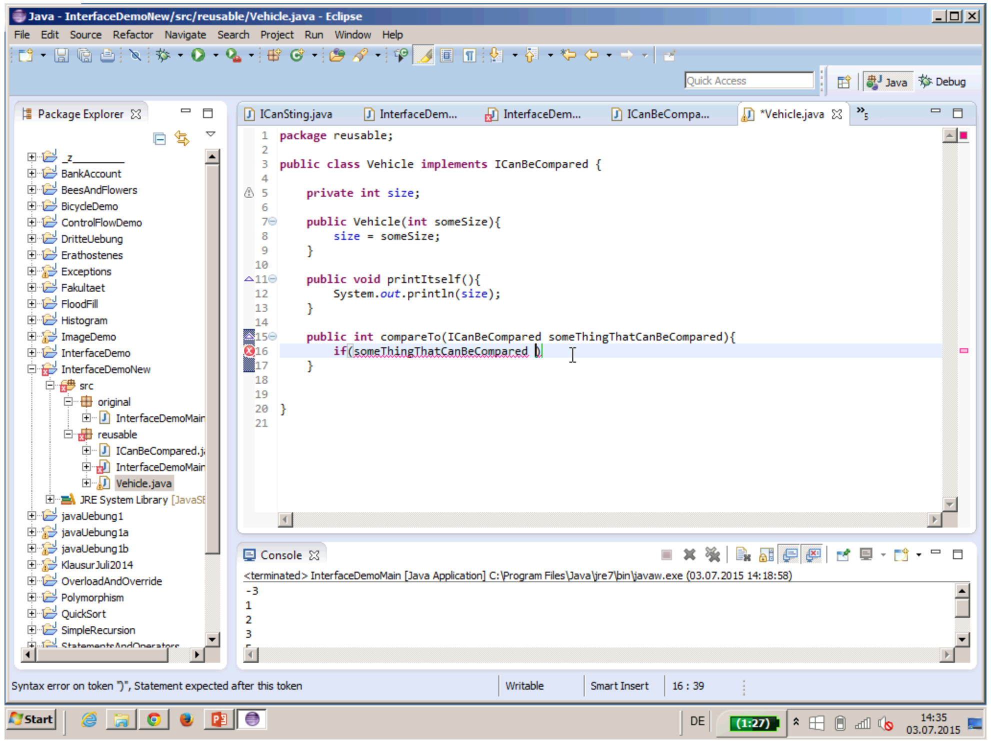


Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        
    }



}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 16 : 9



Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

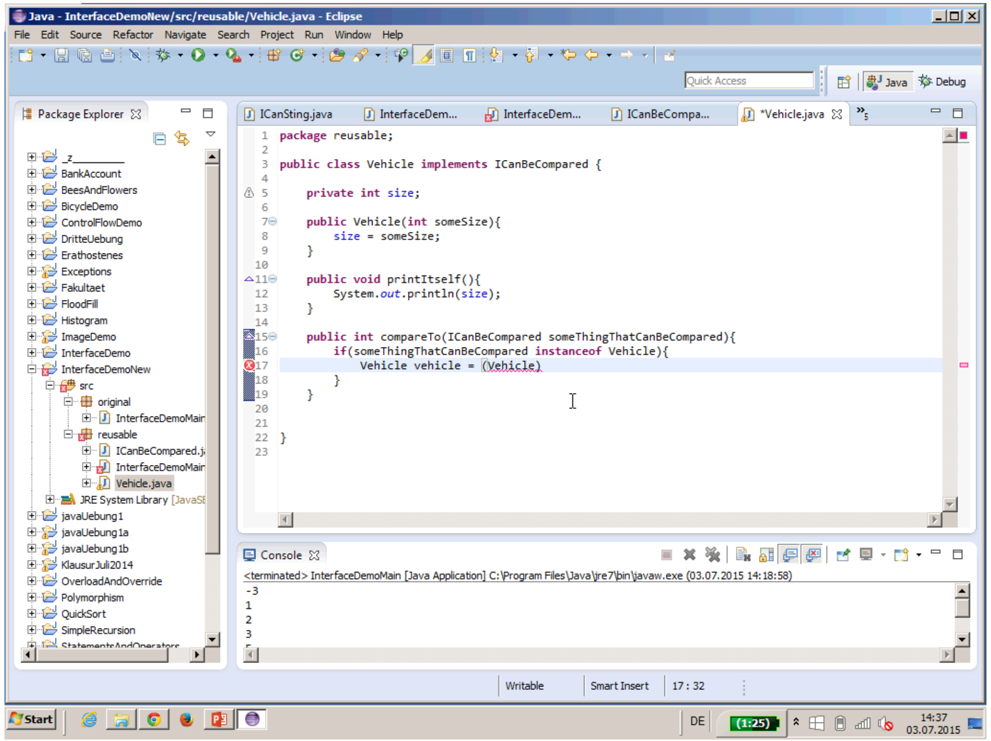
*Vehicle.java

```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared )
    }


}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Syntax error on token ")", Statement expected after this token | Writable | Smart Insert | 16 : 39



Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

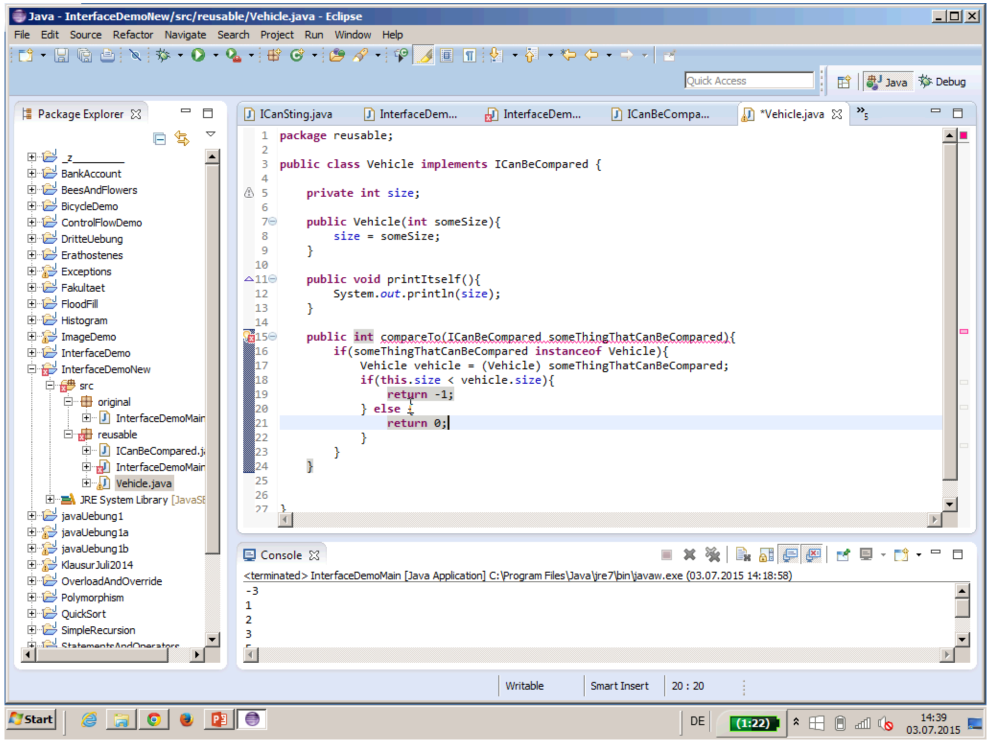
*Vehicle.java

```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle)
        }
    }


}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 17 : 32



Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

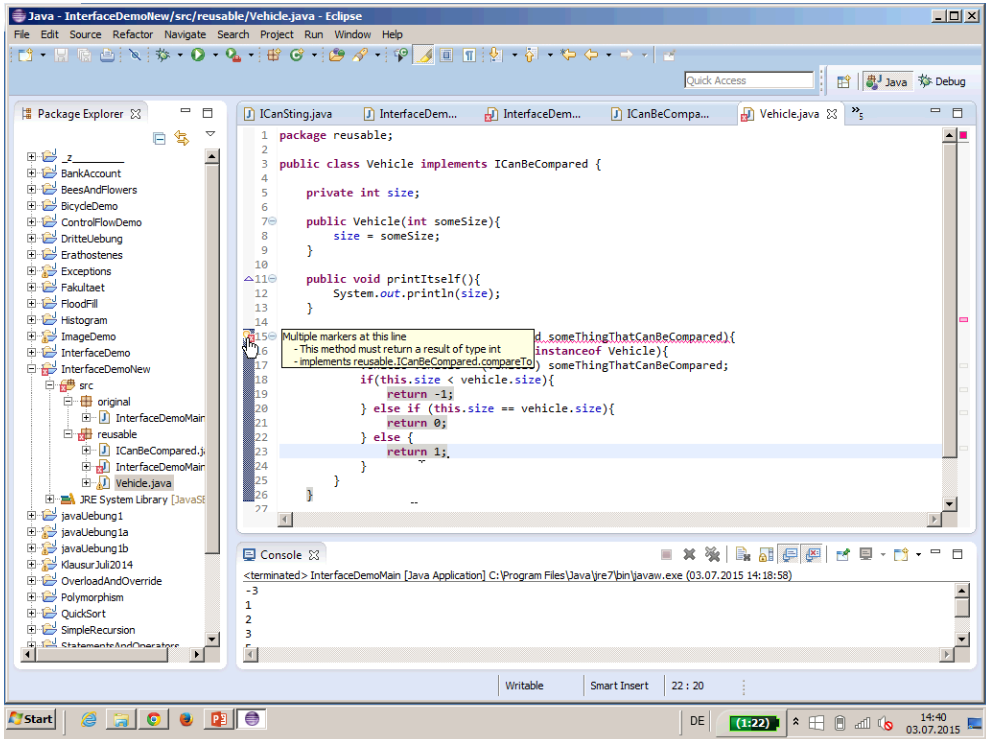
*Vehicle.java

```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else {
                return 0;
            }
        }
    }


}
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 20 : 20



Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

Vehicle.java

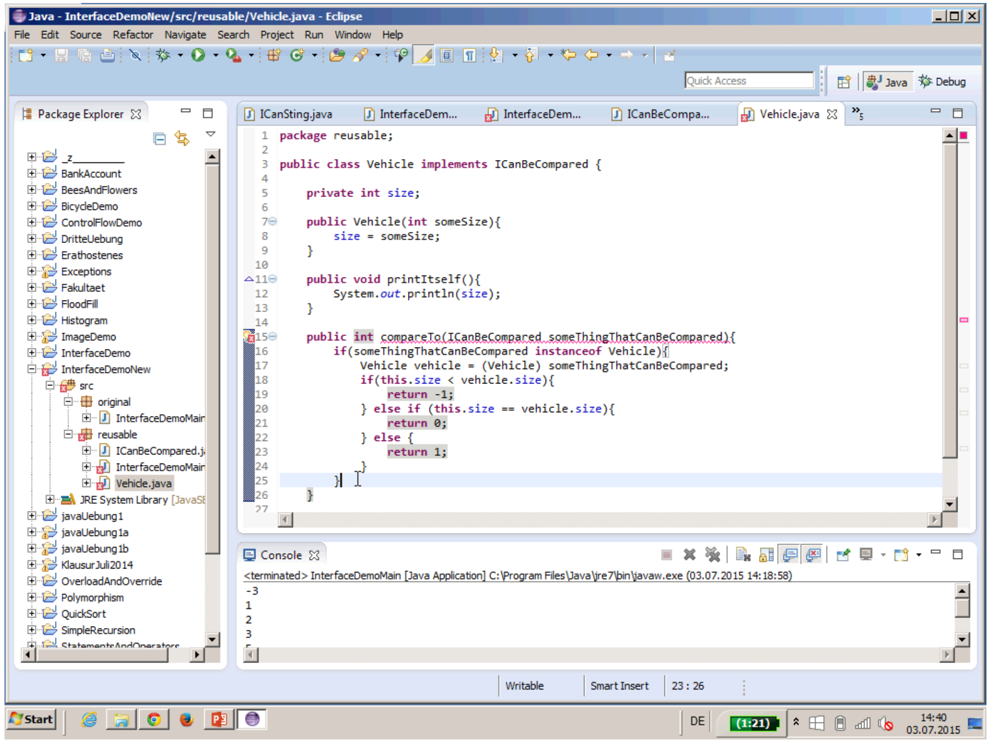
```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

                                   d someThingThatCanBeCompared){
                                   instanceof Vehicle){
                                   ) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        }
    }
```

Multiple markers at this line
- This method must return a result of type int
- implements reusable.ICanBeCompared.compareTo

Console

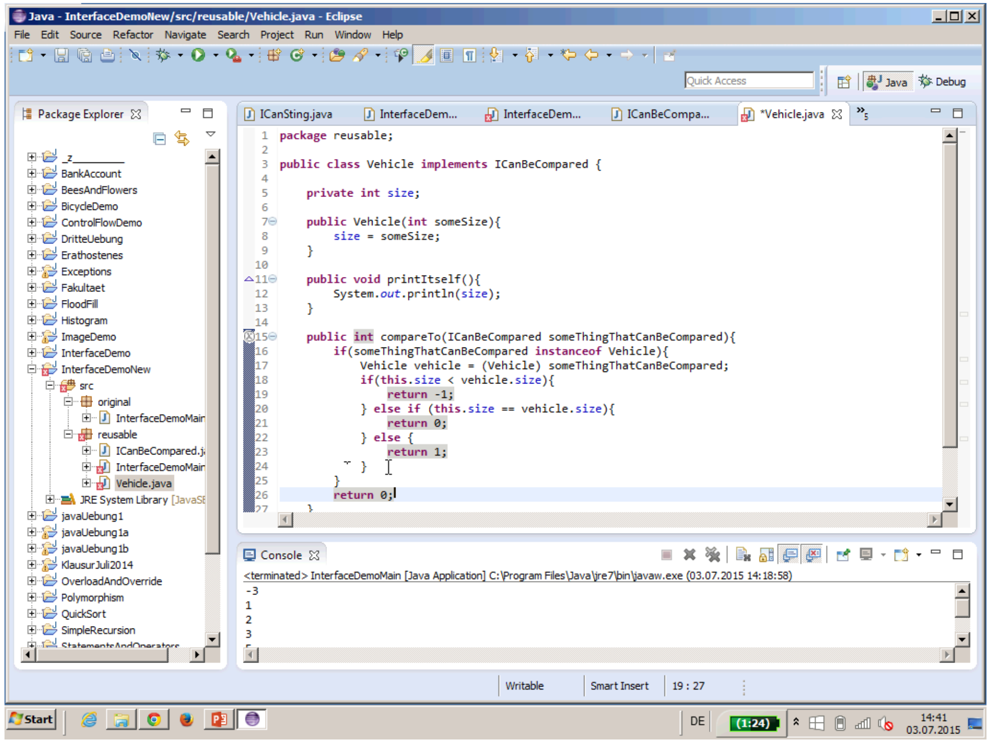
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 22 : 20



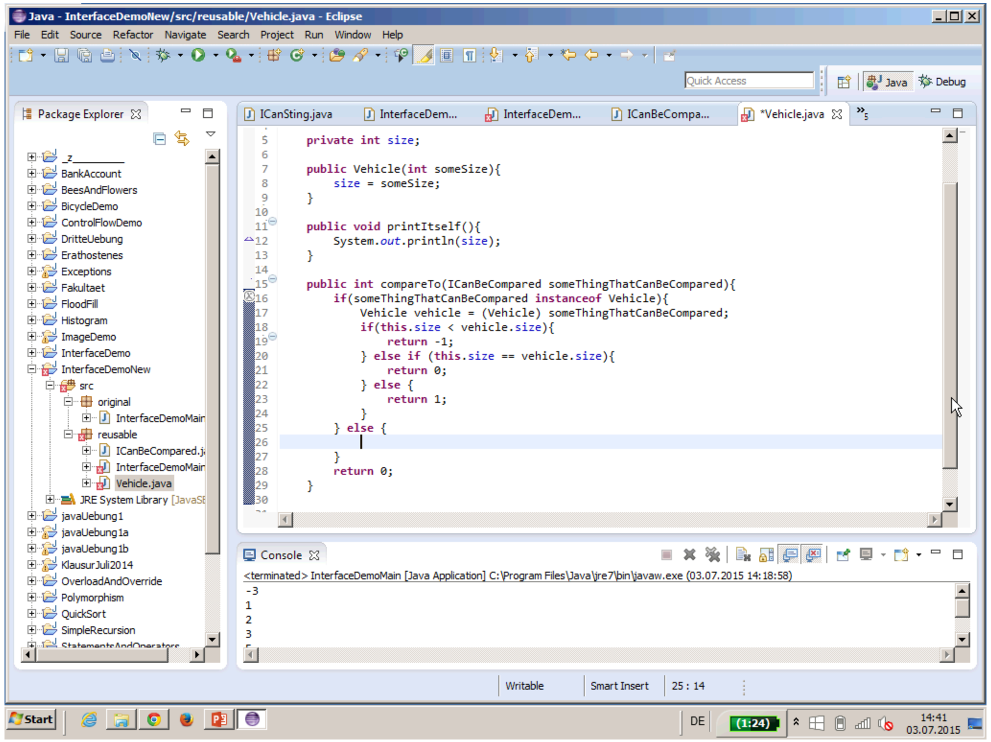
```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        }
    }
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



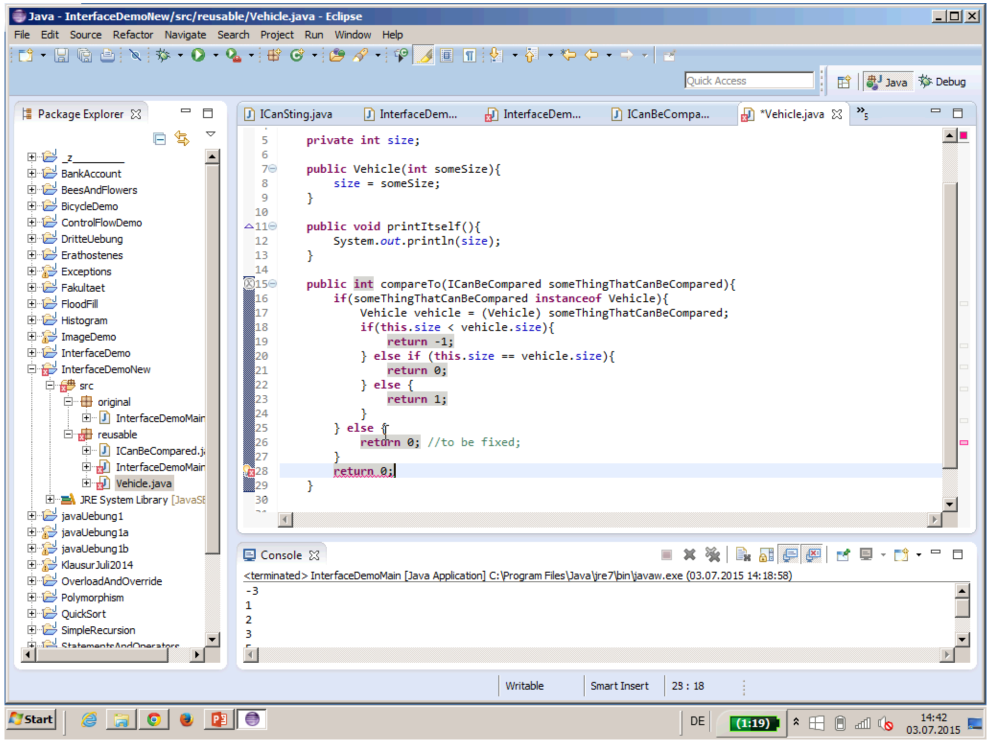
```java
package reusable;

public class Vehicle implements ICanBeCompared {

    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        }
        return 0;
    }
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



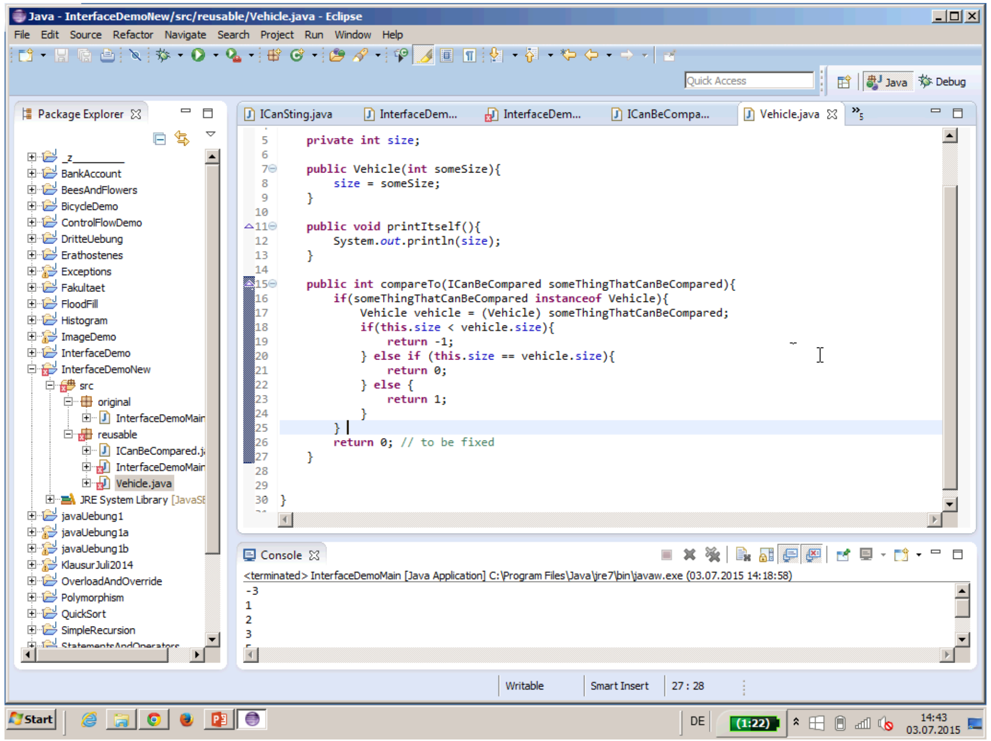
```java
    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        } else {

        }
        return 0;
    }
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



```java
    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        } else {
            return 0; //to be fixed;
        }
        return 0;
    }
```

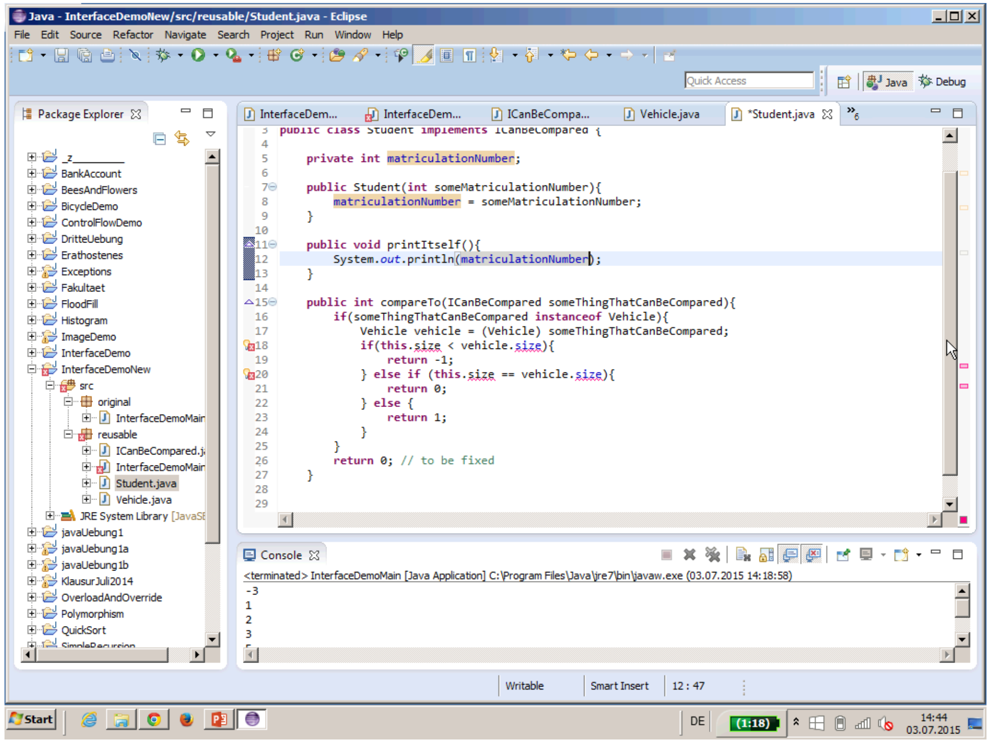
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/Vehicle.java - Eclipse

```java
    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }


}
```

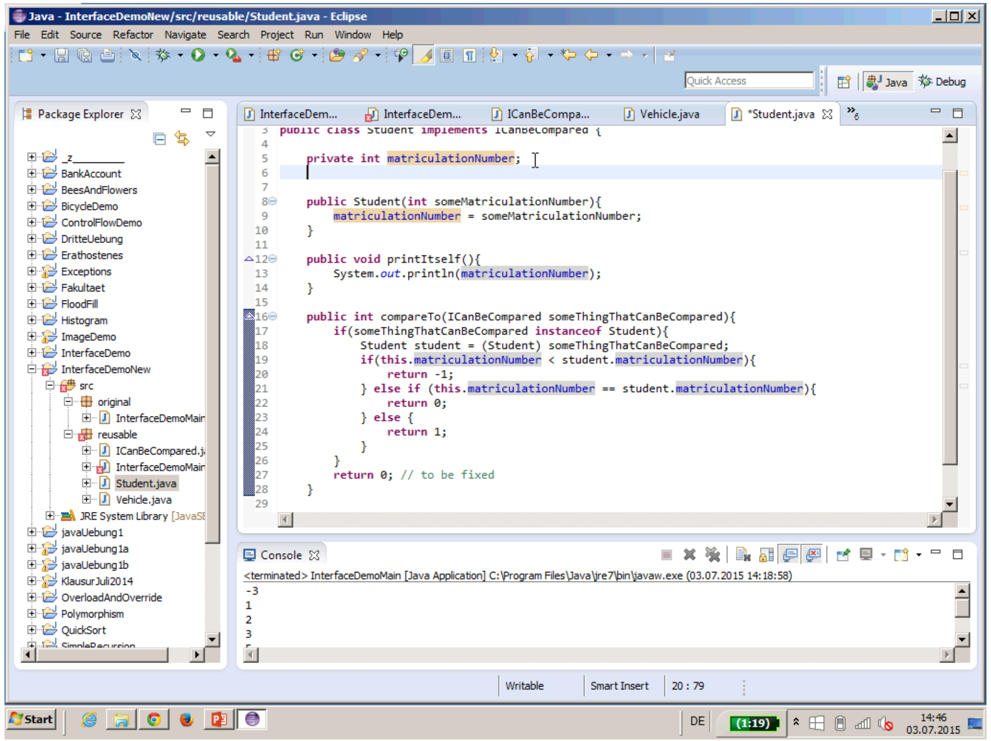
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```java
public class Student implements ICanBeCompared {

    private int matriculationNumber;

    public Student(int someMatriculationNumber){
        matriculationNumber = someMatriculationNumber;
    }

    public void printItself(){
        System.out.println(matriculationNumber);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }

```

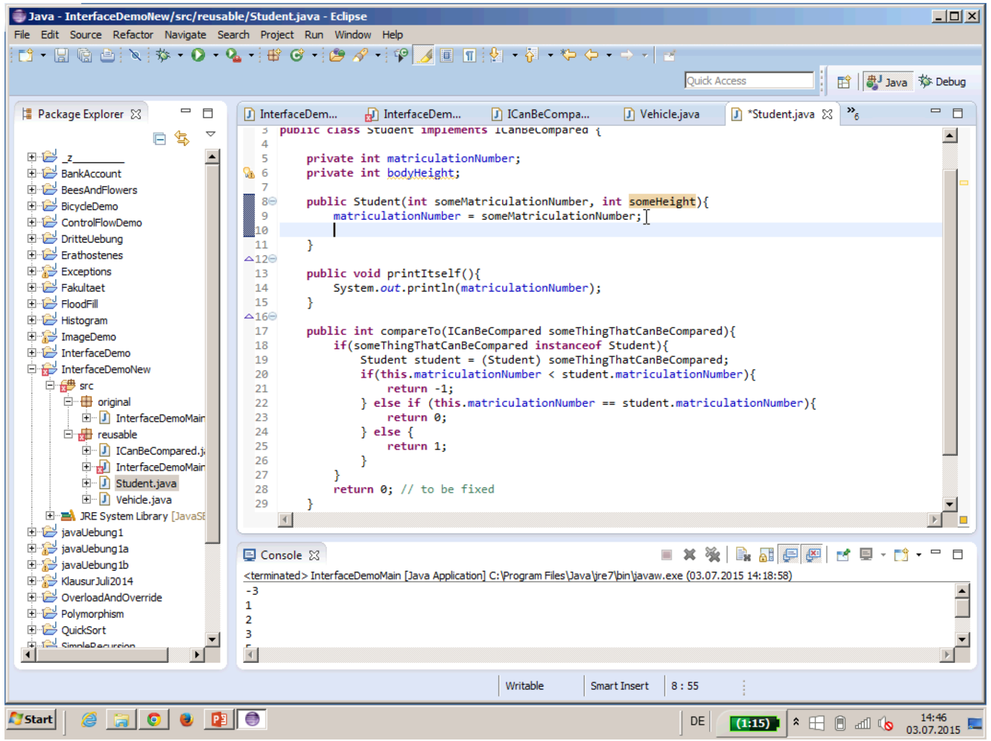
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```java
public class Student implements ICanBeCompared {

    private int matriculationNumber;
    

    public Student(int someMatriculationNumber){
        matriculationNumber = someMatriculationNumber;
    }

    public void printItself(){
        System.out.println(matriculationNumber);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Student){
            Student student = (Student) someThingThatCanBeCompared;
            if(this.matriculationNumber < student.matriculationNumber){
                return -1;
            } else if (this.matriculationNumber == student.matriculationNumber){
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }

```

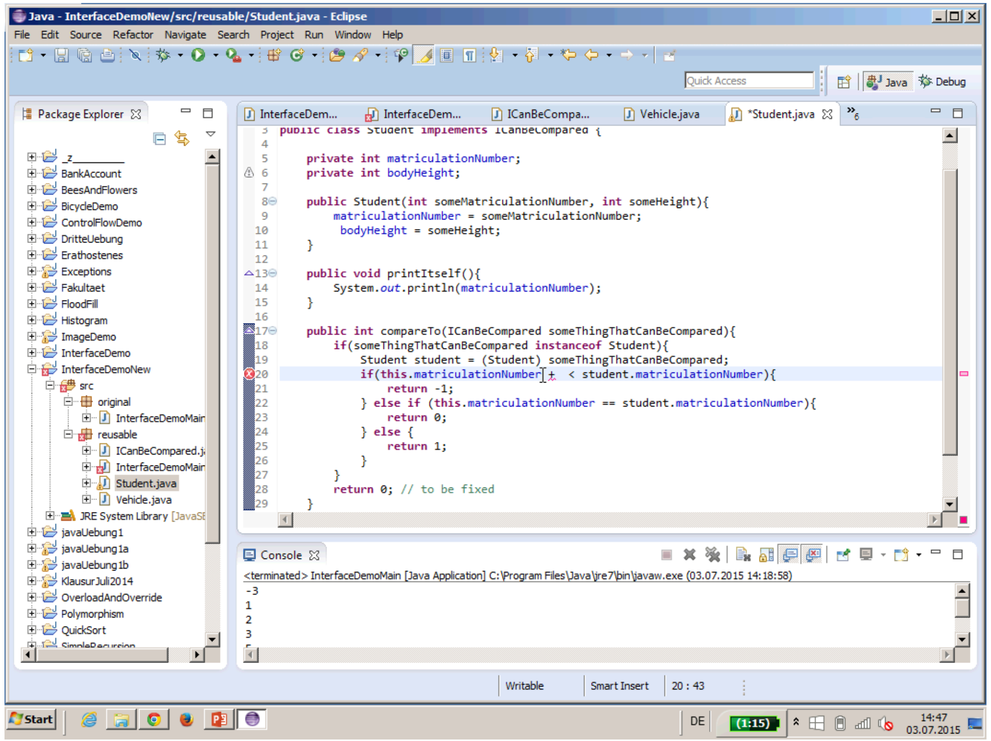
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```java
public class Student implements ICanBeCompared {

    private int matriculationNumber;
    private int bodyHeight;

    public Student(int someMatriculationNumber, int someHeight){
        matriculationNumber = someMatriculationNumber;
        
    }


    public void printItself(){
        System.out.println(matriculationNumber);
    }


    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Student){
            Student student = (Student) someThingThatCanBeCompared;
            if(this.matriculationNumber < student.matriculationNumber){
                return -1;
            } else if (this.matriculationNumber == student.matriculationNumber){
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }
```

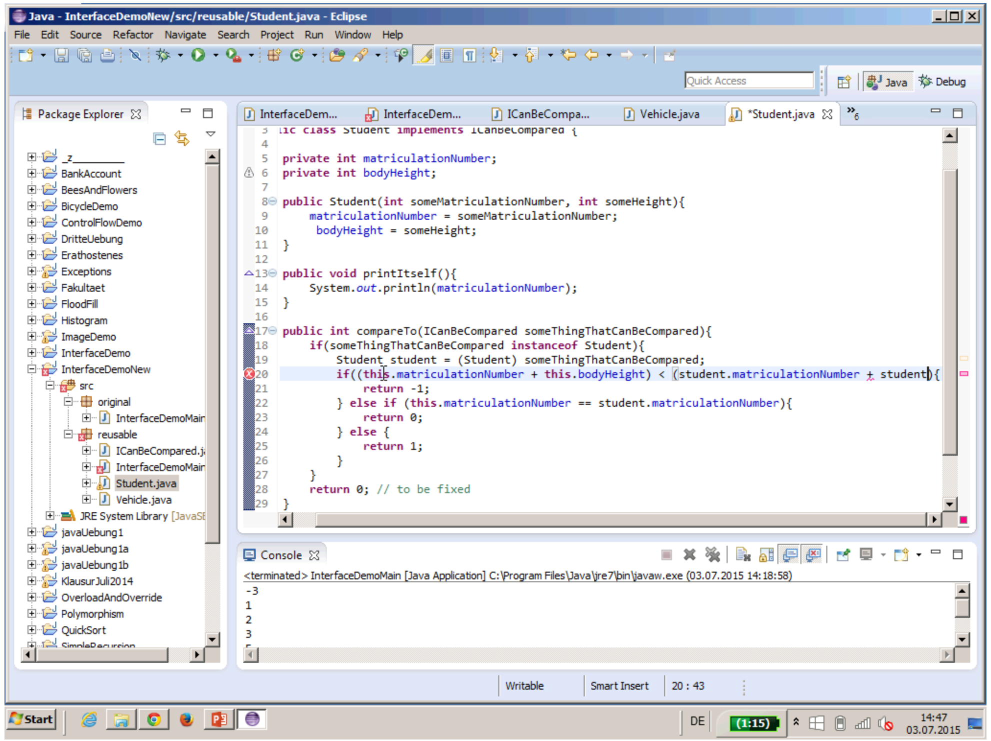
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



```java
public class Student implements ICanBeCompared {

    private int matriculationNumber;
    private int bodyHeight;

    public Student(int someMatriculationNumber, int someHeight){
        matriculationNumber = someMatriculationNumber;
         bodyHeight = someHeight;
    }

    public void printItself(){
        System.out.println(matriculationNumber);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Student){
            Student student = (Student) someThingThatCanBeCompared;
            if(this.matriculationNumber +  < student.matriculationNumber){
                return -1;
            } else if (this.matriculationNumber == student.matriculationNumber){
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }
```

Console

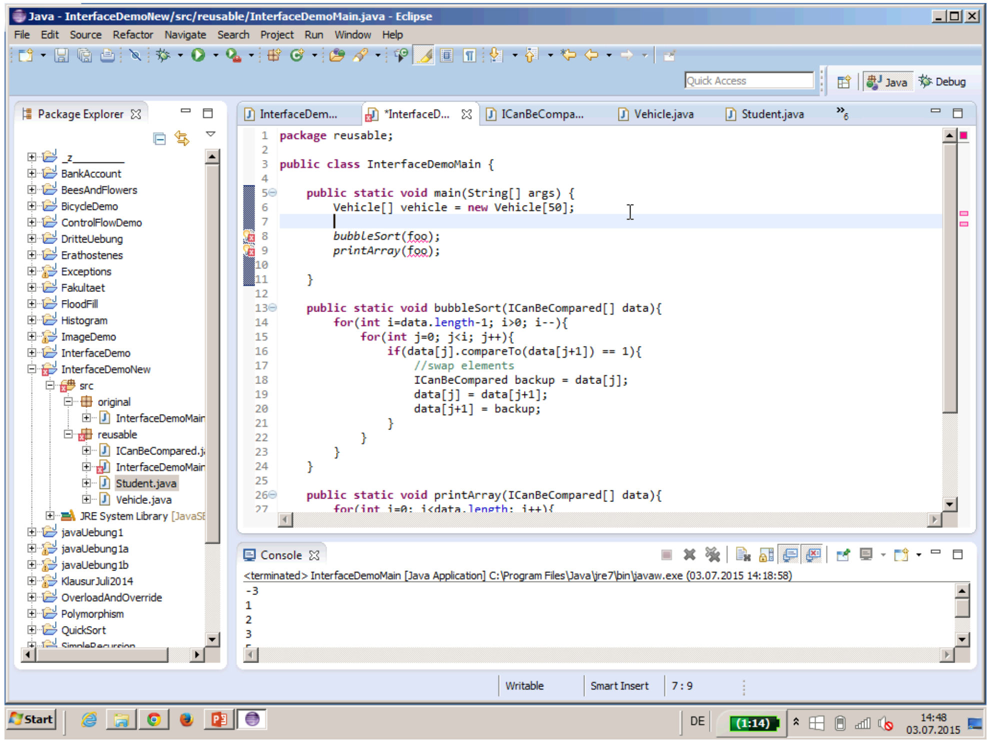
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



```java
lic class Student implements ICanBeCompared {

private int matriculationNumber;
private int bodyHeight;

public Student(int someMatriculationNumber, int someHeight){
    matriculationNumber = someMatriculationNumber;
     bodyHeight = someHeight;
}

public void printItself(){
    System.out.println(matriculationNumber);
}

public int compareTo(ICanBeCompared someThingThatCanBeCompared){
    if(someThingThatCanBeCompared instanceof Student){
        Student student = (Student) someThingThatCanBeCompared;
        if((this.matriculationNumber + this.bodyHeight) < (student.matriculationNumber + student){
            return -1;
        } else if (this.matriculationNumber == student.matriculationNumber){
            return 0;
        } else {
            return 1;
        }
    }
    return 0; // to be fixed
}
```

Console

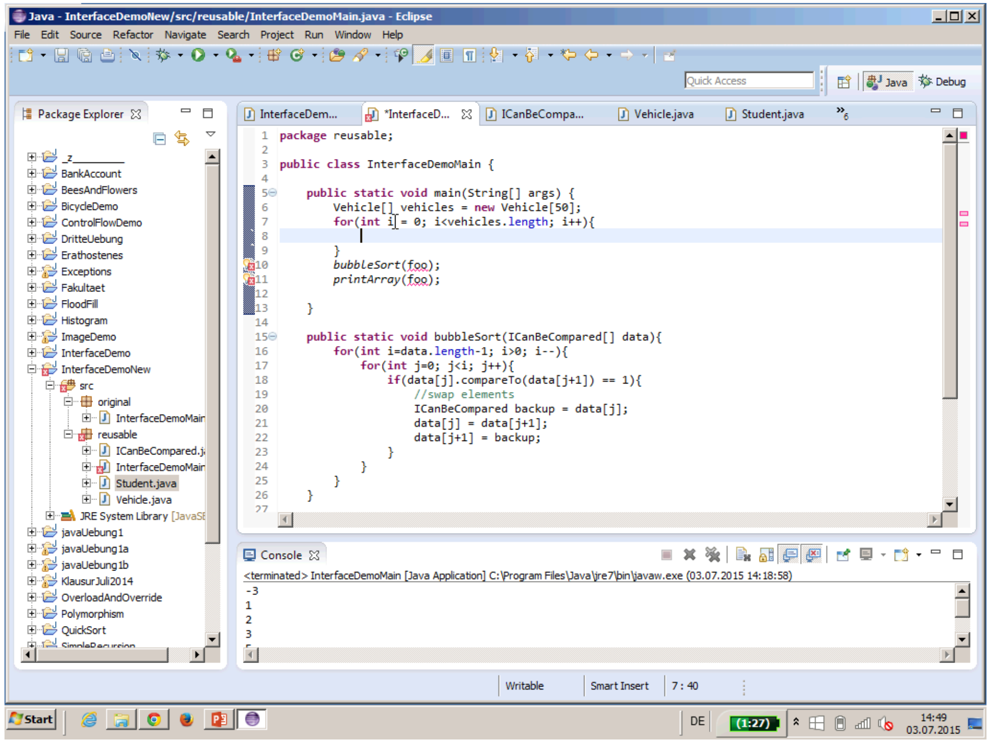
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicle = new Vehicle[50];

        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }

    public static void printArray(ICanBeCompared[] data){
        for(int i=0; i<data.length; i++){
```

Console

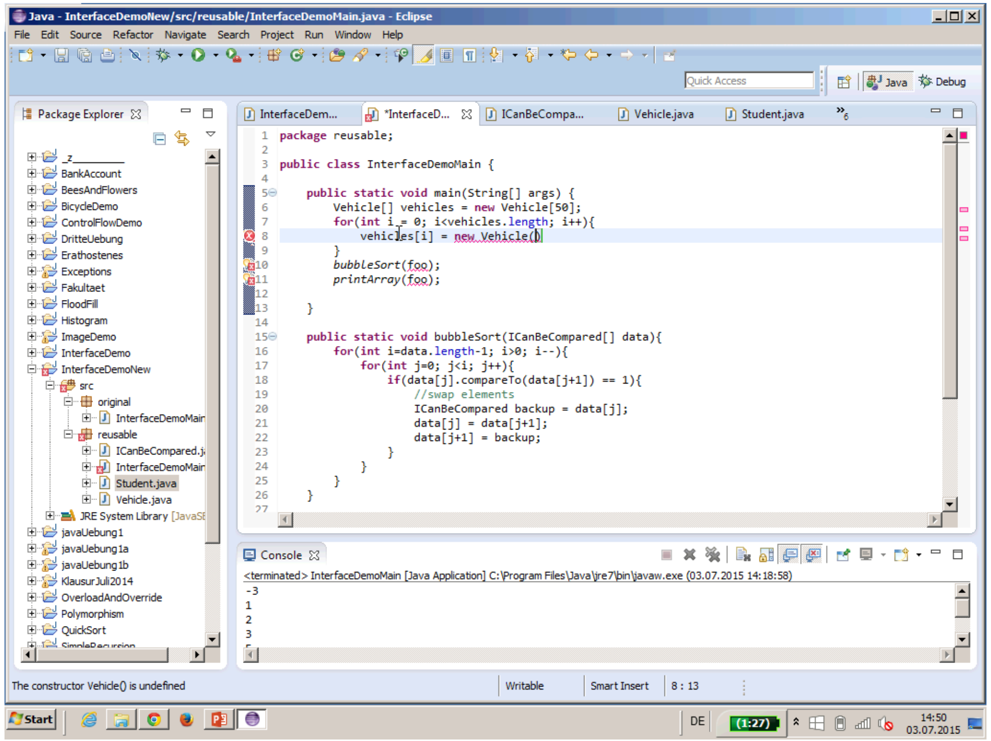
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){

        }
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
```

Console

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

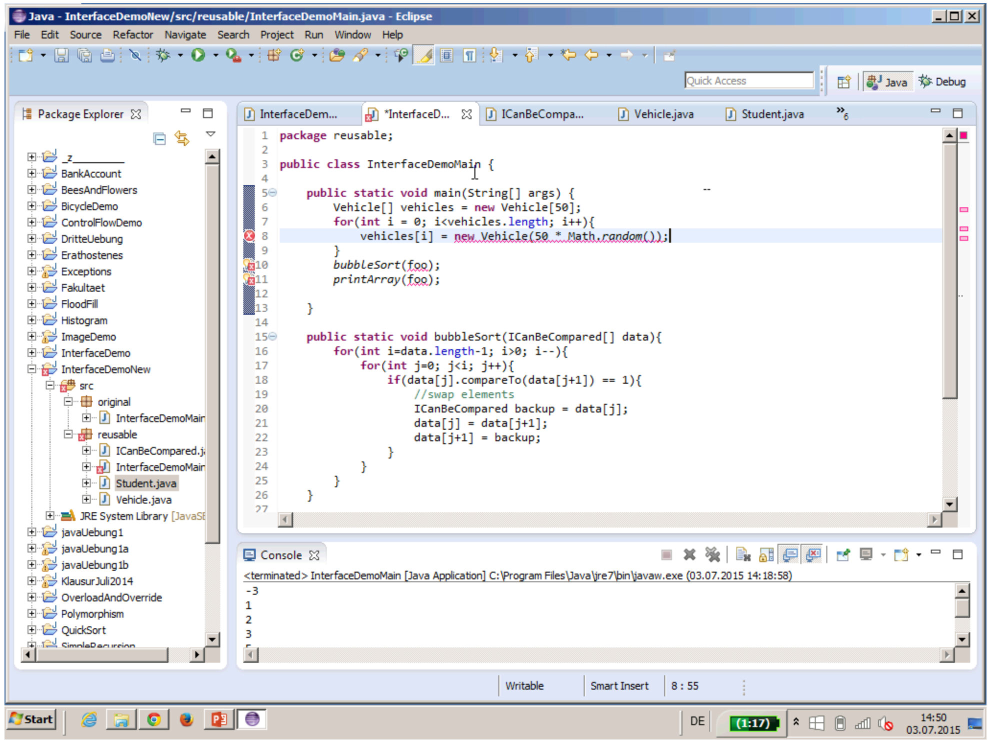


```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle()
        }
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

The constructor Vehicle() is undefined

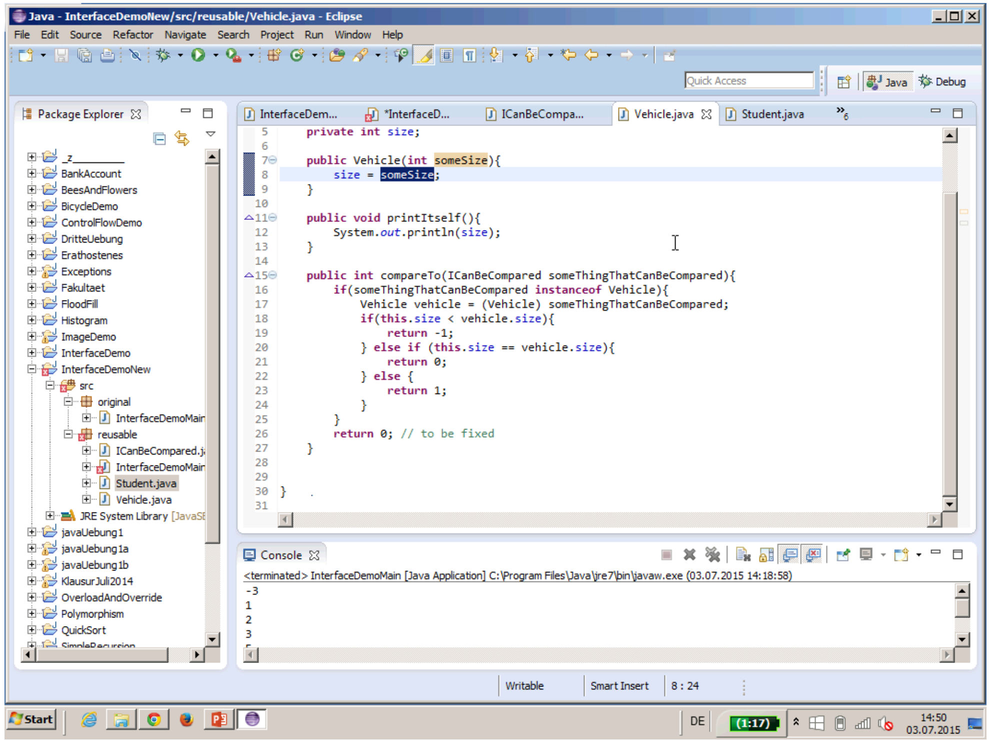
Writable | Smart Insert | 8 : 13



```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle(50 * Math.random());
        }
        bubbleSort(foo);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

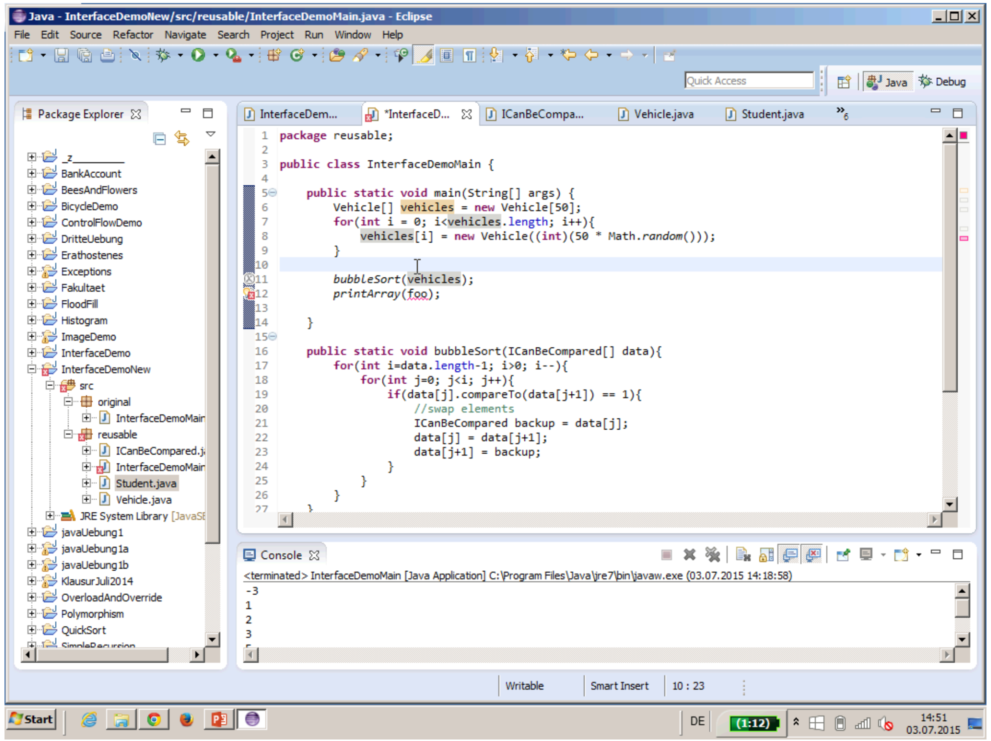
Writable | Smart Insert | 8 : 55



```java
    private int size;

    public Vehicle(int someSize){
        size = someSize;
    }

    public void printItself(){
        System.out.println(size);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Vehicle){
            Vehicle vehicle = (Vehicle) someThingThatCanBeCompared;
            if(this.size < vehicle.size){
                return -1;
            } else if (this.size == vehicle.size){
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }



}
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

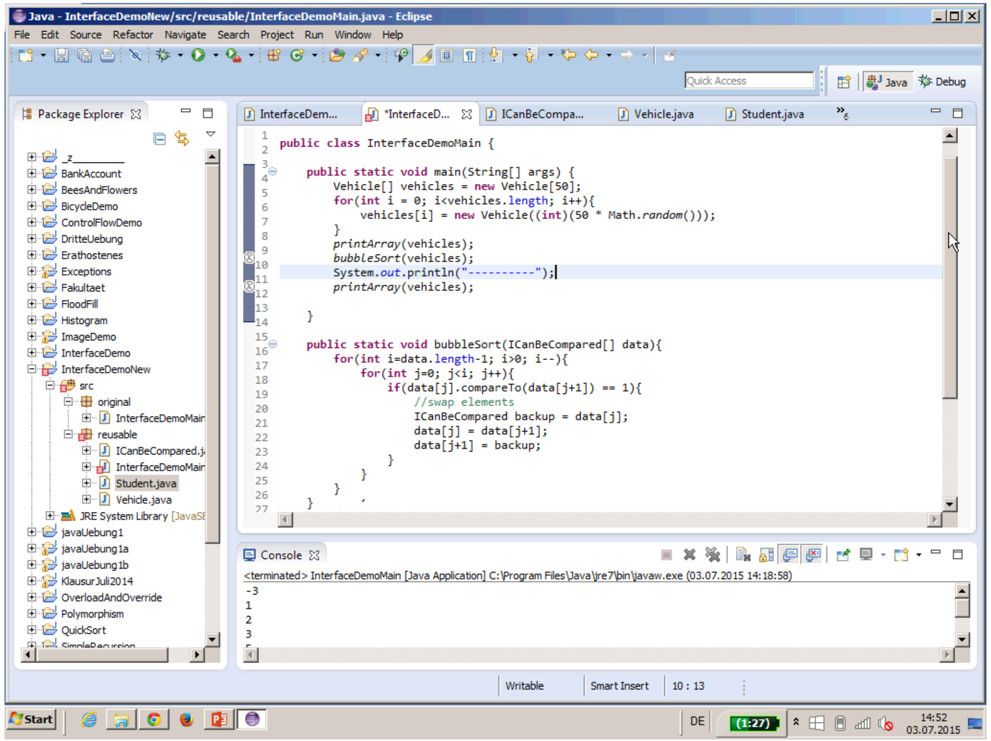
Writable | Smart Insert | 8 : 24



```java
package reusable;

public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle((int)(50 * Math.random()));
        }

        bubbleSort(vehicles);
        printArray(foo);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
```

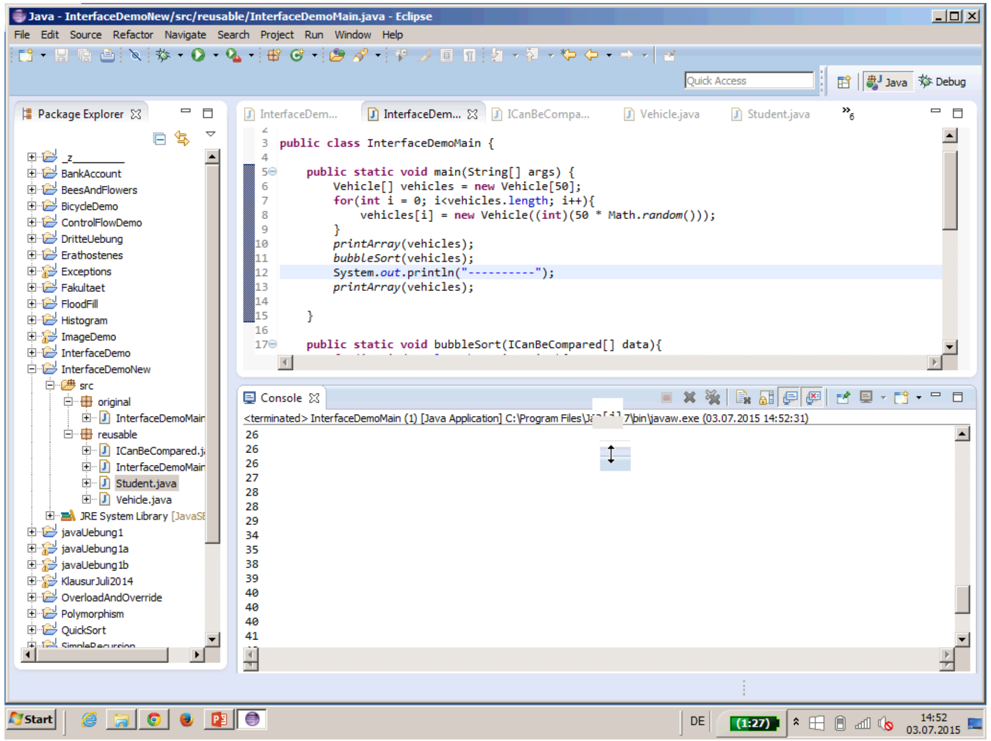
```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```

Writable | Smart Insert | 10 : 23



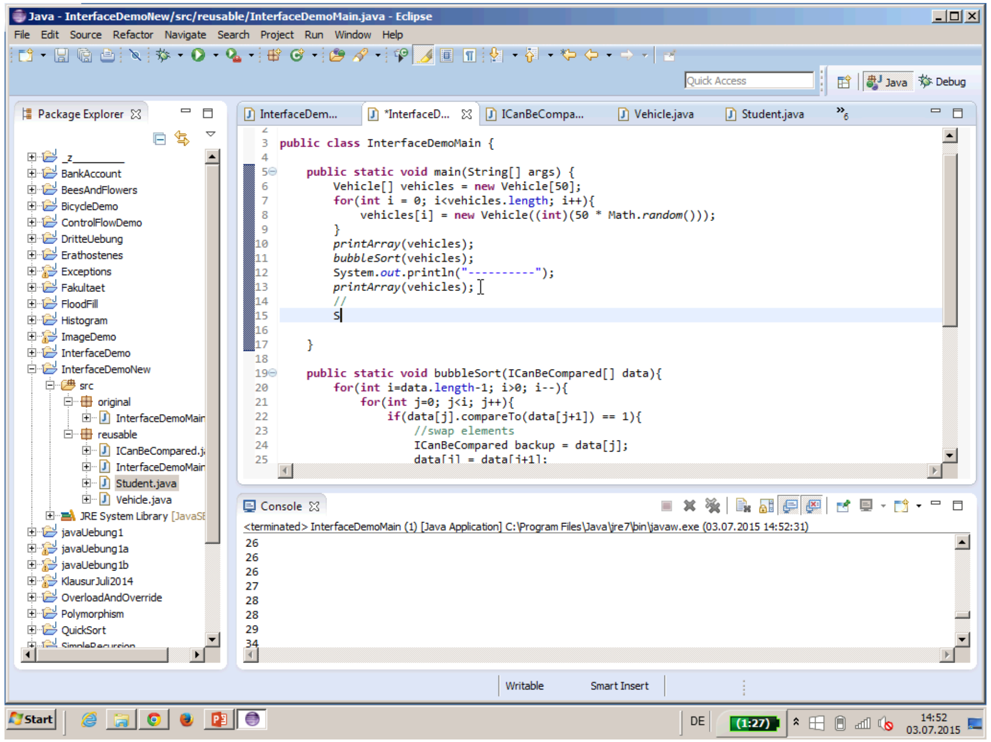
```java
public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle((int)(50 * Math.random()));
        }
        printArray(vehicles);
        bubbleSort(vehicles);
        System.out.println("----------");
        printArray(vehicles);

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
```

```
<terminated> InterfaceDemoMain [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:18:58)
-3
1
2
3
```



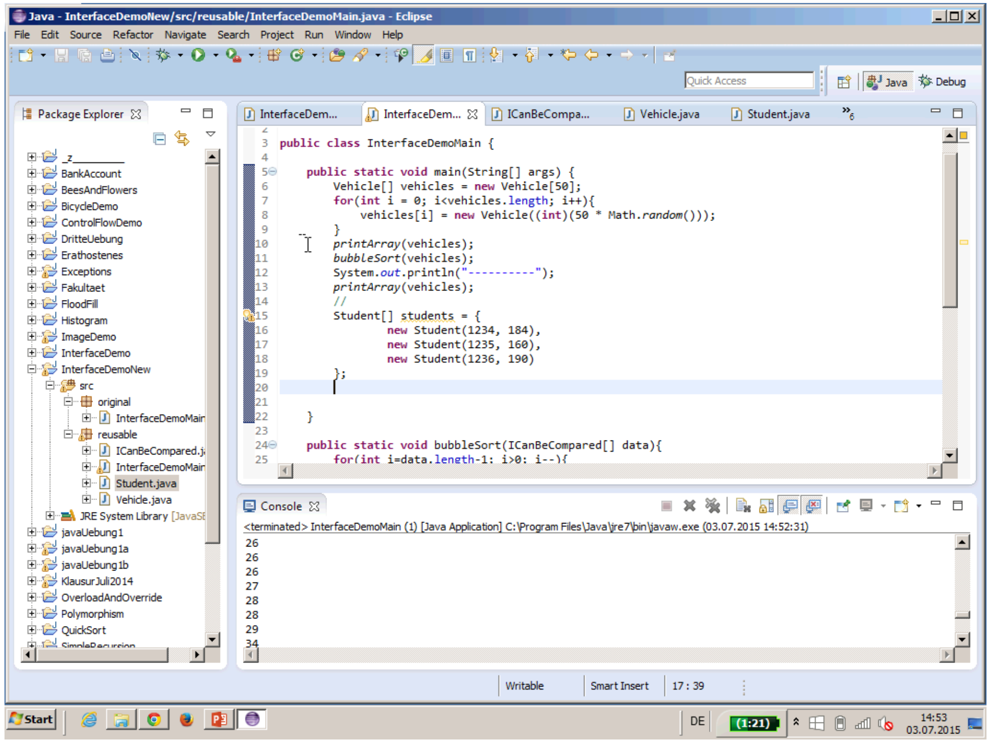
```java
public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle((int)(50 * Math.random()));
        }
        printArray(vehicles);
        bubbleSort(vehicles);
        System.out.println("----------");
        printArray(vehicles);

    }

    public static void bubbleSort(ICanBeCompared[] data){
```

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Ja    7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
34
35
38
39
40
40
40
41
```



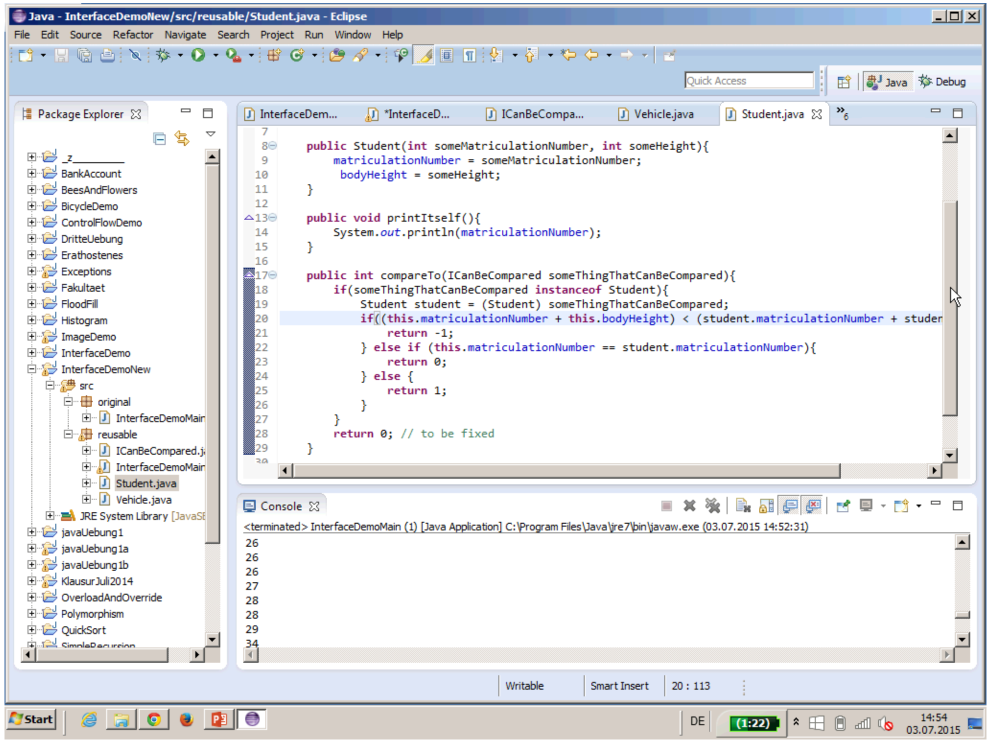
```java
public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle((int)(50 * Math.random()));
        }
        printArray(vehicles);
        bubbleSort(vehicles);
        System.out.println("----------");
        printArray(vehicles);
        //
        S

    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
```

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
34
```



```java
public class InterfaceDemoMain {

    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle((int)(50 * Math.random()));
        }
        printArray(vehicles);
        bubbleSort(vehicles);
        System.out.println("----------");
        printArray(vehicles);
        //
        Student[] students = {
                new Student(1234, 184),
                new Student(1235, 160),
                new Student(1236, 190)
        };


    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
```

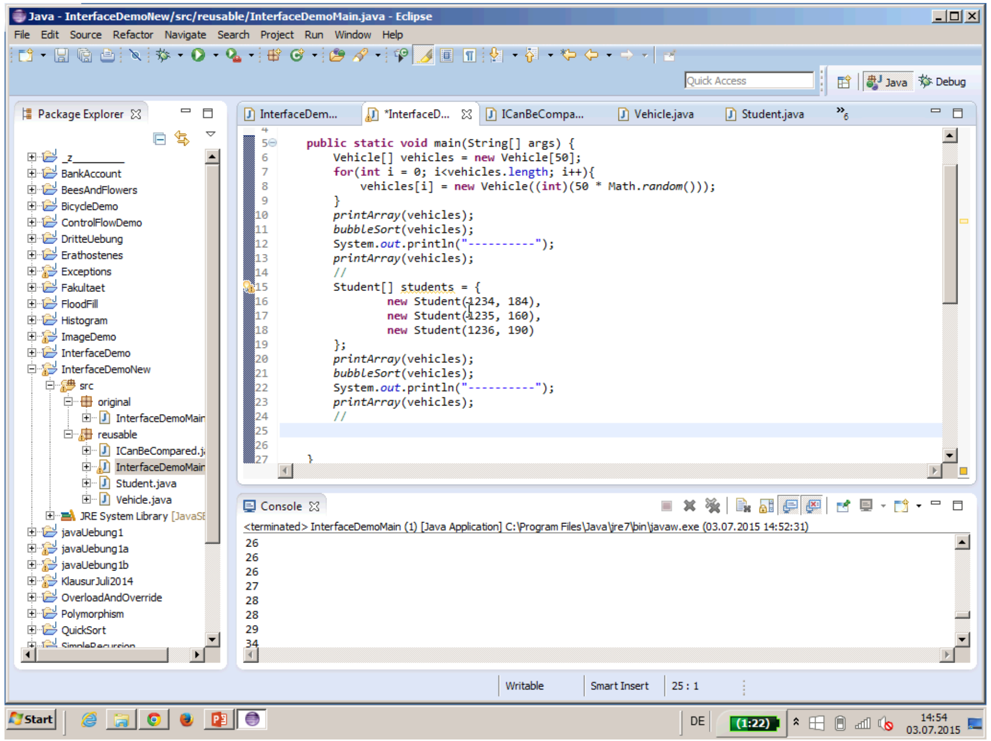
```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
34
```



Java - InterfaceDemoNew/src/reusable/Student.java - Eclipse

```java
    public Student(int someMatriculationNumber, int someHeight){
        matriculationNumber = someMatriculationNumber;
         bodyHeight = someHeight;
    }

    public void printItself(){
        System.out.println(matriculationNumber);
    }

    public int compareTo(ICanBeCompared someThingThatCanBeCompared){
        if(someThingThatCanBeCompared instanceof Student){
            Student student = (Student) someThingThatCanBeCompared;
            if((this.matriculationNumber + this.bodyHeight) < (student.matriculationNumber + studer
                return -1;
            } else if (this.matriculationNumber == student.matriculationNumber){
                return 0;
            } else {
                return 1;
            }
        }
        return 0; // to be fixed
    }
```

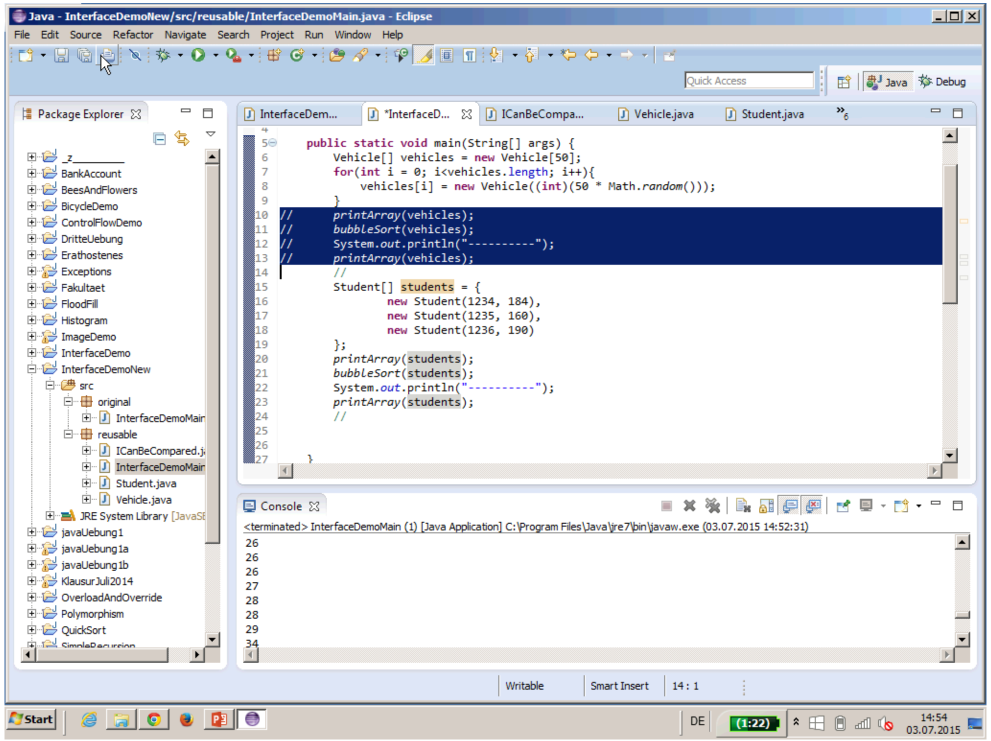
```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
34
```



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```java
    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle((int)(50 * Math.random()));
        }
        printArray(vehicles);
        bubbleSort(vehicles);
        System.out.println("----------");
        printArray(vehicles);
        //
        Student[] students = {
                new Student(1234, 184),
                new Student(1235, 160),
                new Student(1236, 190)
        };
        printArray(vehicles);
        bubbleSort(vehicles);
        System.out.println("----------");
        printArray(vehicles);
        //


    }
```

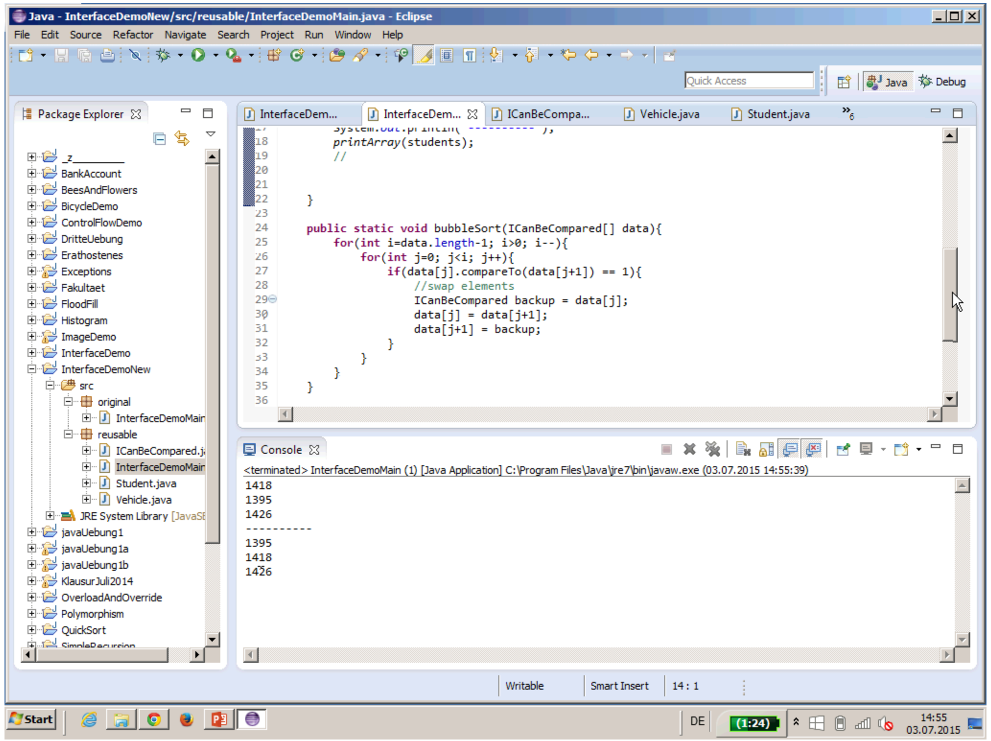
```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
34
```



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```java
    public static void main(String[] args) {
        Vehicle[] vehicles = new Vehicle[50];
        for(int i = 0; i<vehicles.length; i++){
            vehicles[i] = new Vehicle((int)(50 * Math.random()));
        }
//      printArray(vehicles);
//      bubbleSort(vehicles);
//      System.out.println("----------");
//      printArray(vehicles);
        //
        Student[] students = {
                new Student(1234, 184),
                new Student(1235, 160),
                new Student(1236, 190)
        };
        printArray(students);
        bubbleSort(students);
        System.out.println("----------");
        printArray(students);
        //


    }
```

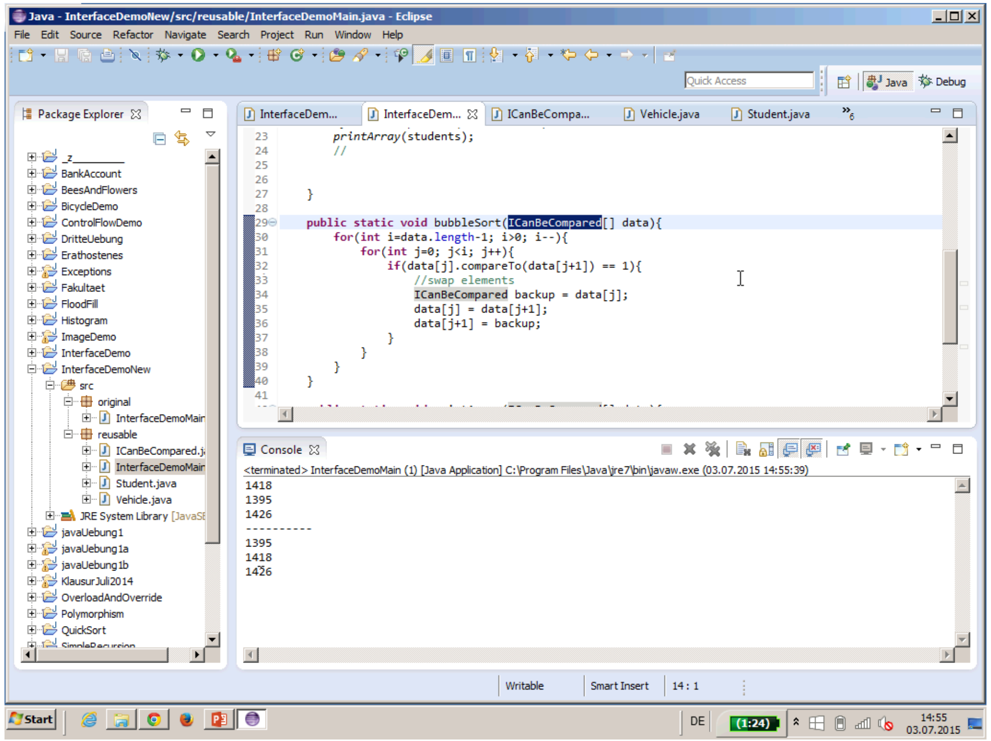
```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:52:31)
26
26
26
27
28
28
29
34
```



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```java
        System.out.println("----------");
        printArray(students);
        //


    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
```

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:55:39)
1418
1395
1426
----------
1395
1418
1426
```



Java - InterfaceDemoNew/src/reusable/InterfaceDemoMain.java - Eclipse

```java
        printArray(students);
        //


    }

    public static void bubbleSort(ICanBeCompared[] data){
        for(int i=data.length-1; i>0; i--){
            for(int j=0; j<i; j++){
                if(data[j].compareTo(data[j+1]) == 1){
                    //swap elements
                    ICanBeCompared backup = data[j];
                    data[j] = data[j+1];
                    data[j+1] = backup;
                }
            }
        }
    }
```

```
<terminated> InterfaceDemoMain (1) [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (03.07.2015 14:55:39)
1418
1395
1426
----------
1395
1418
1426
```