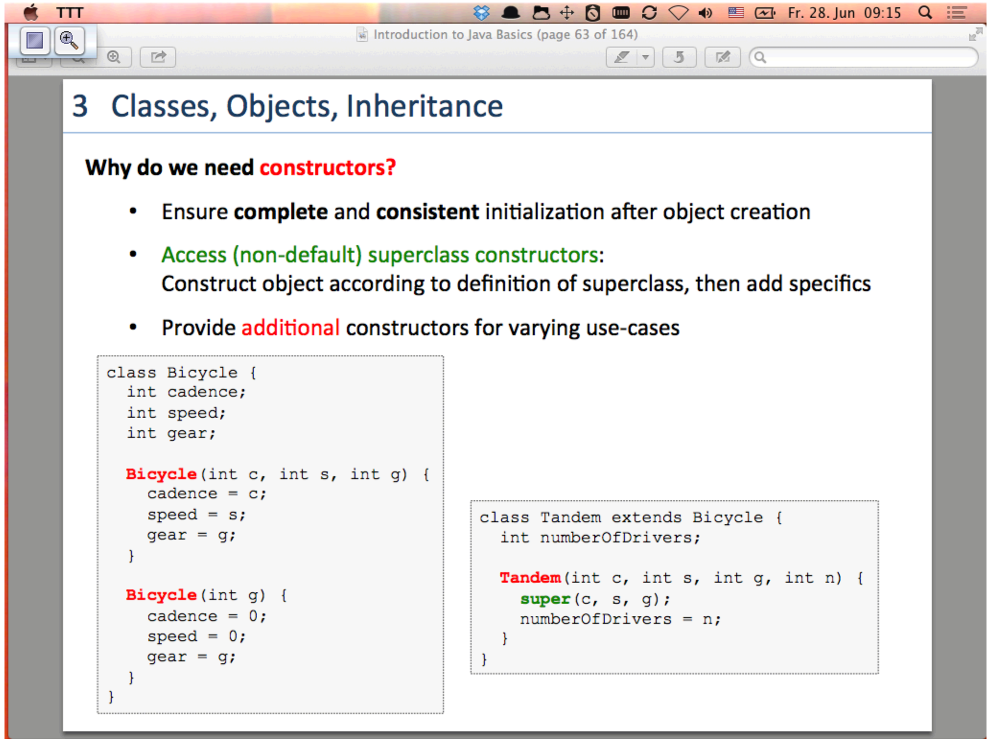
Script **generated by TTT**

Title: Lehmann: Uebung_Einf_HF (28.06.2013)

Date: Fri Jun 28 09:15:36 CEST 2013

Duration: 89:58 min

Pages: 67



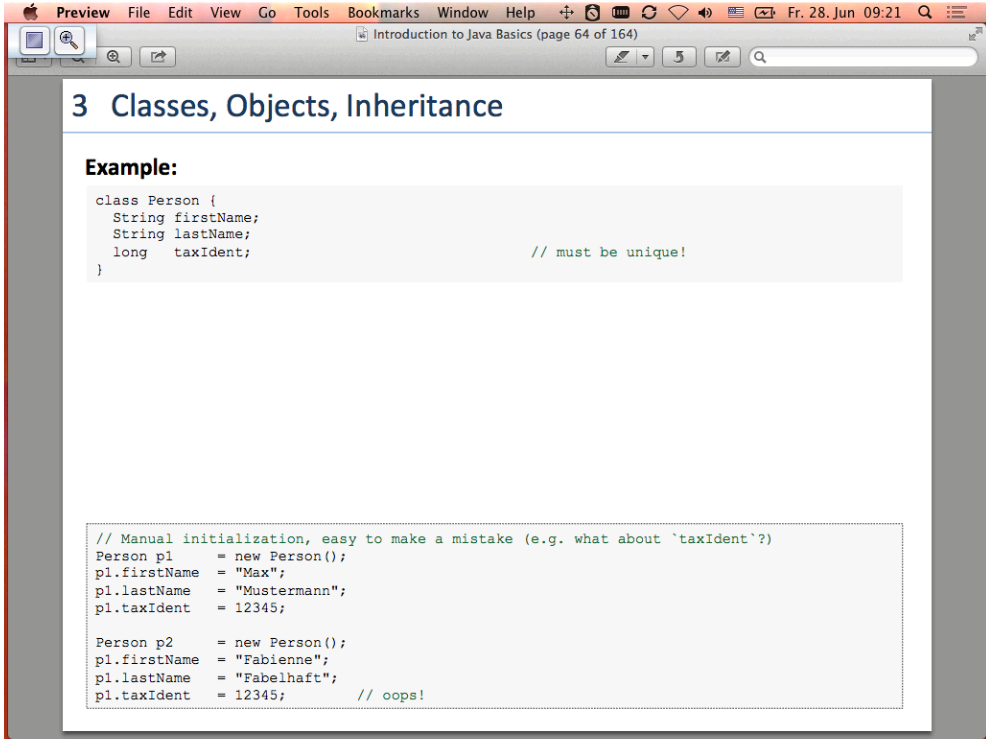
3 Classes, Objects, Inheritance

Why do we need constructors?

- Ensure **complete** and **consistent** initialization after object creation
- Access (non-default) superclass constructors:
  Construct object according to definition of superclass, then add specifics
- Provide additional constructors for varying use-cases

```
class Bicycle {
  int cadence;
  int speed;
  int gear;

  Bicycle(int c, int s, int g) {
    cadence = c;
    speed = s;
    gear = g;
  }

  Bicycle(int g) {
    cadence = 0;
    speed = 0;
    gear = g;
  }
}
```

```
class Tandem extends Bicycle {
  int numberOfDrivers;

  Tandem(int c, int s, int g, int n) {
    super(c, s, g);
    numberOfDrivers = n;
  }
}
```

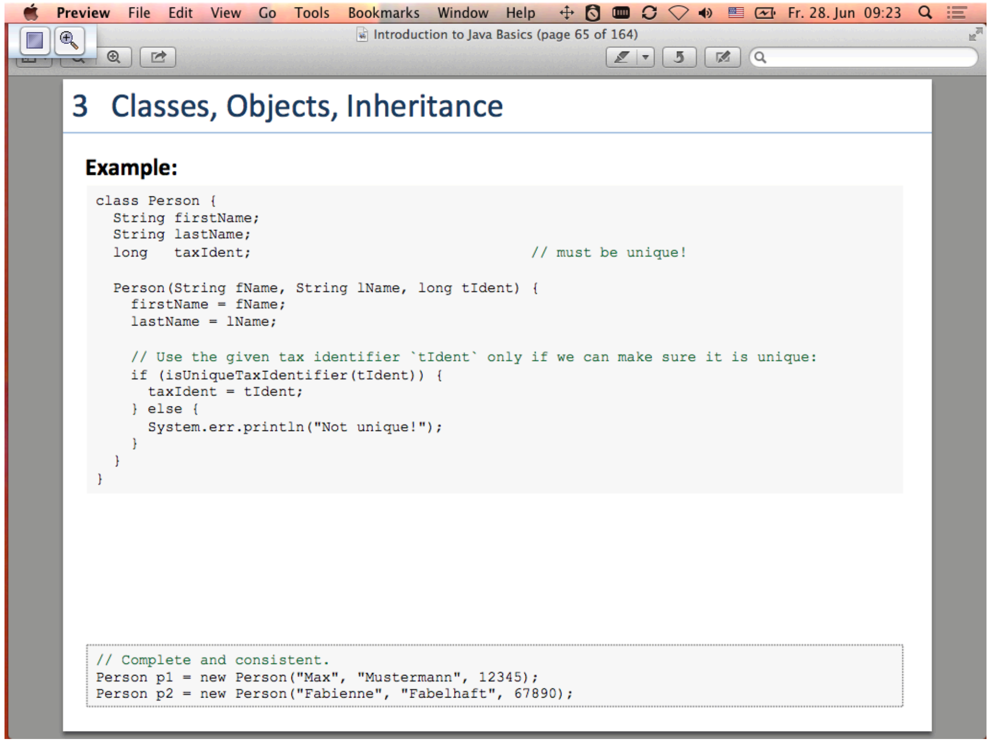

Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:21
Introduction to Java Basics (page 64 of 164)

3 Classes, Objects, Inheritance

Example:

```
class Person {
  String firstName;
  String lastName;
  long   taxIdent;                                  // must be unique!
}
```

```
// Manual initialization, easy to make a mistake (e.g. what about `taxIdent`?)
Person p1     = new Person();
p1.firstName  = "Max";
p1.lastName   = "Mustermann";
p1.taxIdent   = 12345;

Person p2     = new Person();
p1.firstName  = "Fabienne";
p1.lastName   = "Fabelhaft";
p1.taxIdent   = 12345;        // oops!
```

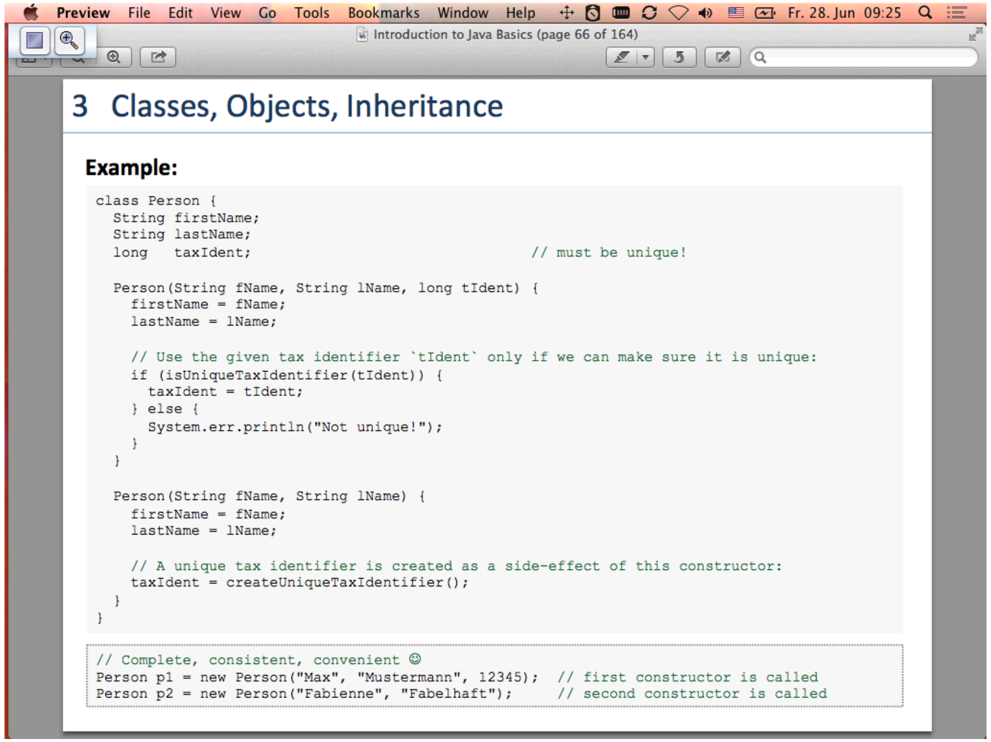

Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:23
Introduction to Java Basics (page 65 of 164)

3 Classes, Objects, Inheritance

Example:

```
class Person {
  String firstName;
  String lastName;
  long   taxIdent;                                  // must be unique!

  Person(String fName, String lName, long tIdent) {
    firstName = fName;
    lastName = lName;

    // Use the given tax identifier `tIdent` only if we can make sure it is unique:
    if (isUniqueTaxIdentifier(tIdent)) {
      taxIdent = tIdent;
    } else {
      System.err.println("Not unique!");
    }
  }
}
```

```
// Complete and consistent.
Person p1 = new Person("Max", "Mustermann", 12345);
Person p2 = new Person("Fabienne", "Fabelhaft", 67890);
```

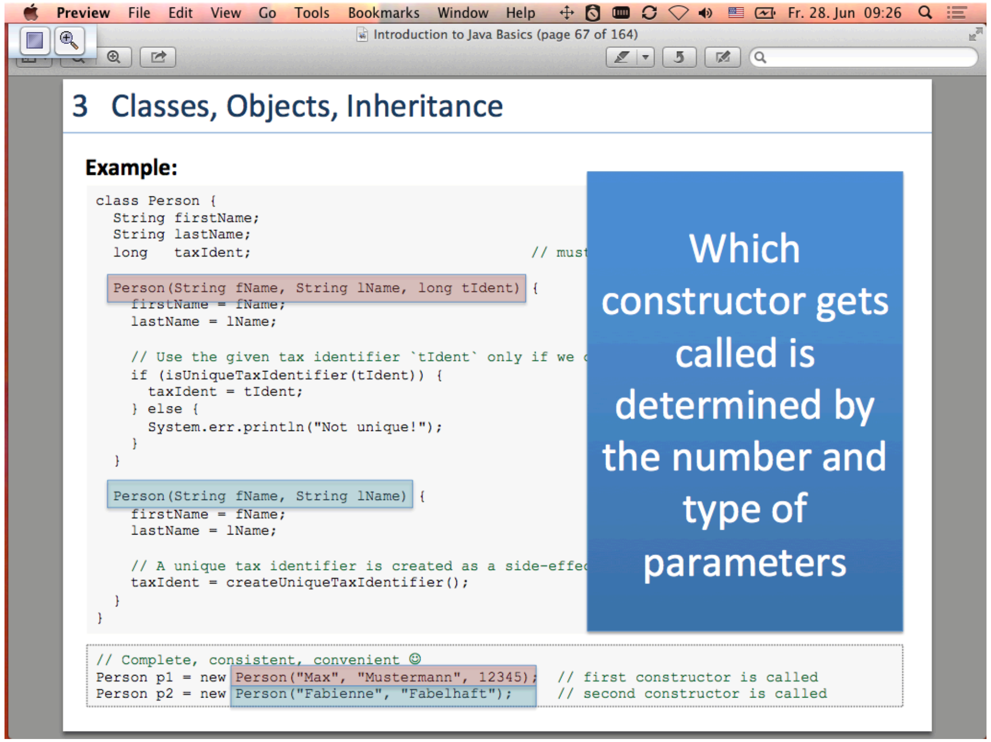

Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:25
Introduction to Java Basics (page 66 of 164)

3 Classes, Objects, Inheritance

Example:

```
class Person {
  String firstName;
  String lastName;
  long   taxIdent;                              // must be unique!

  Person(String fName, String lName, long tIdent) {
    firstName = fName;
    lastName = lName;

    // Use the given tax identifier `tIdent` only if we can make sure it is unique:
    if (isUniqueTaxIdentifier(tIdent)) {
      taxIdent = tIdent;
    } else {
      System.err.println("Not unique!");
    }
  }

  Person(String fName, String lName) {
    firstName = fName;
    lastName = lName;

    // A unique tax identifier is created as a side-effect of this constructor:
    taxIdent = createUniqueTaxIdentifier();
  }
}
```

```
// Complete, consistent, convenient ☺
Person p1 = new Person("Max", "Mustermann", 12345);  // first constructor is called
Person p2 = new Person("Fabienne", "Fabelhaft");     // second constructor is called
```


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:26
Introduction to Java Basics (page 67 of 164)

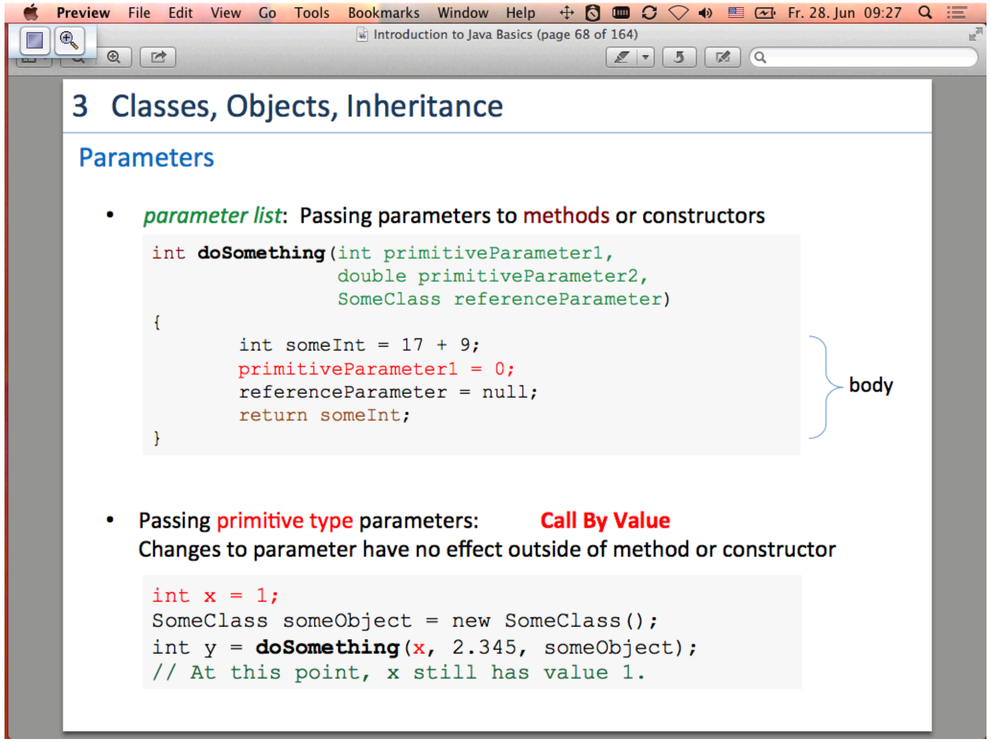
3 Classes, Objects, Inheritance

Example:

```
class Person {
  String firstName;
  String lastName;
  long   taxIdent;                              // must

  Person(String fName, String lName, long tIdent) {
    firstName = fName;
    lastName = lName;

    // Use the given tax identifier `tIdent` only if we c
    if (isUniqueTaxIdentifier(tIdent)) {
      taxIdent = tIdent;
    } else {
      System.err.println("Not unique!");
    }
  }

  Person(String fName, String lName) {
    firstName = fName;
    lastName = lName;

    // A unique tax identifier is created as a side-effe
    taxIdent = createUniqueTaxIdentifier();
  }
}
```

Which constructor gets called is determined by the number and type of parameters

```
// Complete, consistent, convenient ☺
Person p1 = new Person("Max", "Mustermann", 12345);  // first constructor is called
Person p2 = new Person("Fabienne", "Fabelhaft");     // second constructor is called
```


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:27
Introduction to Java Basics (page 68 of 164)

3 Classes, Objects, Inheritance

Parameters

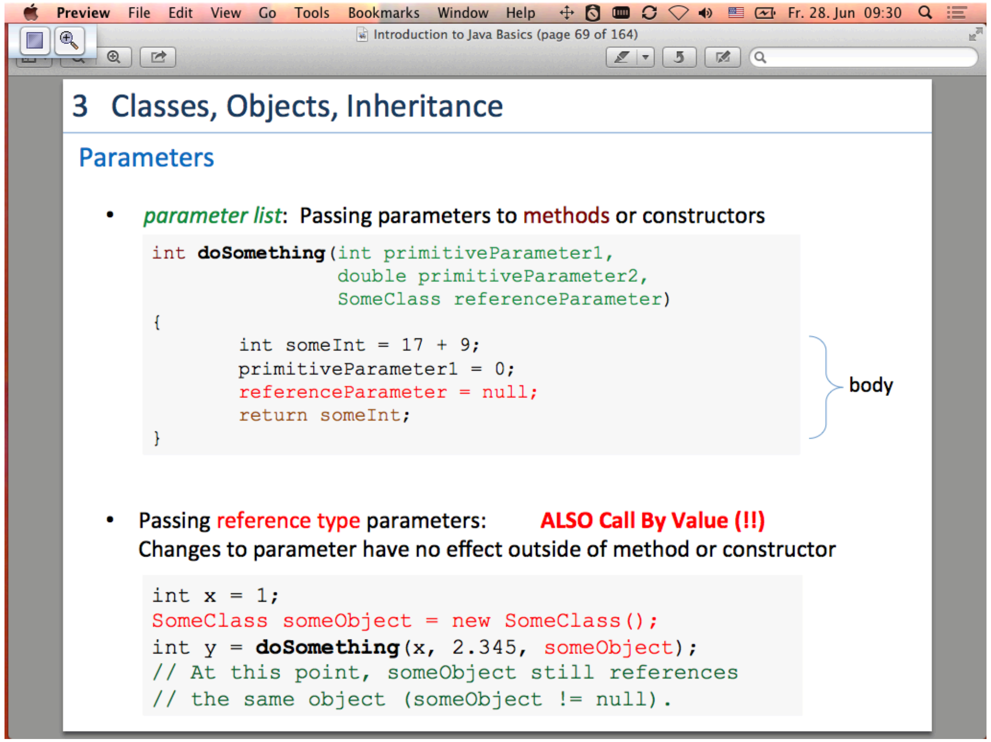
- *parameter list*: Passing parameters to methods or constructors

```
int doSomething(int primitiveParameter1,
                double primitiveParameter2,
                SomeClass referenceParameter)
{
        int someInt = 17 + 9;
        primitiveParameter1 = 0;
        referenceParameter = null;
        return someInt;
}
```

body

- Passing primitive type parameters: **Call By Value**
  Changes to parameter have no effect outside of method or constructor

```
int x = 1;
SomeClass someObject = new SomeClass();
int y = doSomething(x, 2.345, someObject);
// At this point, x still has value 1.
```


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:30
Introduction to Java Basics (page 69 of 164)

3 Classes, Objects, Inheritance

Parameters

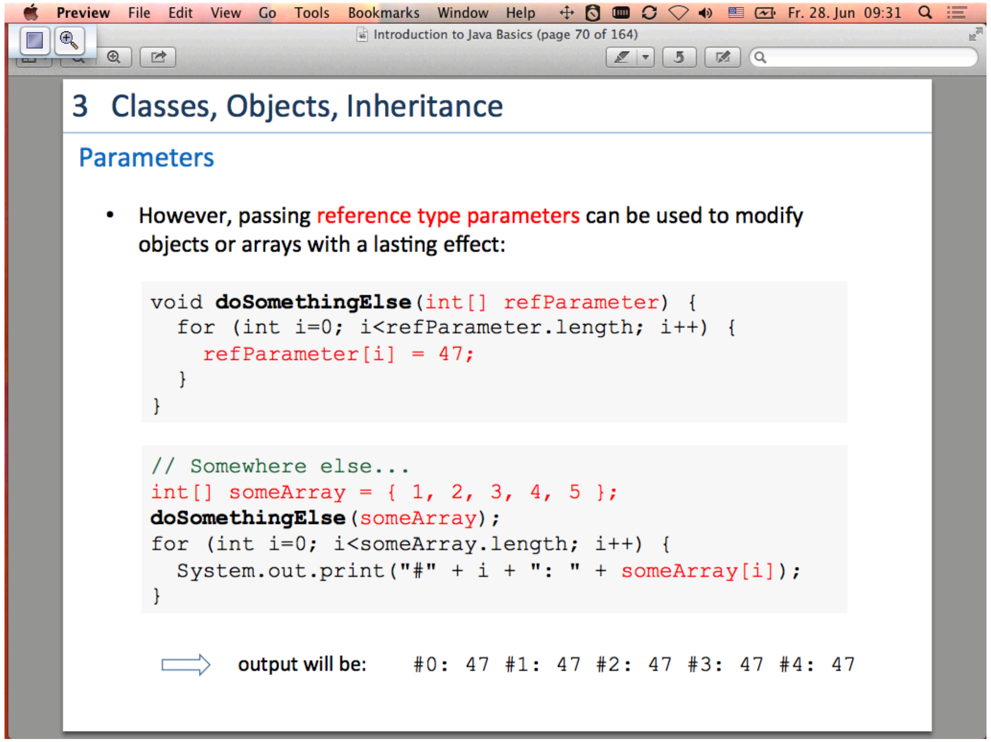
- *parameter list*: Passing parameters to methods or constructors

```
int doSomething(int primitiveParameter1,
                double primitiveParameter2,
                SomeClass referenceParameter)
{
        int someInt = 17 + 9;
        primitiveParameter1 = 0;
        referenceParameter = null;
        return someInt;
}
```

body

- Passing reference type parameters: **ALSO Call By Value (!!)**
  Changes to parameter have no effect outside of method or constructor

```
int x = 1;
SomeClass someObject = new SomeClass();
int y = doSomething(x, 2.345, someObject);
// At this point, someObject still references
// the same object (someObject != null).
```


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:31
Introduction to Java Basics (page 70 of 164)

3 Classes, Objects, Inheritance

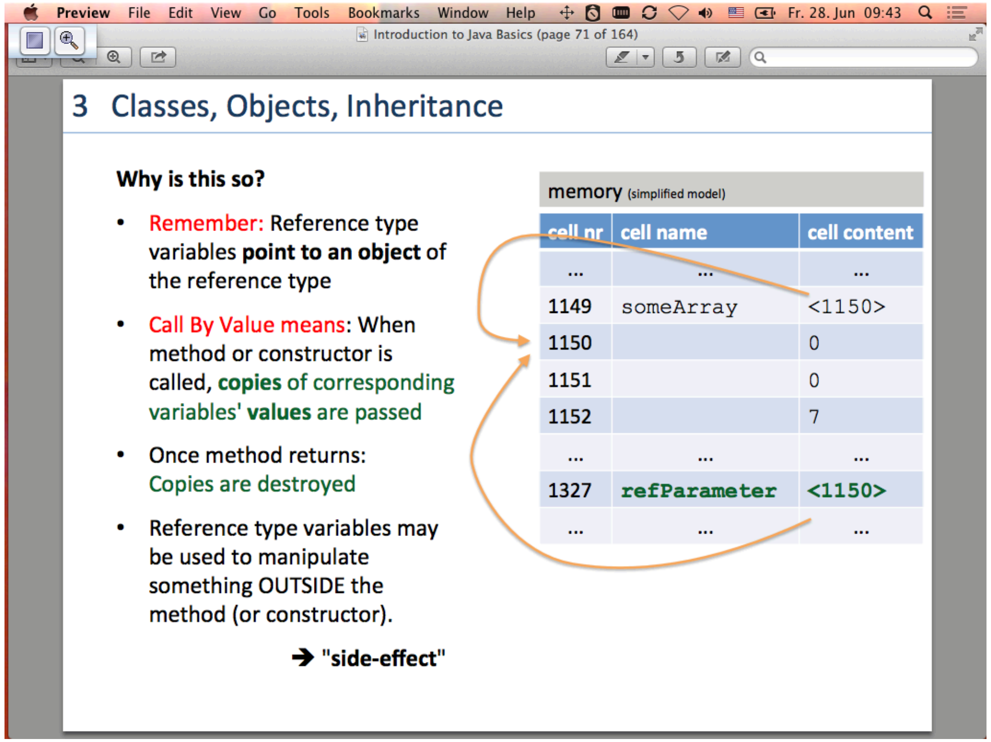
Parameters

- However, passing reference type parameters can be used to modify objects or arrays with a lasting effect:

```
void doSomethingElse(int[] refParameter) {
  for (int i=0; i<refParameter.length; i++) {
    refParameter[i] = 47;
  }
}
```

```
// Somewhere else...
int[] someArray = { 1, 2, 3, 4, 5 };
doSomethingElse(someArray);
for (int i=0; i<someArray.length; i++) {
  System.out.print("#" + i + ": " + someArray[i]);
}
```

⇨ output will be: `#0: 47 #1: 47 #2: 47 #3: 47 #4: 47`


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:43
Introduction to Java Basics (page 71 of 164)

3 Classes, Objects, Inheritance

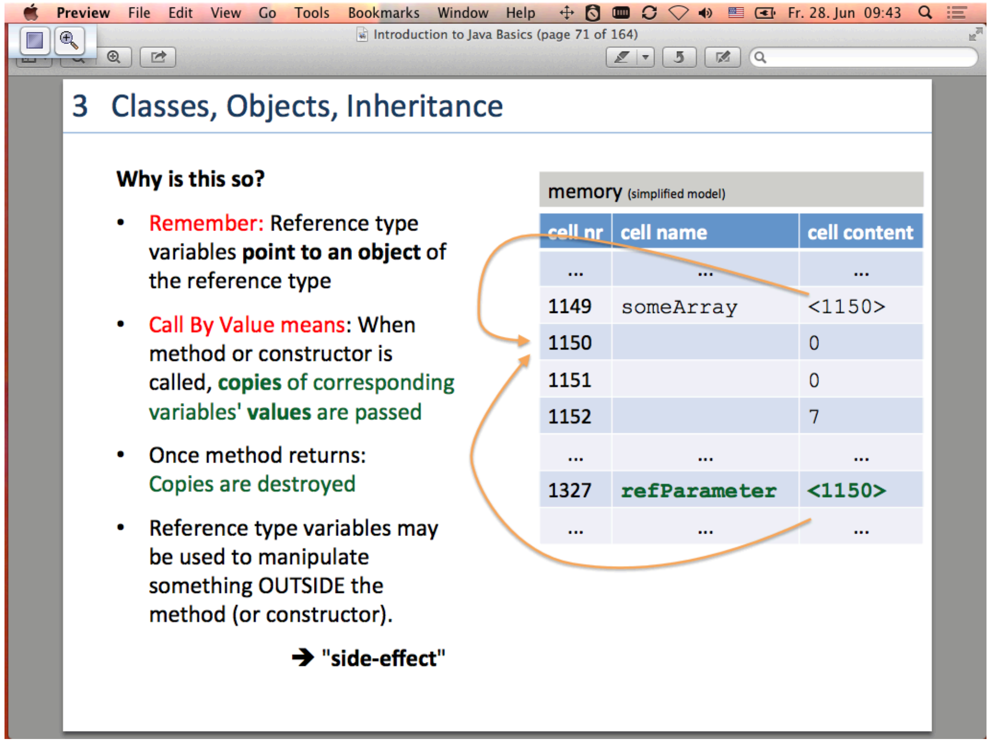
Why is this so?

- Remember: Reference type variables **point to an object** of the reference type
- Call By Value means: When method or constructor is called, **copies** of corresponding variables' **values** are passed
- Once method returns: Copies are destroyed
- Reference type variables may be used to manipulate something OUTSIDE the method (or constructor).

➔ **"side-effect"**

memory (simplified model)

cell nr	cell name	cell content
...	...	...
1149	someArray	<1150>
1150		0
1151		0
1152		7
...	...	...
1327	**refParameter**	**<1150>**
...	...	...



Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:43
Introduction to Java Basics (page 71 of 164)

3 Classes, Objects, Inheritance

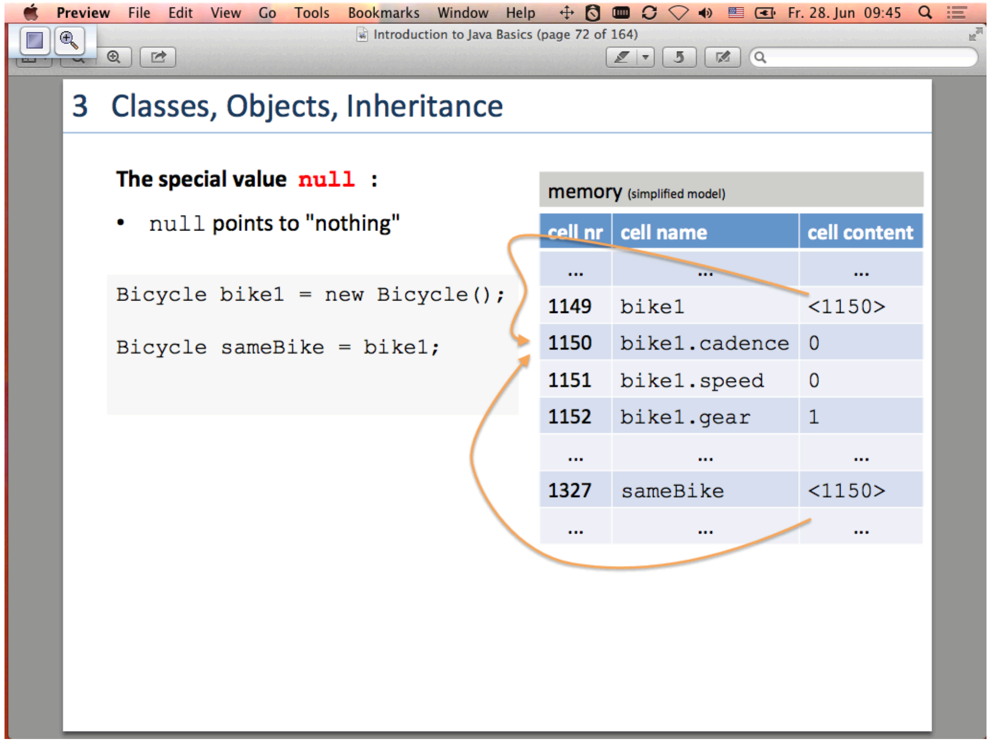
Why is this so?

- Remember: Reference type variables **point to an object** of the reference type
- Call By Value means: When method or constructor is called, **copies** of corresponding variables' **values** are passed
- Once method returns: Copies are destroyed
- Reference type variables may be used to manipulate something OUTSIDE the method (or constructor).

➔ **"side-effect"**

memory (simplified model)

cell nr	cell name	cell content
...	...	...
1149	someArray	<1150>
1150		0
1151		0
1152		7
...	...	...
1327	**refParameter**	**<1150>**
...	...	...


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:45
Introduction to Java Basics (page 72 of 164)

3 Classes, Objects, Inheritance

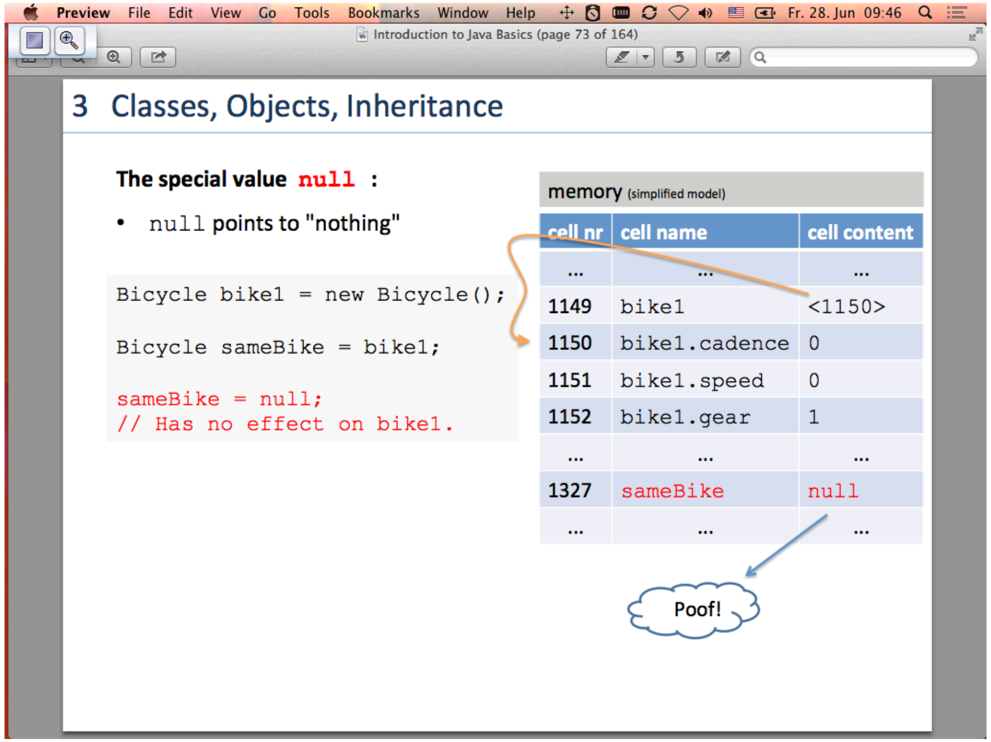
The special value `null` :

- `null` points to "nothing"

```
Bicycle bike1 = new Bicycle();

Bicycle sameBike = bike1;
```

memory (simplified model)

cell nr	cell name	cell content
...	...	...
1149	bike1	<1150>
1150	bike1.cadence	0
1151	bike1.speed	0
1152	bike1.gear	1
...	...	...
1327	sameBike	<1150>
...	...	...


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:46
Introduction to Java Basics (page 73 of 164)

3 Classes, Objects, Inheritance

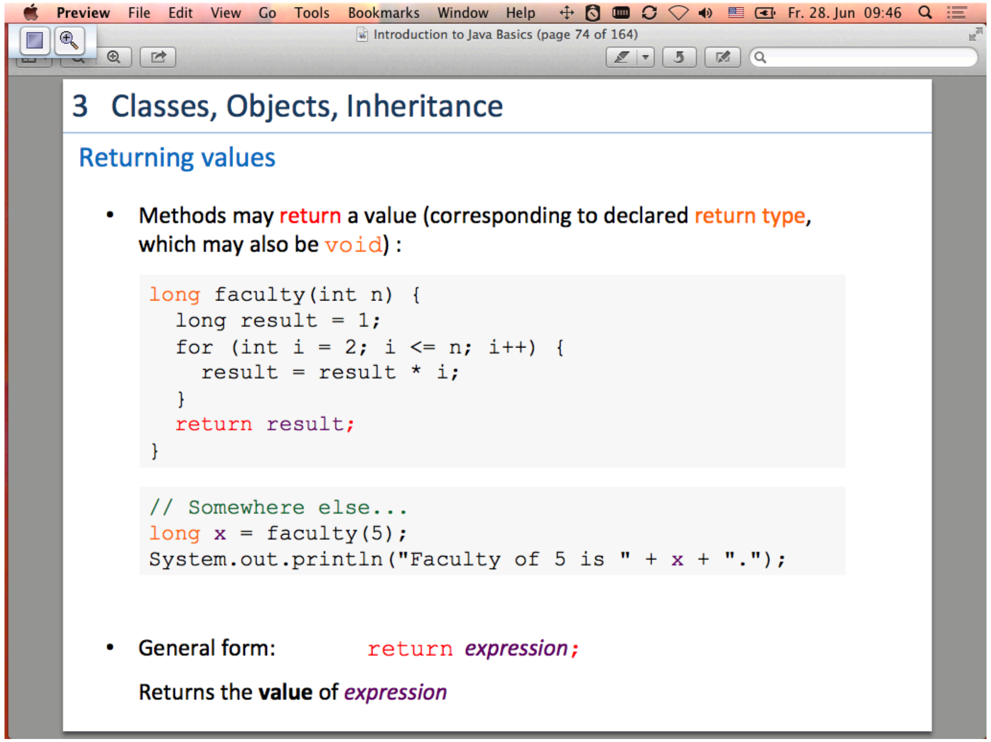
The special value null :

- null points to "nothing"

```
Bicycle bike1 = new Bicycle();

Bicycle sameBike = bike1;

sameBike = null;
// Has no effect on bike1.
```

memory (simplified model)

cell nr	cell name	cell content
...	...	...
1149	bike1	<1150>
1150	bike1.cadence	0
1151	bike1.speed	0
1152	bike1.gear	1
...	...	...
1327	sameBike	null
...	...	...


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:46
Introduction to Java Basics (page 74 of 164)

3 Classes, Objects, Inheritance

Returning values

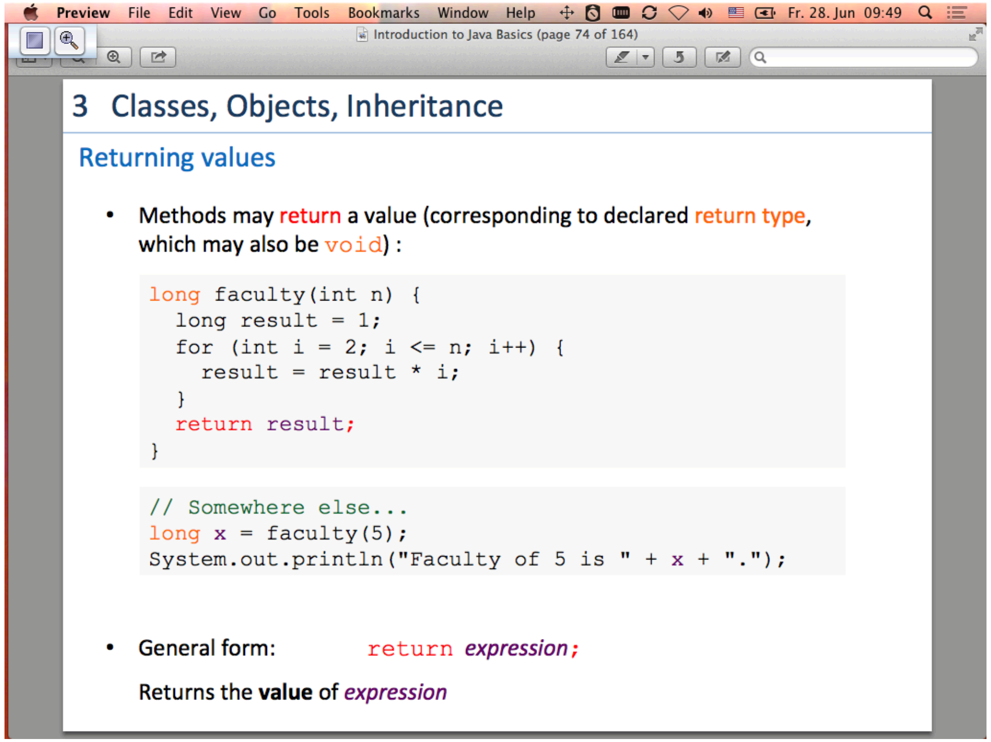
- Methods may return a value (corresponding to declared return type, which may also be void) :

```
long faculty(int n) {
  long result = 1;
  for (int i = 2; i <= n; i++) {
    result = result * i;
  }
  return result;
}
```

```
// Somewhere else...
long x = faculty(5);
System.out.println("Faculty of 5 is " + x + ".");
```

- General form: return *expression*;

  Returns the **value** of *expression*


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:49
Introduction to Java Basics (page 74 of 164)

3 Classes, Objects, Inheritance

Returning values

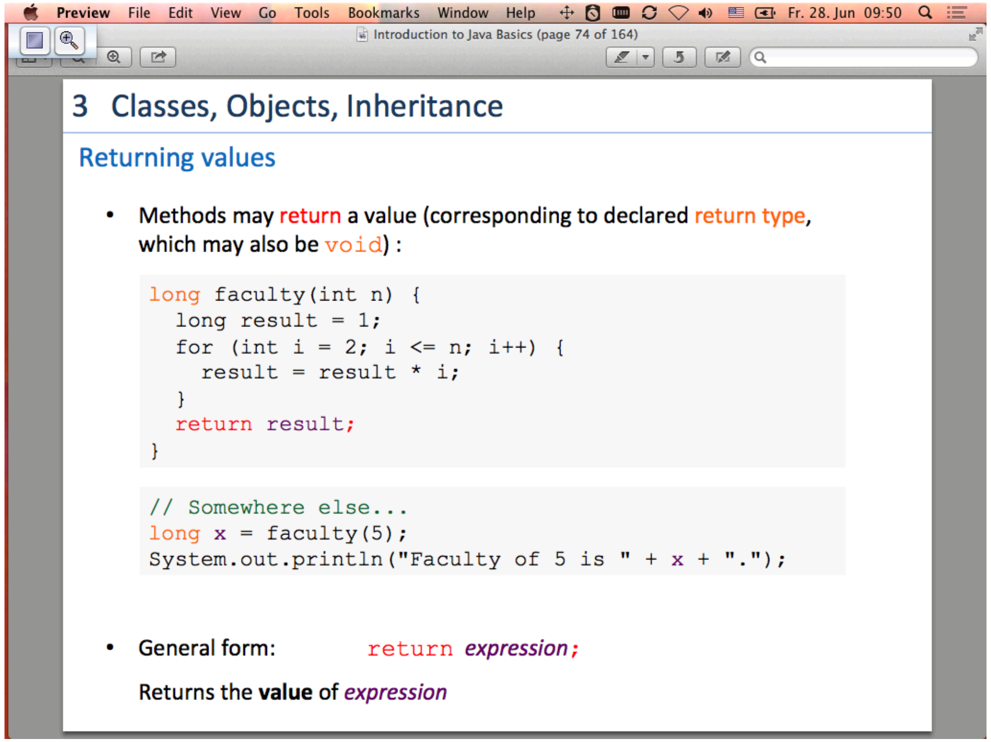
- Methods may return a value (corresponding to declared return type, which may also be void) :

```
long faculty(int n) {
  long result = 1;
  for (int i = 2; i <= n; i++) {
    result = result * i;
  }
  return result;
}
```

```
// Somewhere else...
long x = faculty(5);
System.out.println("Faculty of 5 is " + x + ".");
```

- General form: return *expression*;

  Returns the **value** of *expression*


Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:50
Introduction to Java Basics (page 74 of 164)

3 Classes, Objects, Inheritance

Returning values

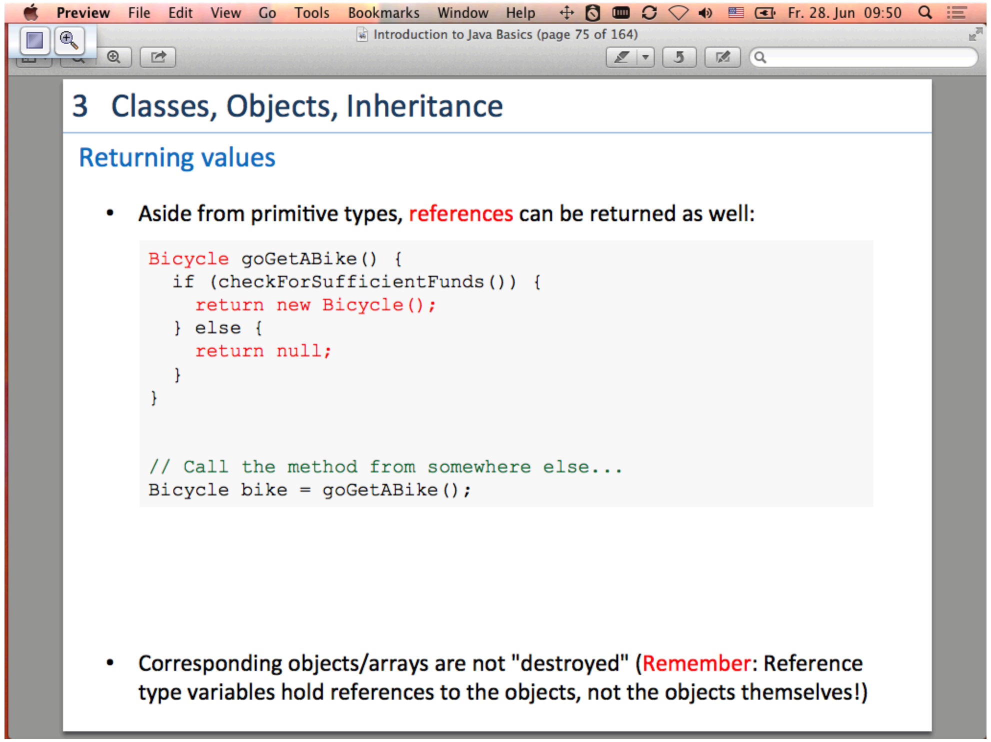
- Methods may return a value (corresponding to declared return type, which may also be void) :

```
long faculty(int n) {
  long result = 1;
  for (int i = 2; i <= n; i++) {
    result = result * i;
  }
  return result;
}
```

```
// Somewhere else...
long x = faculty(5);
System.out.println("Faculty of 5 is " + x + ".");
```

- General form: return *expression*;

  Returns the **value** of *expression*



Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:50
Introduction to Java Basics (page 75 of 164)

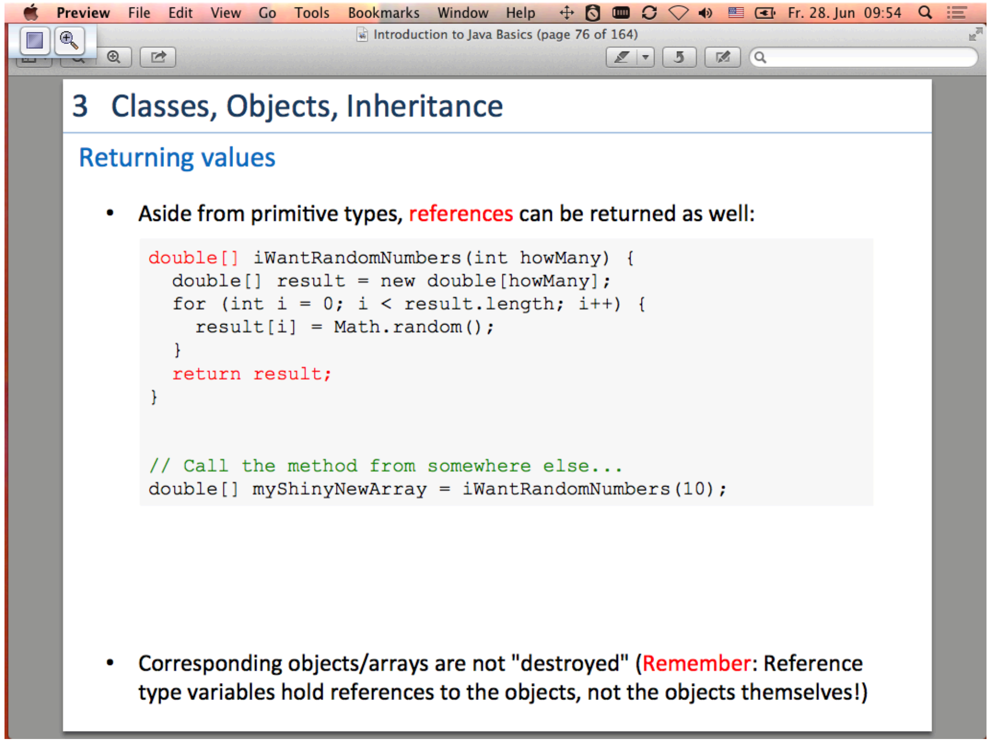
3 Classes, Objects, Inheritance

Returning values

- Aside from primitive types, references can be returned as well:

```
Bicycle goGetABike() {
  if (checkForSufficientFunds()) {
    return new Bicycle();
  } else {
    return null;
  }
}


// Call the method from somewhere else...
Bicycle bike = goGetABike();
```

- Corresponding objects/arrays are not "destroyed" (Remember: Reference type variables hold references to the objects, not the objects themselves!)



Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 09:54
Introduction to Java Basics (page 76 of 164)

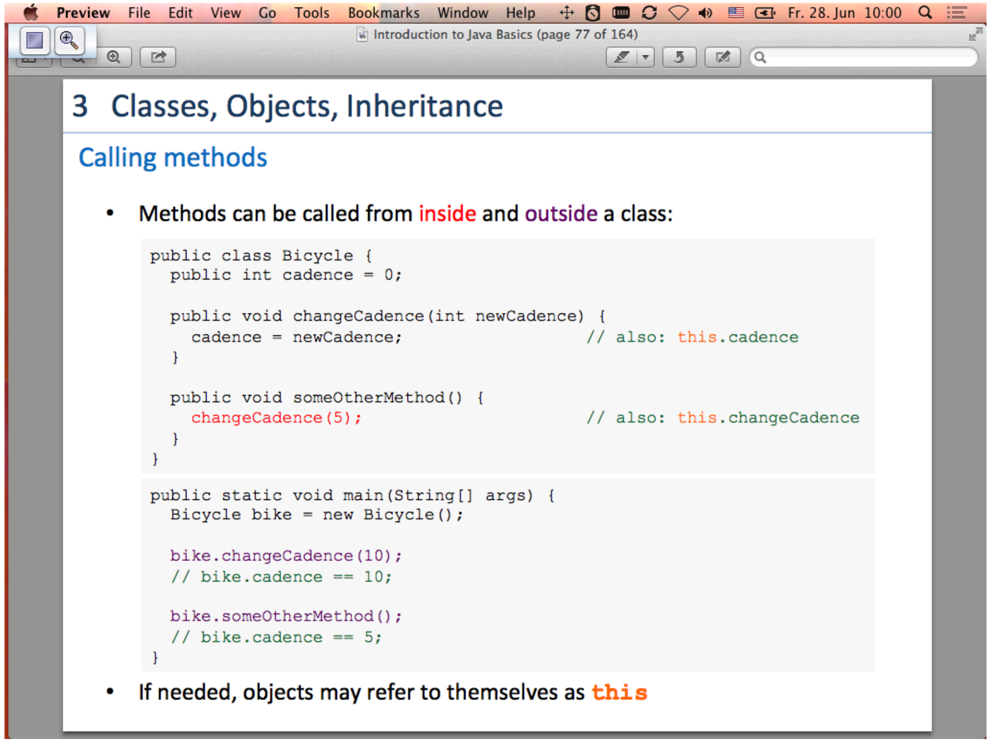
3 Classes, Objects, Inheritance

Returning values

- Aside from primitive types, references can be returned as well:

```
double[] iWantRandomNumbers(int howMany) {
  double[] result = new double[howMany];
  for (int i = 0; i < result.length; i++) {
    result[i] = Math.random();
  }
  return result;
}


// Call the method from somewhere else...
double[] myShinyNewArray = iWantRandomNumbers(10);
```

- Corresponding objects/arrays are not "destroyed" (Remember: Reference type variables hold references to the objects, not the objects themselves!)



Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 10:00
Introduction to Java Basics (page 77 of 164)

3 Classes, Objects, Inheritance

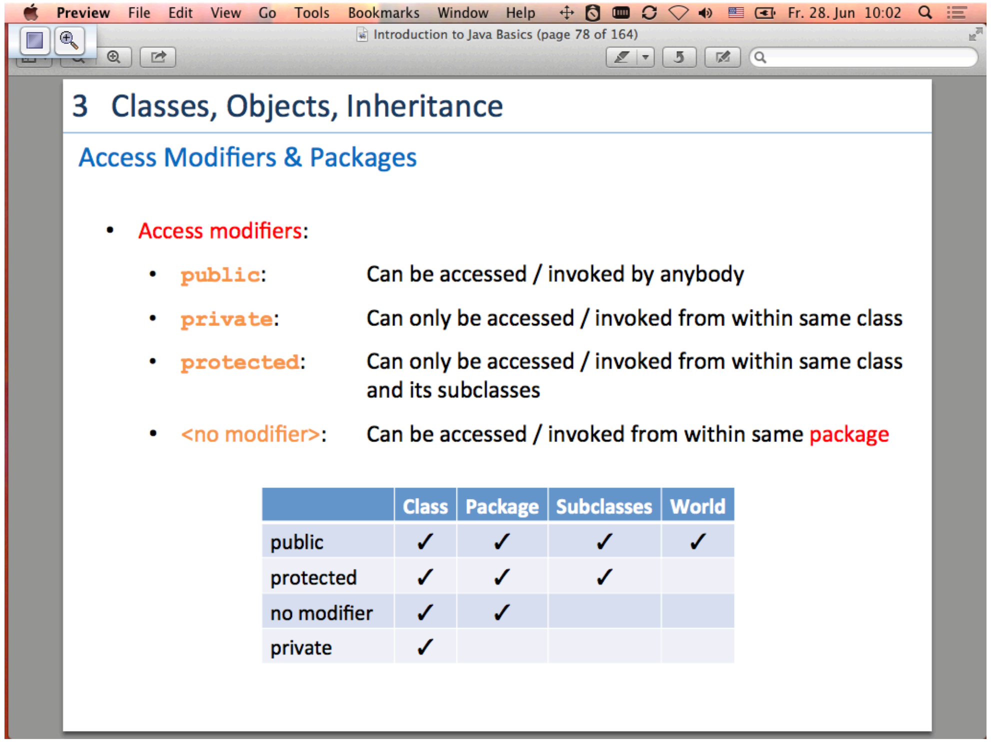
Calling methods

- Methods can be called from inside and outside a class:

```
public class Bicycle {
  public int cadence = 0;

  public void changeCadence(int newCadence) {
    cadence = newCadence;                 // also: this.cadence
  }

  public void someOtherMethod() {
    changeCadence(5);                     // also: this.changeCadence
  }
}
```

```
public static void main(String[] args) {
  Bicycle bike = new Bicycle();

  bike.changeCadence(10);
  // bike.cadence == 10;

  bike.someOtherMethod();
  // bike.cadence == 5;
}
```

- If needed, objects may refer to themselves as `this`



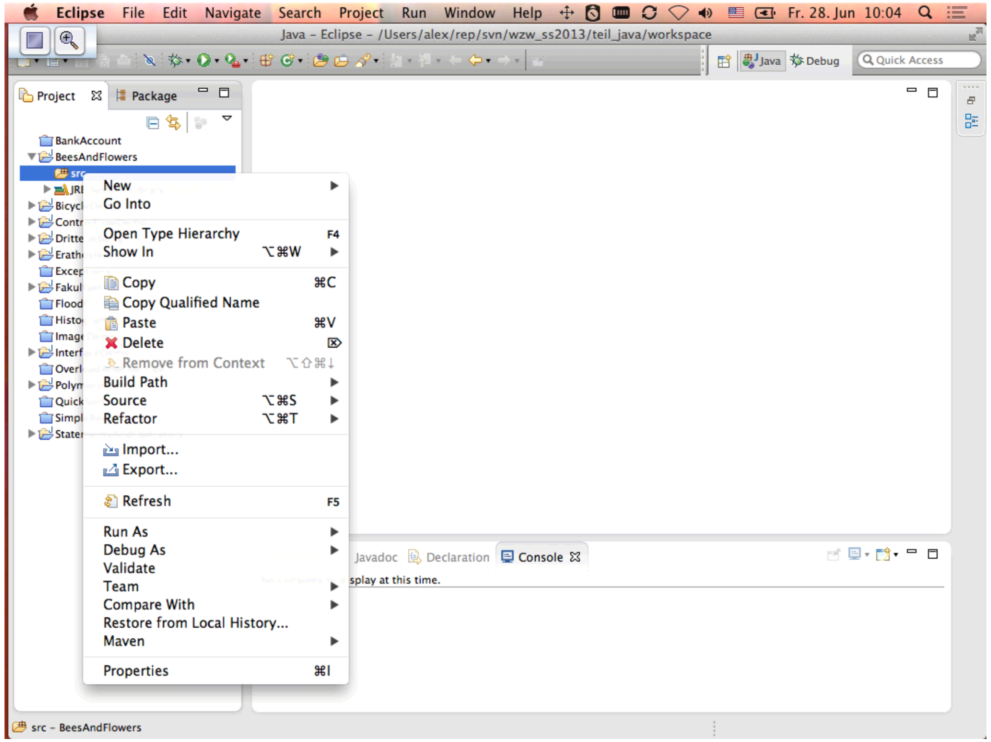
Preview File Edit View Go Tools Bookmarks Window Help
Fr. 28. Jun 10:02
Introduction to Java Basics (page 78 of 164)

3 Classes, Objects, Inheritance

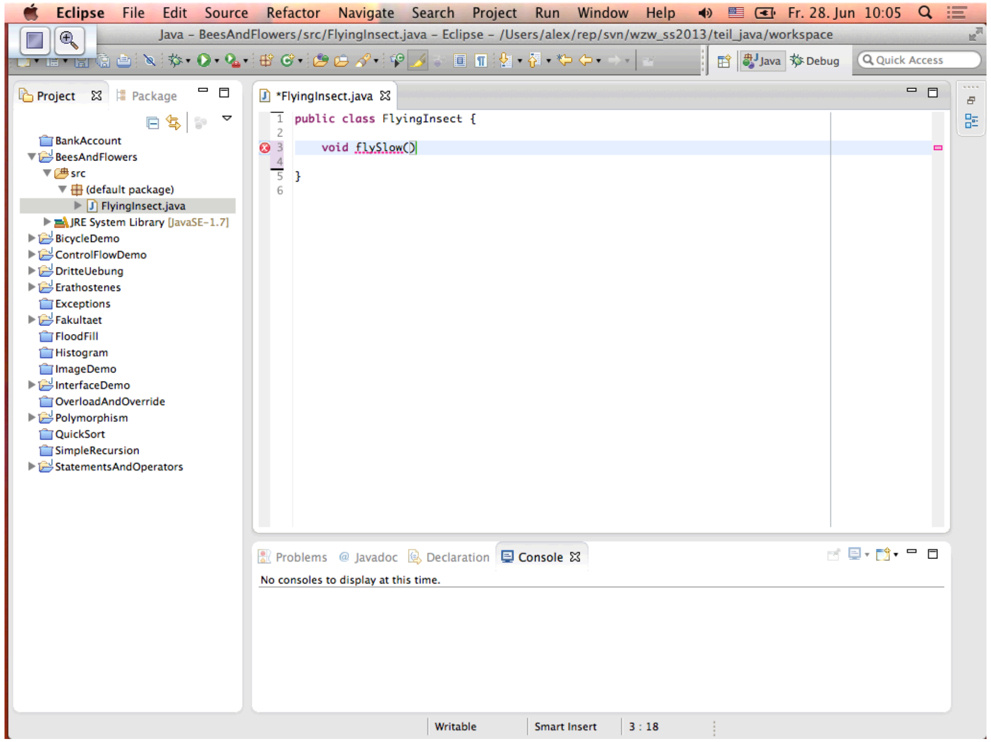
Access Modifiers & Packages

- Access modifiers:
  - `public`: Can be accessed / invoked by anybody
  - `private`: Can only be accessed / invoked from within same class
  - `protected`: Can only be accessed / invoked from within same class and its subclasses
  - <no modifier>: Can be accessed / invoked from within same package

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

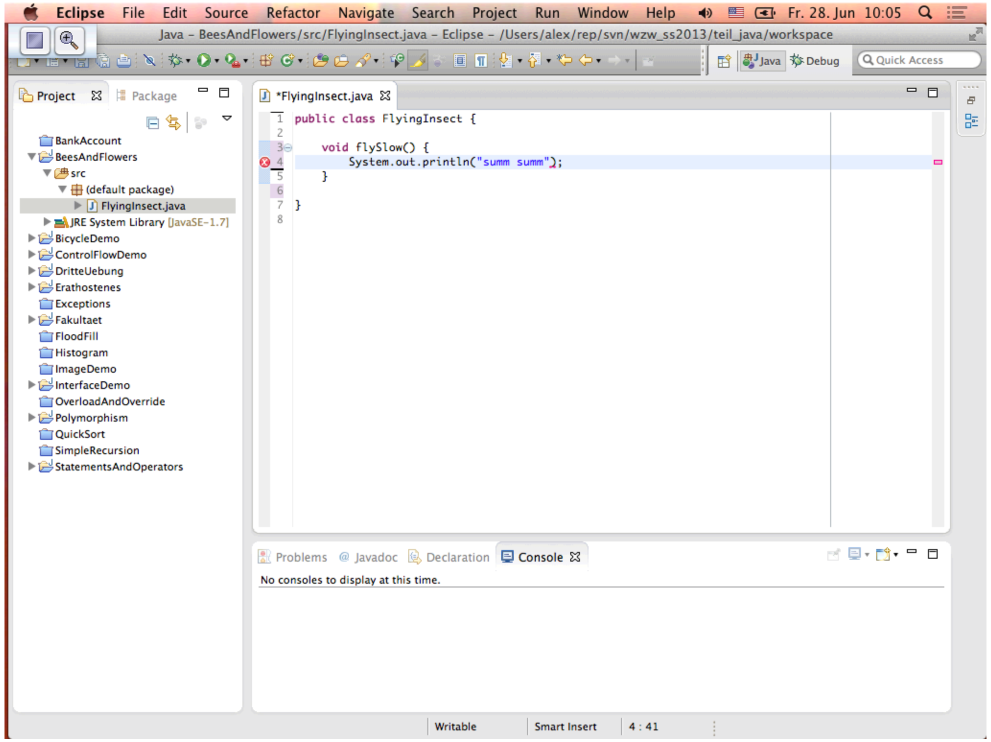


splay at this time.



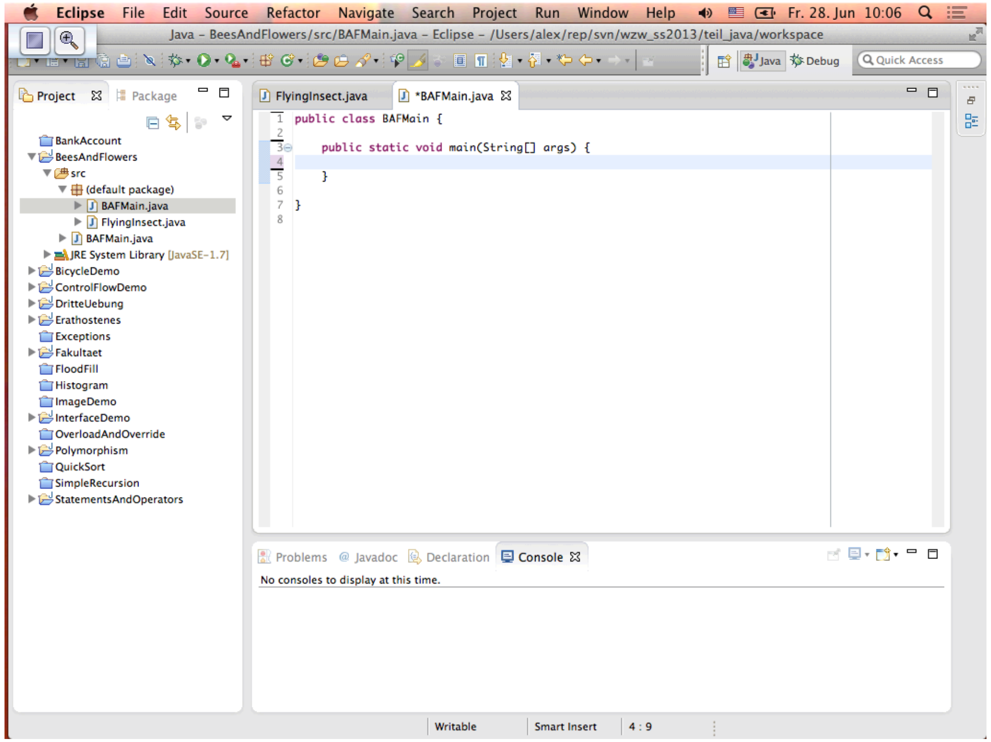
```
public class FlyingInsect {

    void flySlow()

}
```

No consoles to display at this time.



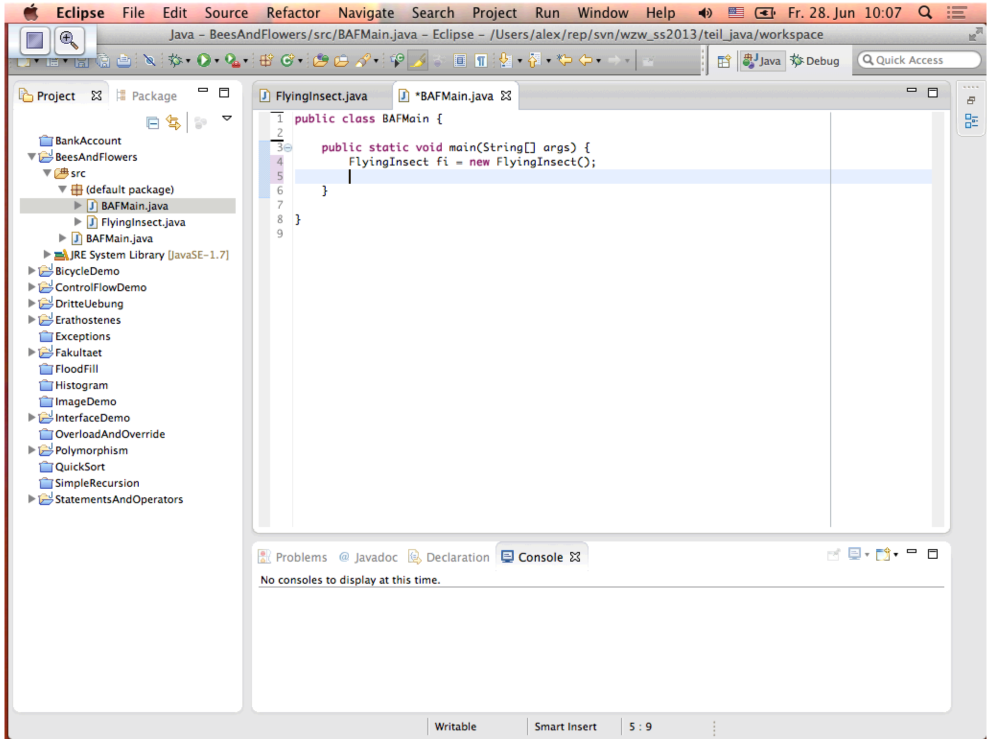
```
public class FlyingInsect {

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

}
```

No consoles to display at this time.



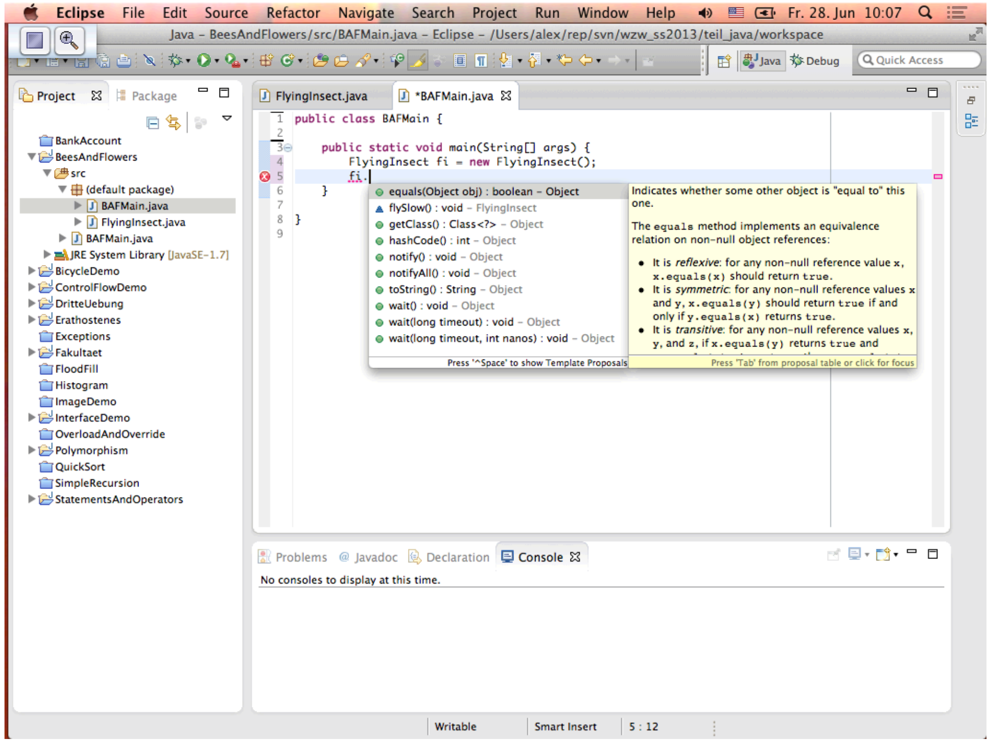
```
public class BAFMain {

    public static void main(String[] args) {

    }

}
```

No consoles to display at this time.



```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        
    }

}
```

No consoles to display at this time.



```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.
    }

}
```

- equals(Object obj) : boolean – Object
- flySlow() : void – FlyingInsect
- getClass() : Class<?> – Object
- hashCode() : int – Object
- notify() : void – Object
- notifyAll() : void – Object
- toString() : String – Object
- wait() : void – Object
- wait(long timeout) : void – Object
- wait(long timeout, int nanos) : void – Object

Press '^Space' to show Template Proposals

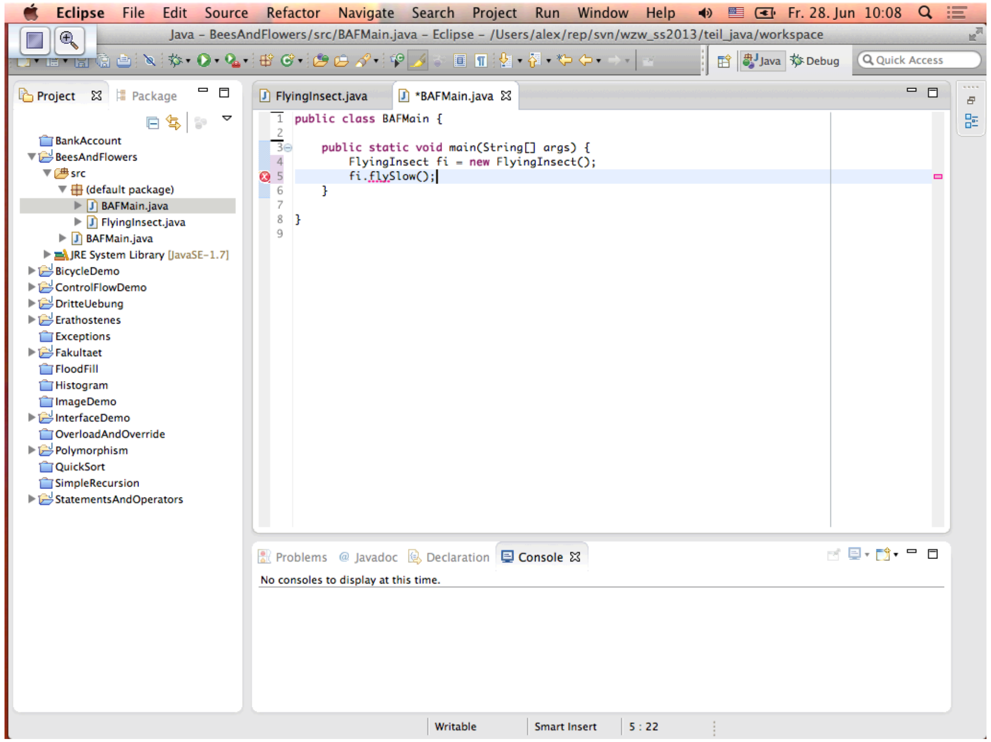
Indicates whether some other object is "equal to" this one.

The `equals` method implements an equivalence relation on non-null object references:

- It is *reflexive*: for any non-null reference value `x`, `x.equals(x)` should return `true`.
- It is *symmetric*: for any non-null reference values `x` and `y`, `x.equals(y)` should return `true` if and only if `y.equals(x)` returns `true`.
- It is *transitive*: for any non-null reference values `x`, `y`, and `z`, if `x.equals(y)` returns `true` and

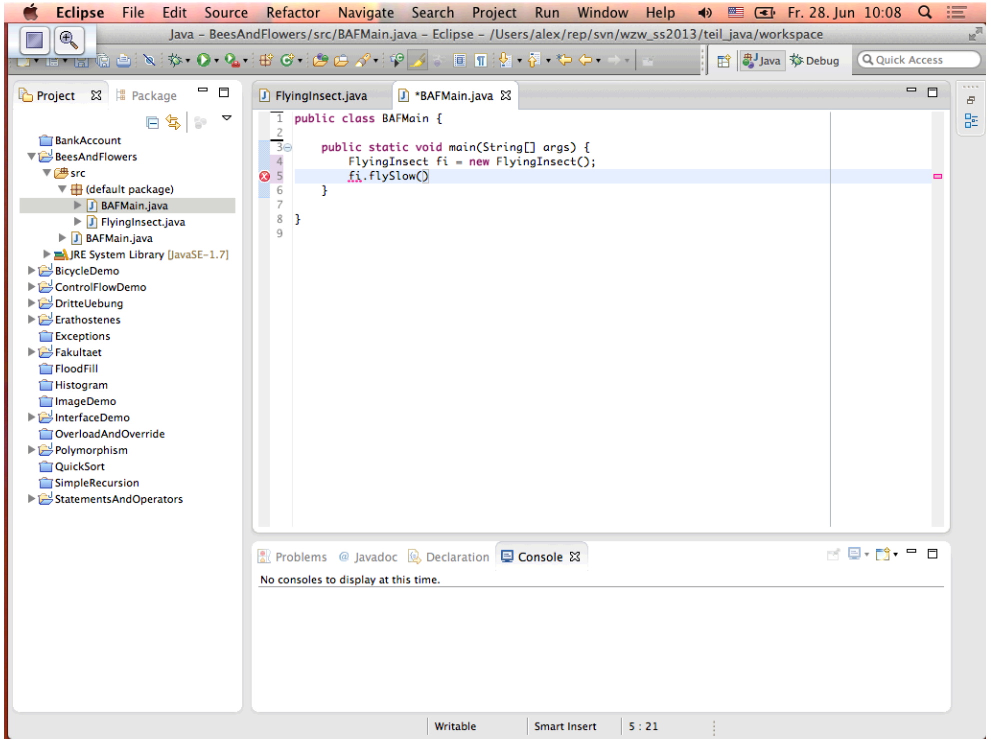
Press 'Tab' from proposal table or click for focus

No consoles to display at this time.



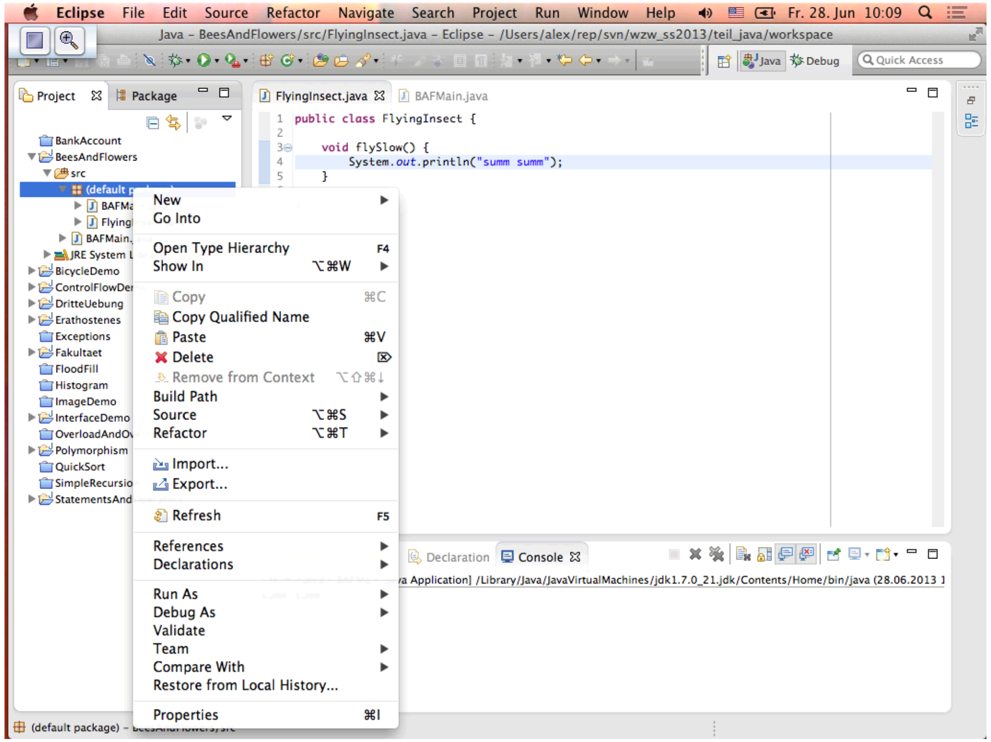
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();
    }

}
```

No consoles to display at this time.



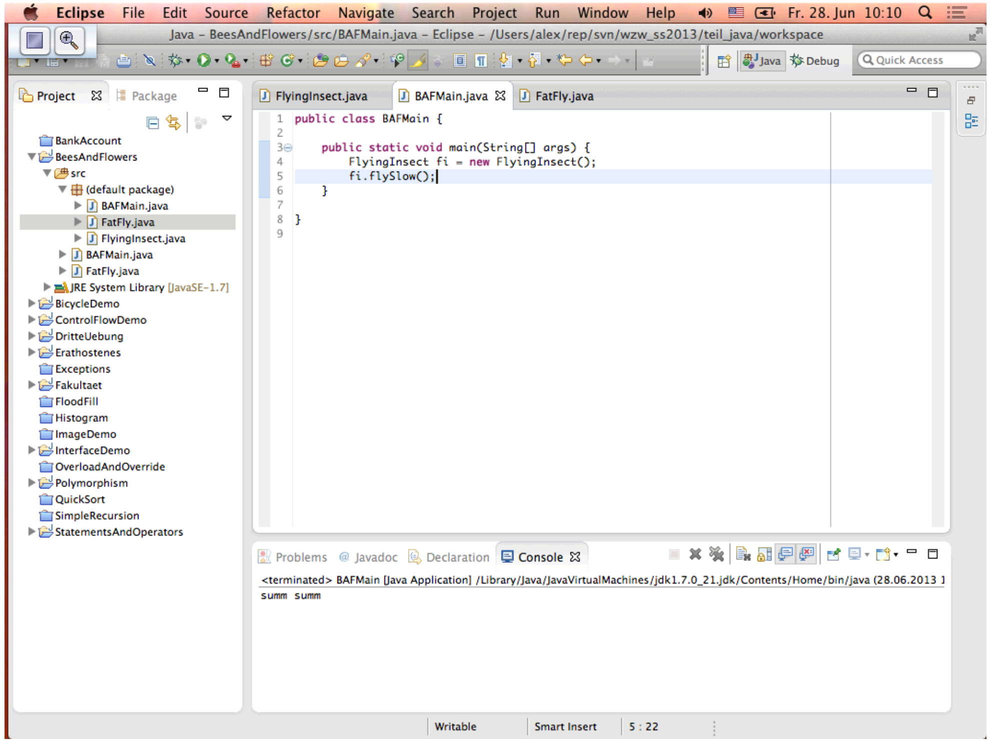
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow()
    }

}
```

No consoles to display at this time.



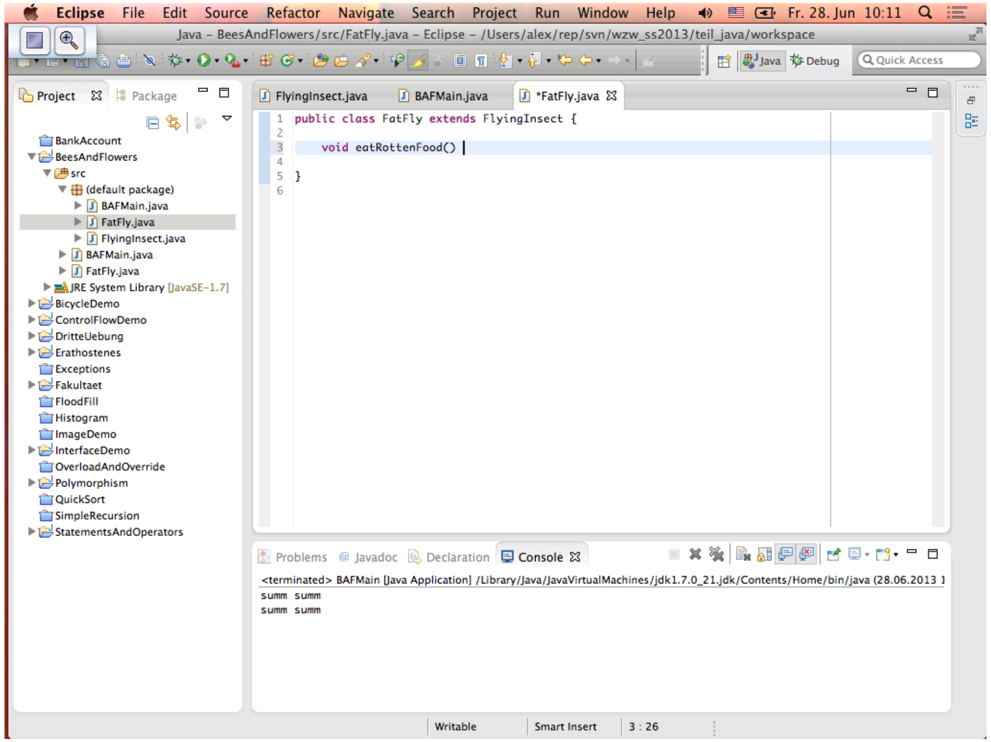
```
public class FlyingInsect {

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

```
a Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
```



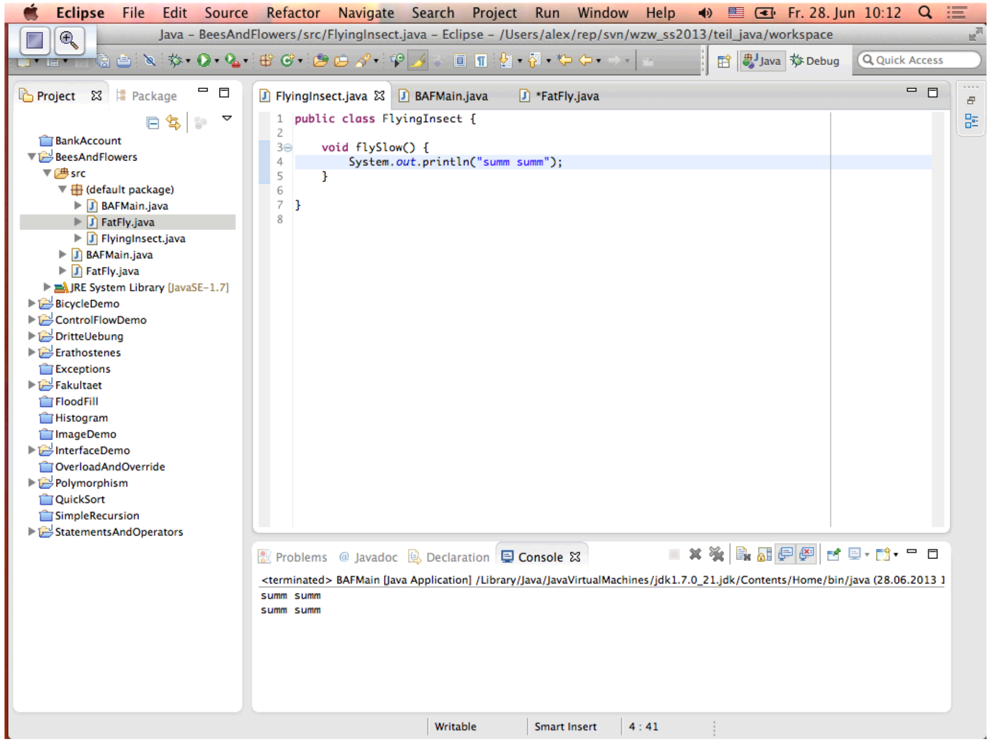
```
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
```



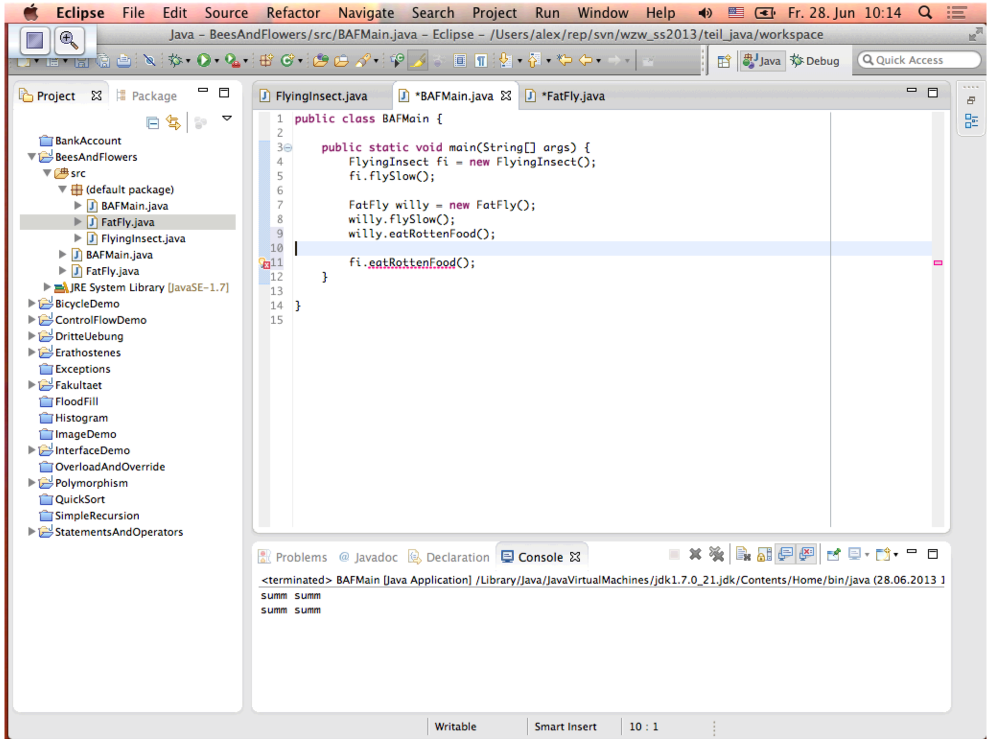
```
public class FatFly extends FlyingInsect {

    void eatRottenFood() 

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```



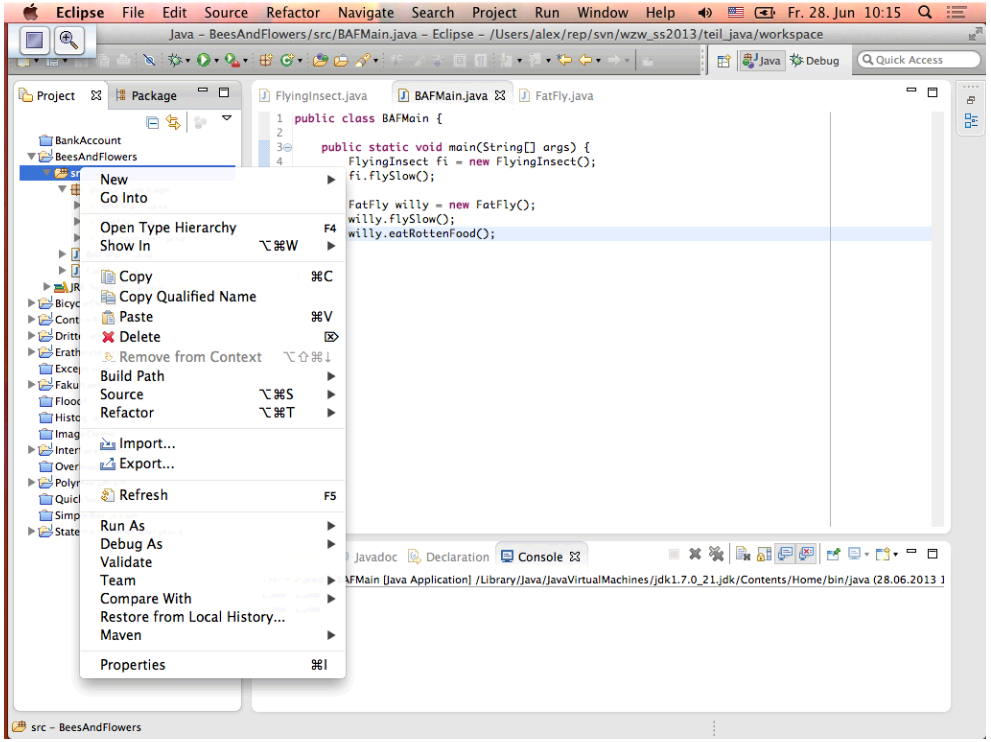
```
public class FlyingInsect {

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

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```



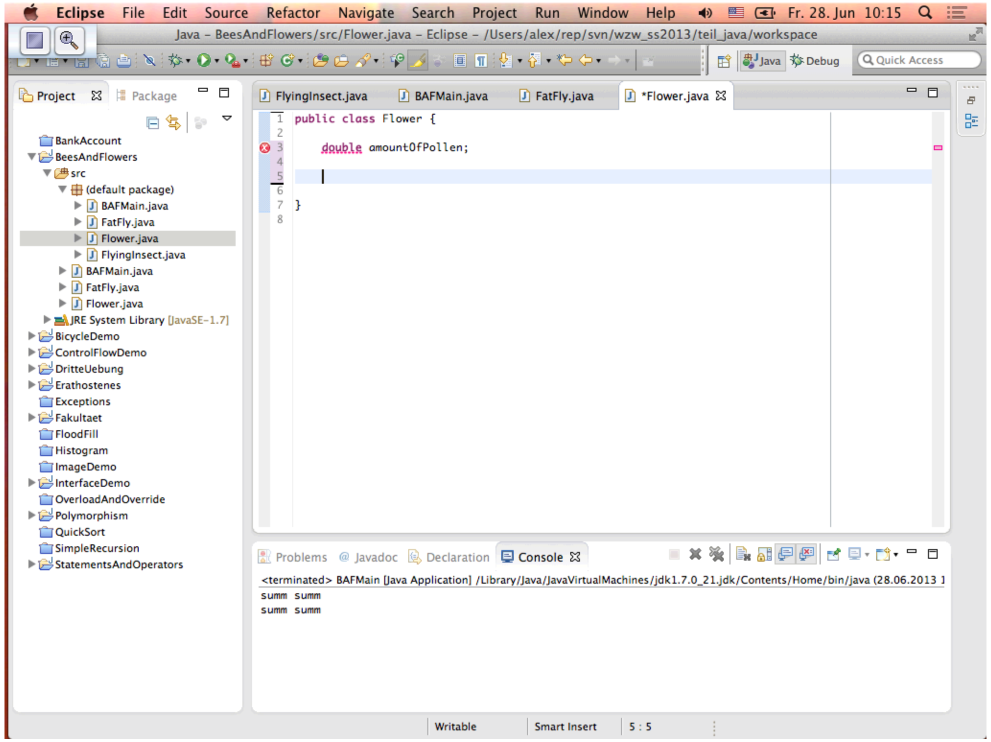
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        fi.eatRottenFood();
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```



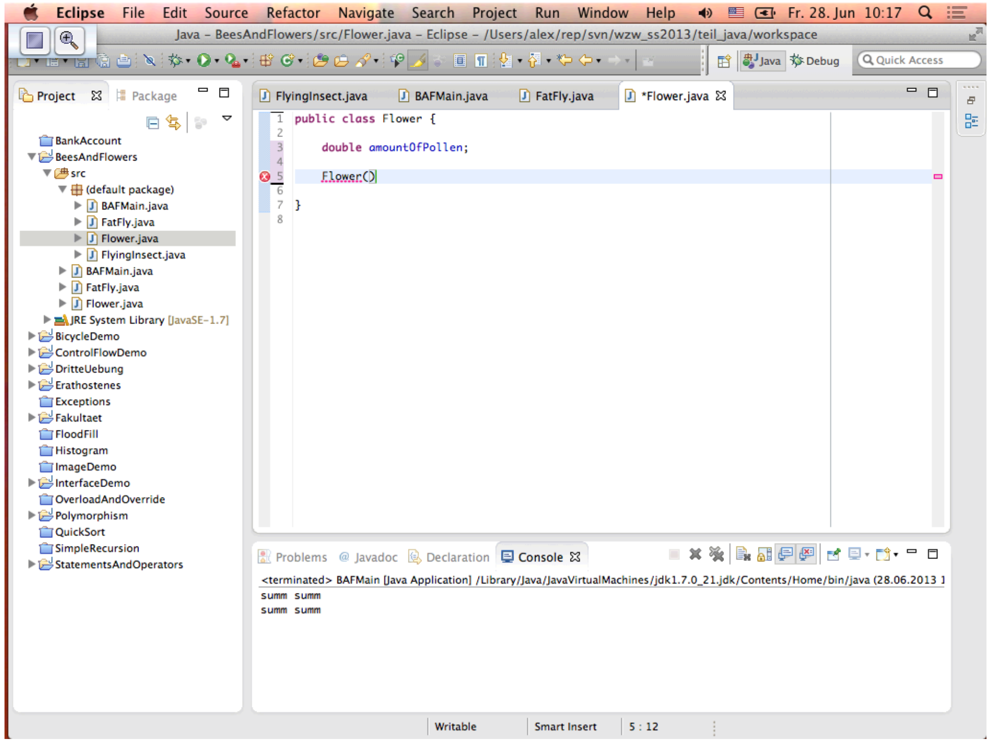
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
```



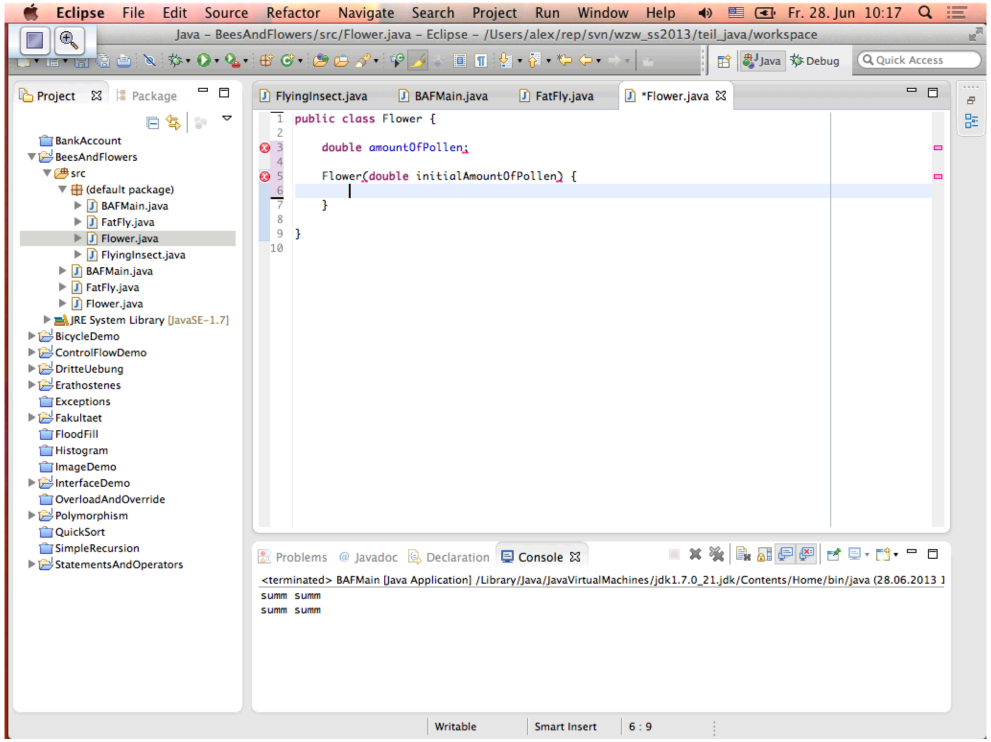
```java
public class Flower {

    double amountOfPollen;



}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```



```java
public class Flower {

    double amountOfPollen;

    Flower()

}
```

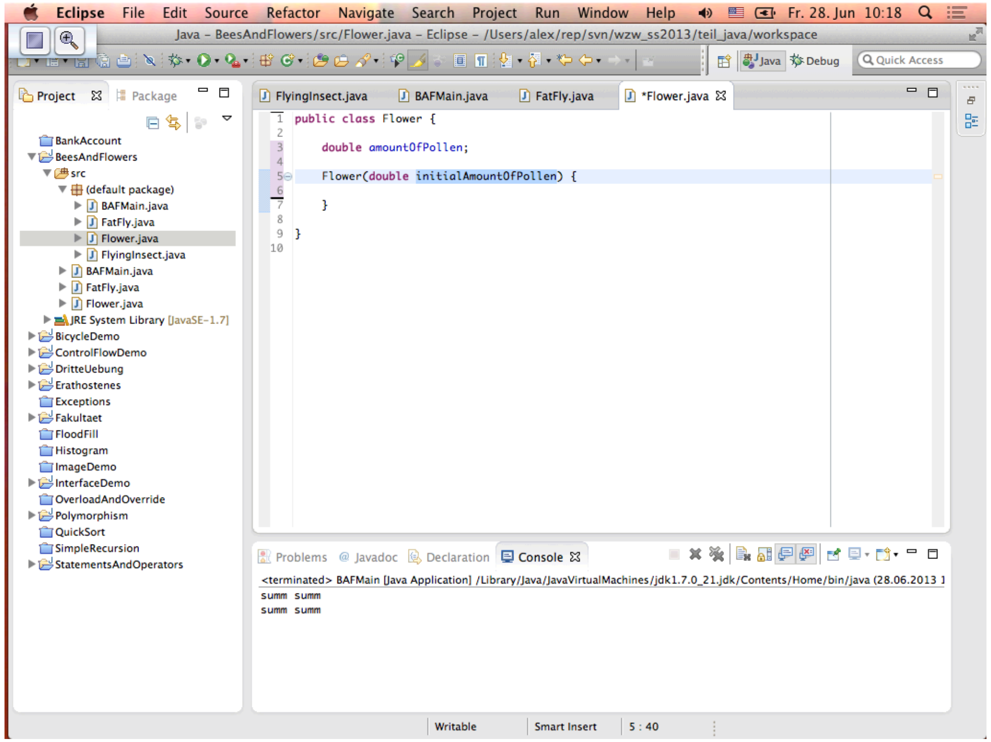
```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```



```
public class Flower {

    double amountOfPollen;

    Flower(double initialAmountOfPollen) {

    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```

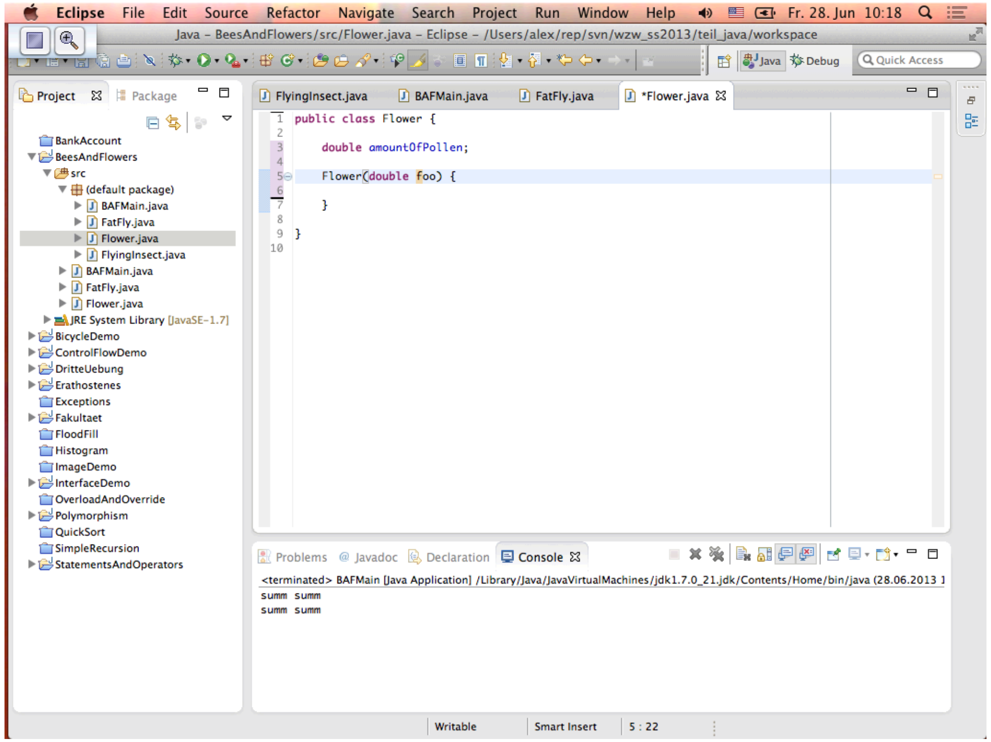
Writable | Smart Insert | 6 : 9


Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
Fr. 28. Jun 10:18
Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
Java
Debug
Quick Access
Project
Package
FlyingInsect.java
BAFMain.java
FatFly.java
*Flower.java
Problems
Javadoc
Declaration
Console

```
public class Flower {

    double amountOfPollen;

    Flower(double initialAmountOfPollen) {

    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```

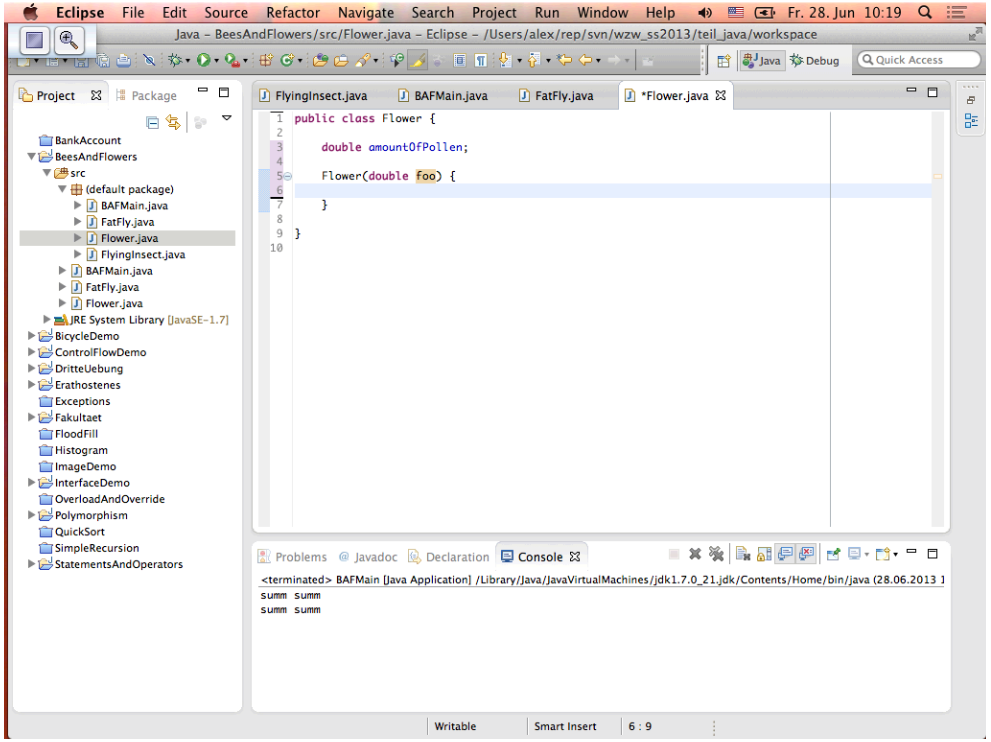
Writable | Smart Insert | 5 : 40


Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
Fr. 28. Jun 10:18
Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
Java
Debug
Quick Access
Project
Package
FlyingInsect.java
BAFMain.java
FatFly.java
*Flower.java
Problems
Javadoc
Declaration
Console

```
public class Flower {

    double amountOfPollen;

    Flower(double foo) {

    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```

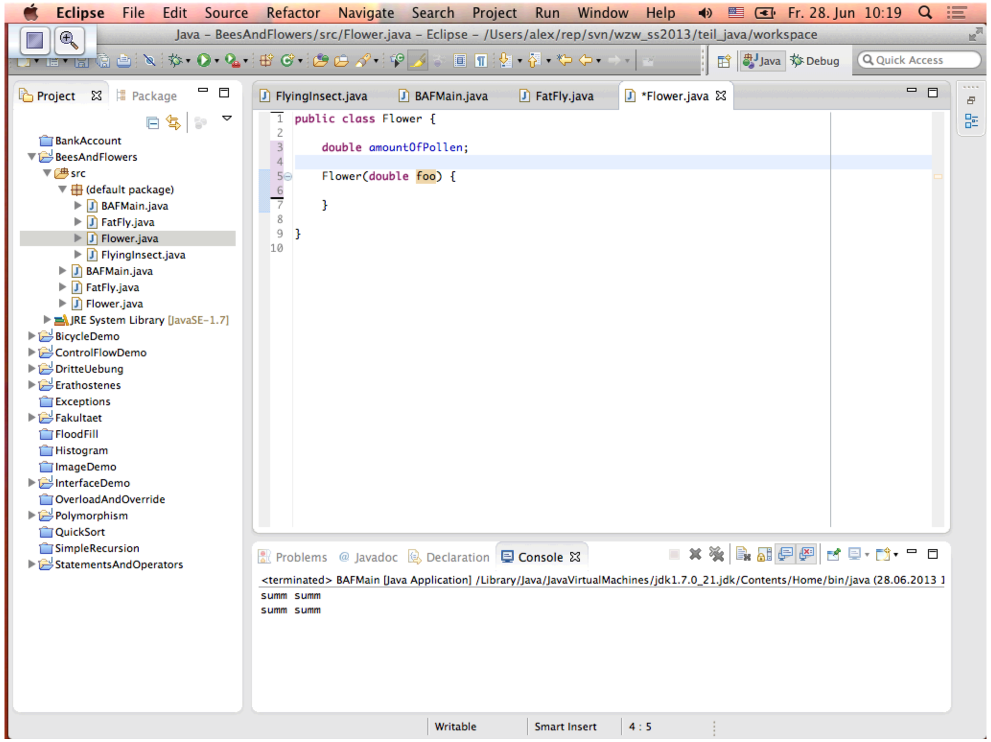
Writable | Smart Insert | 5 : 22


Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
Fr. 28. Jun 10:19
Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
Java
Debug
Quick Access
Project
Package
FlyingInsect.java
BAFMain.java
FatFly.java
*Flower.java
Problems
Javadoc
Declaration
Console

```
public class Flower {

    double amountOfPollen;

    Flower(double foo) {

    }

}
```

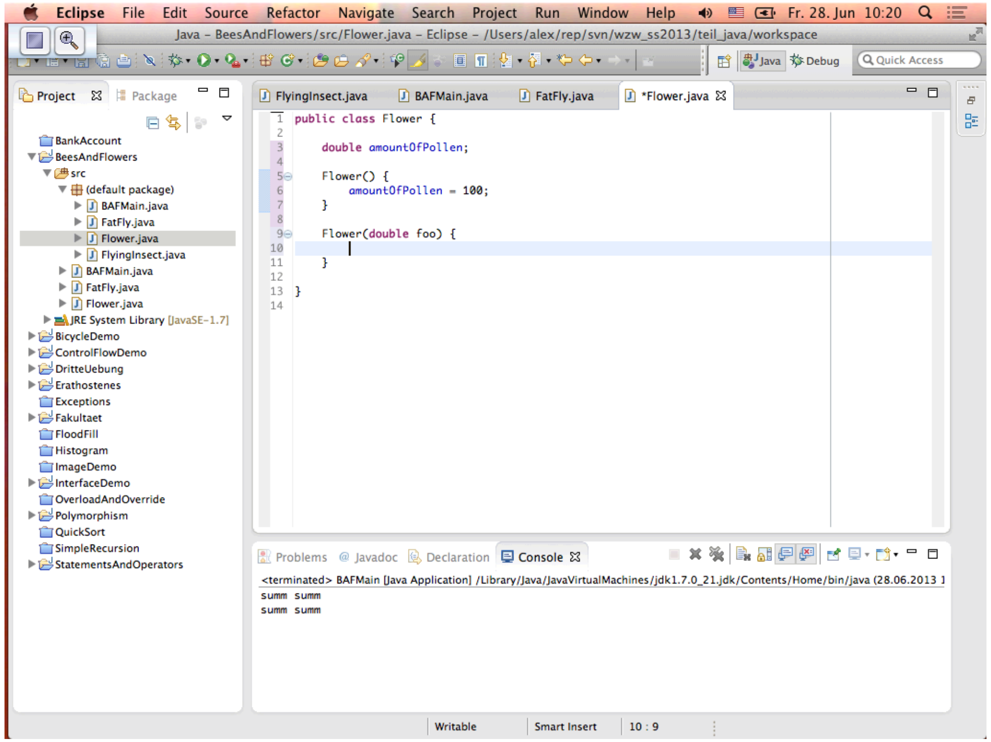
```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```

Writable | Smart Insert | 6 : 9



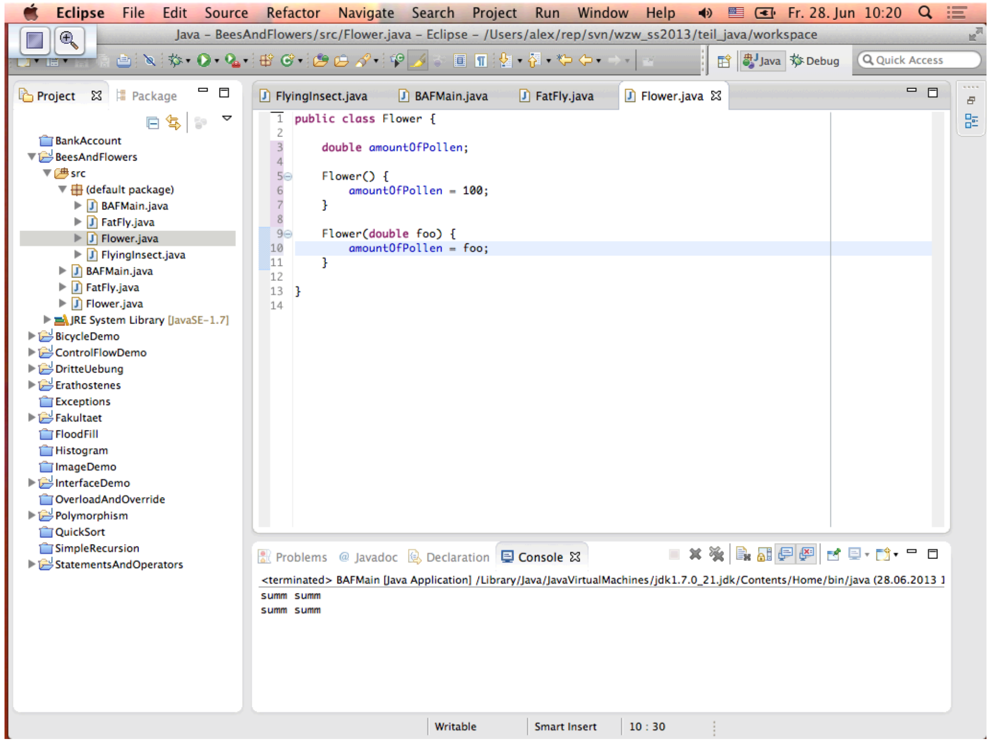
```
public class Flower {

    double amountOfPollen;

    Flower(double foo) {

    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```


Fr. 28. Jun 10:20
Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
*Flower.java
Writable
Smart Insert
10 : 9

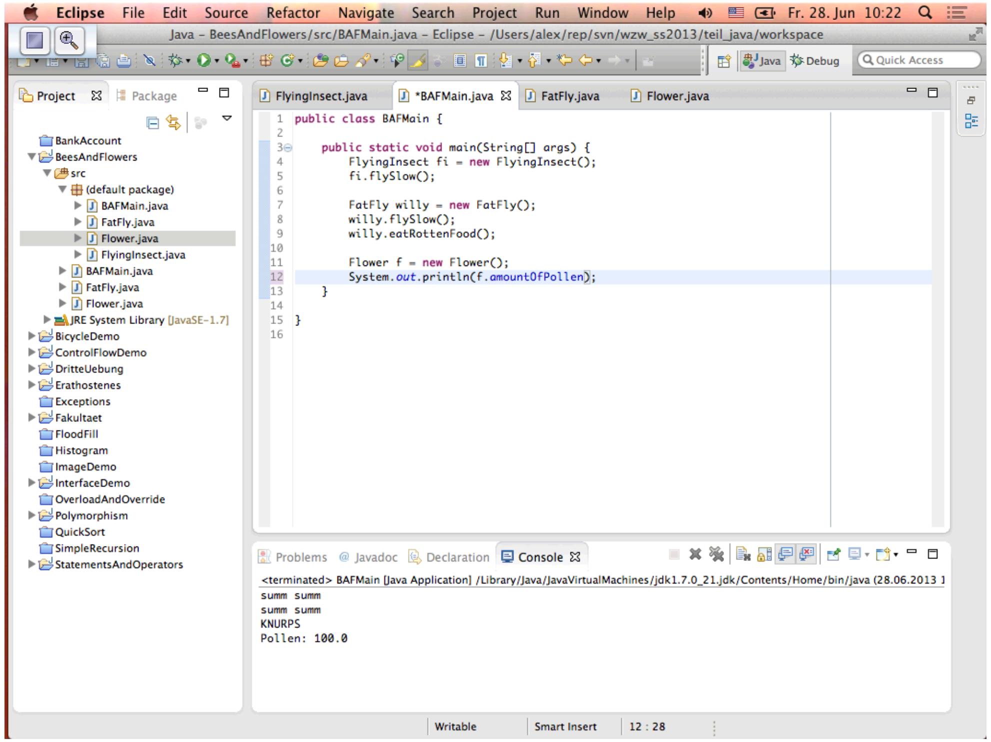
```
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```


Fr. 28. Jun 10:20
Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
Flower.java
Writable
Smart Insert
10 : 30

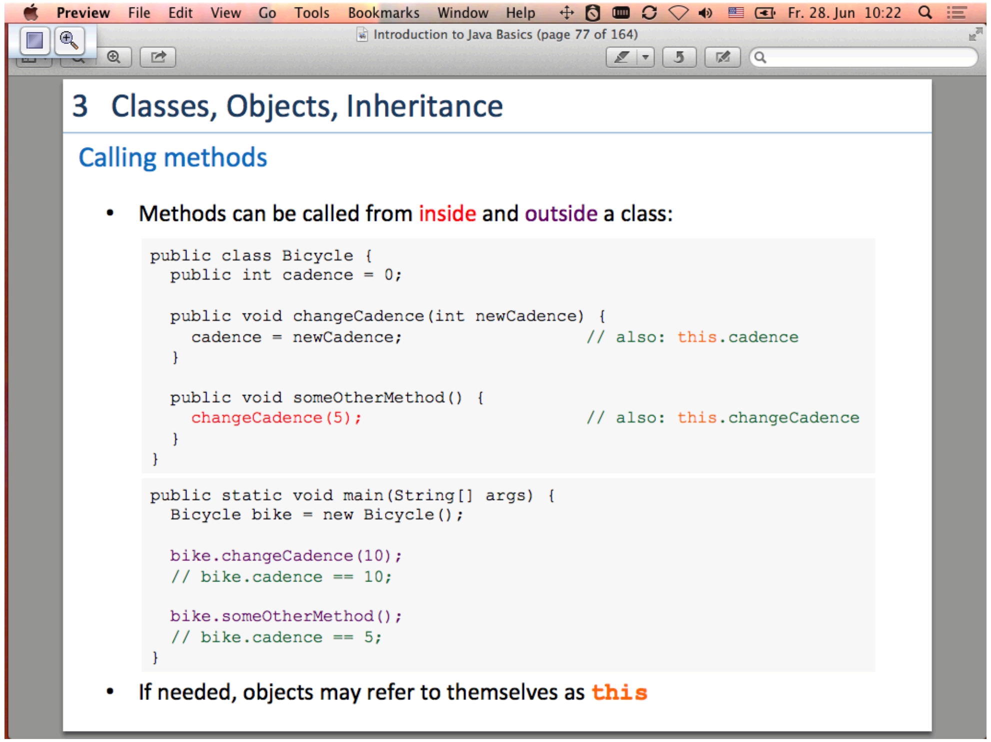
```
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
```


Fr. 28. Jun 10:22
Java – BeesAndFlowers/src/BAFMain.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
*BAFMain.java
Writable
Smart Insert
12 : 28

```
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower();
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
Pollen: 100.0
```


Preview File Edit View Go Tools Bookmarks Window Help Fr. 28. Jun 10:22
Introduction to Java Basics (page 77 of 164)

3 Classes, Objects, Inheritance

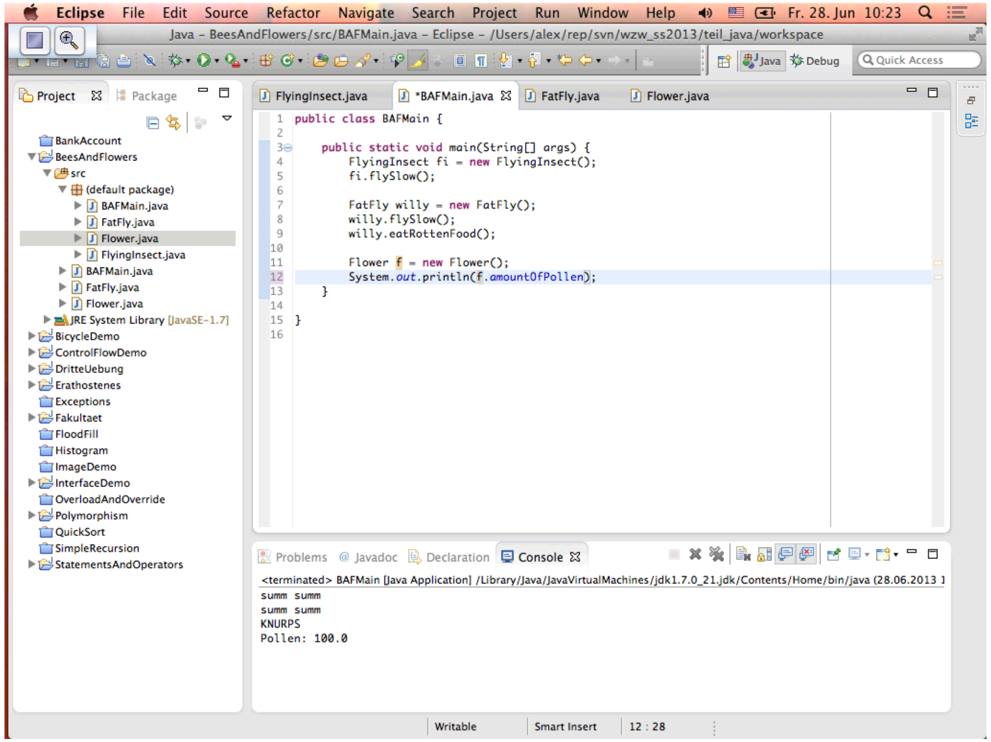
Calling methods

- Methods can be called from inside and outside a class:

```
public class Bicycle {
  public int cadence = 0;

  public void changeCadence(int newCadence)  {
    cadence = newCadence;                     // also: this.cadence
  }

  public void someOtherMethod() {
    changeCadence(5);                         // also: this.changeCadence
  }
}
```

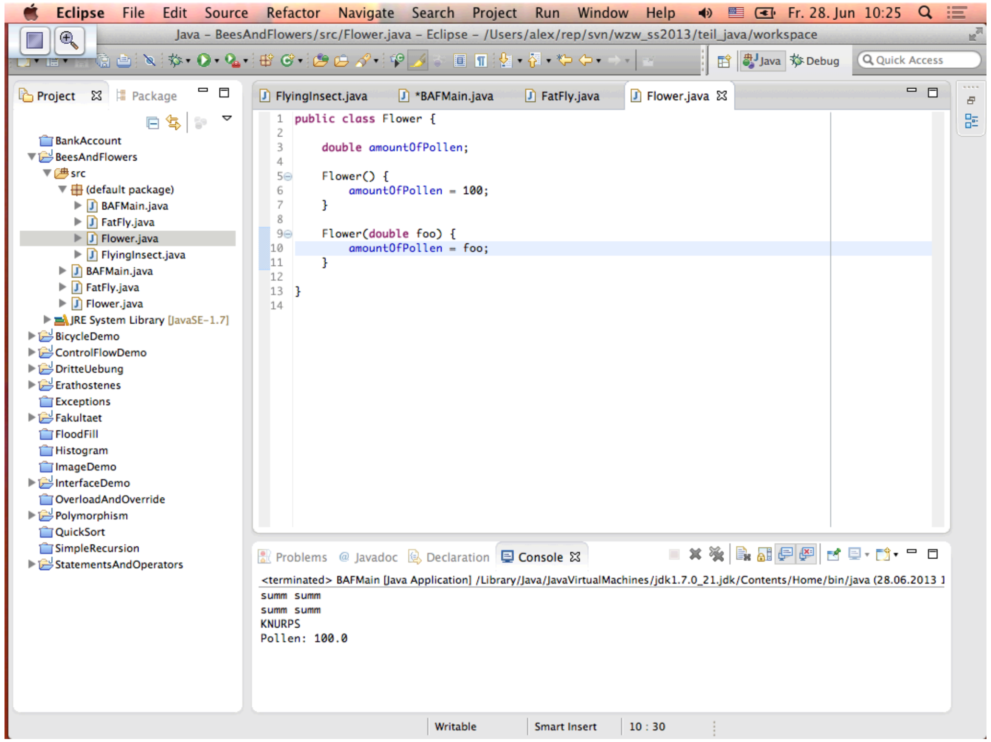
```
public static void main(String[] args) {
  Bicycle bike = new Bicycle();

  bike.changeCadence(10);
  // bike.cadence == 10;

  bike.someOtherMethod();
  // bike.cadence == 5;
}
```

- If needed, objects may refer to themselves as `this`



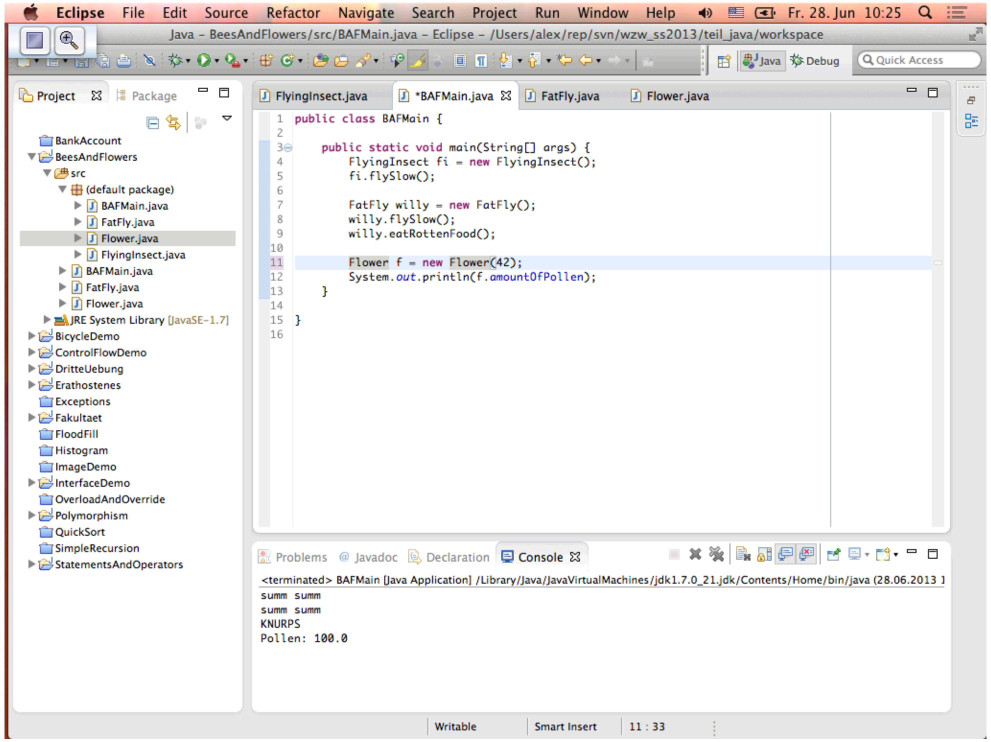
```
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower();
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
Pollen: 100.0
```



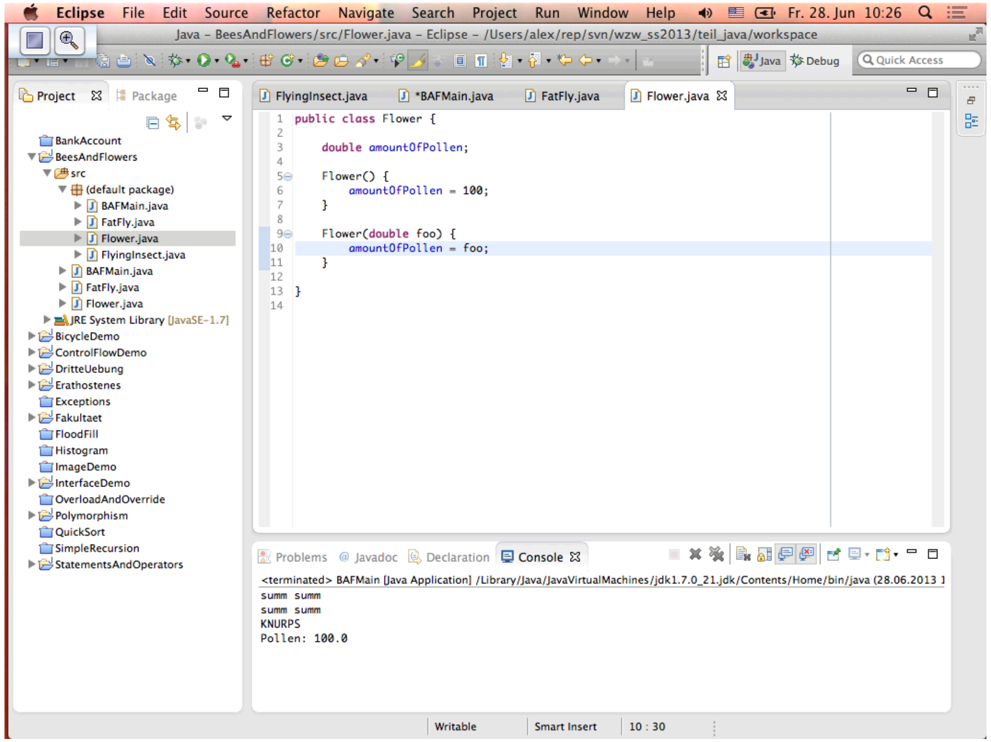
```
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
Pollen: 100.0
```



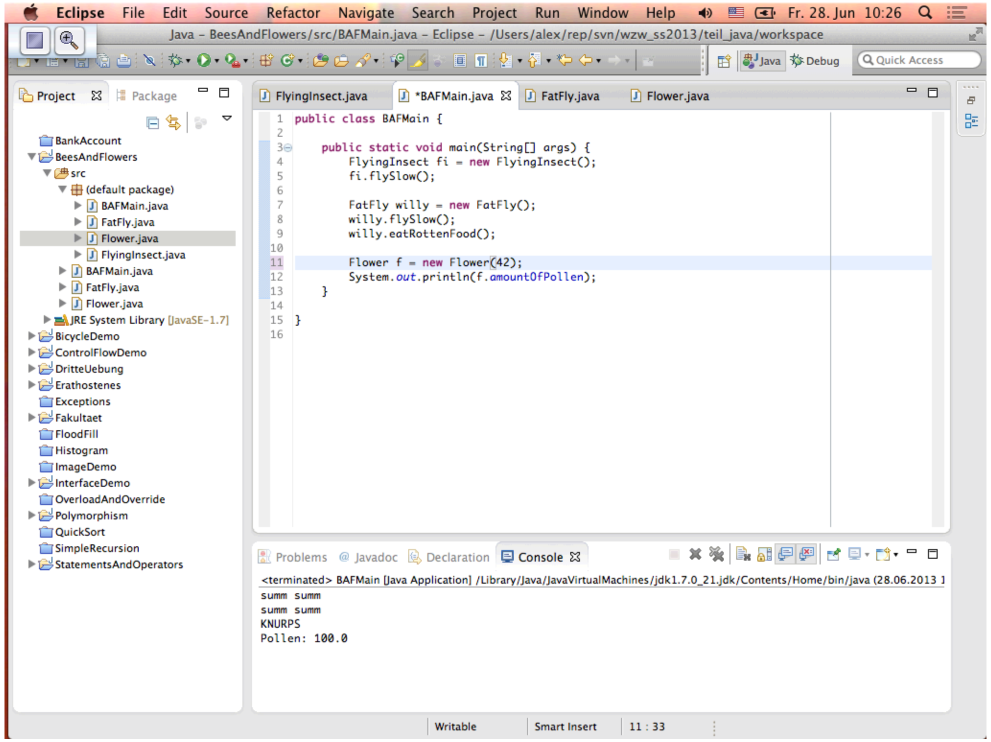
```
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
Pollen: 100.0
```



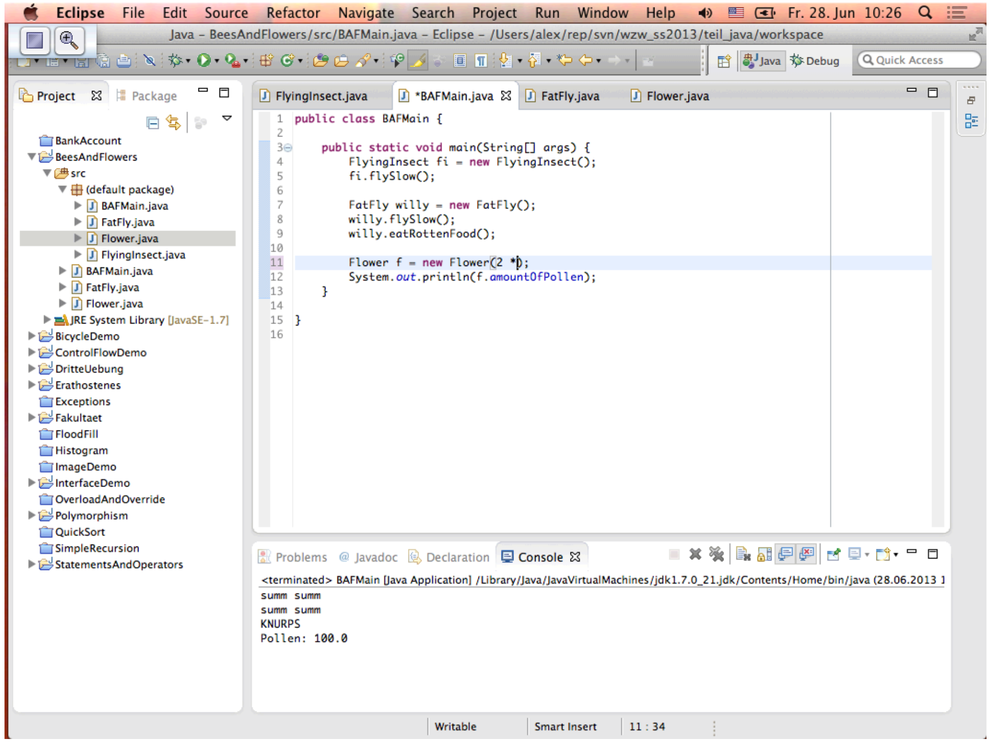
```
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
Pollen: 100.0
```



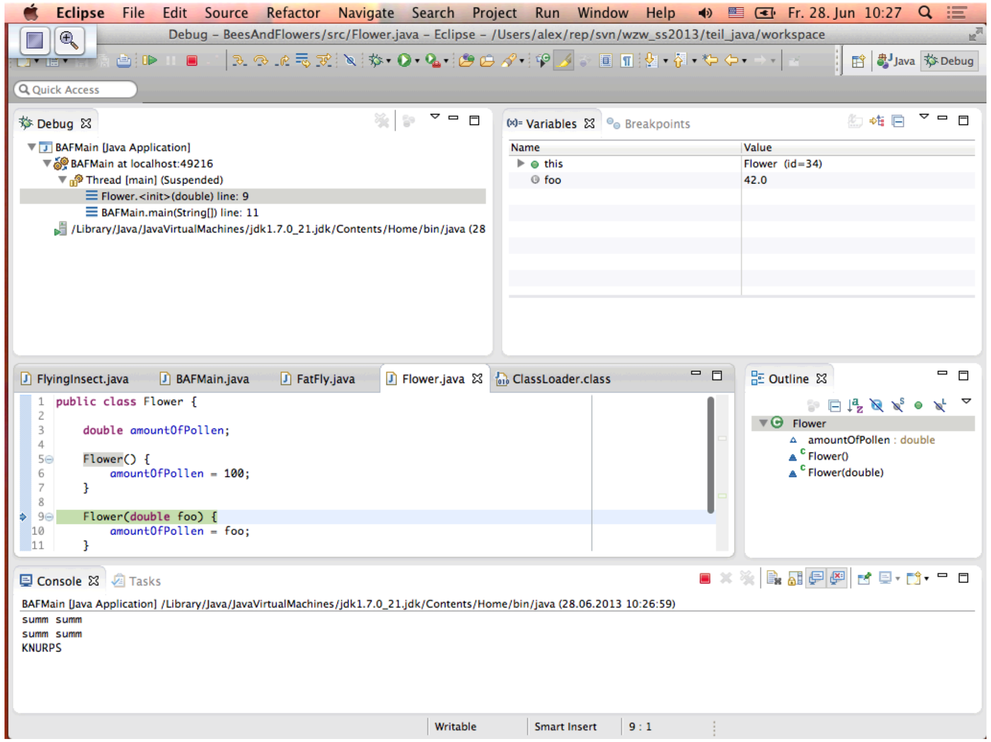
```
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
Pollen: 100.0
```



```
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(2 *);
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
Pollen: 100.0
```

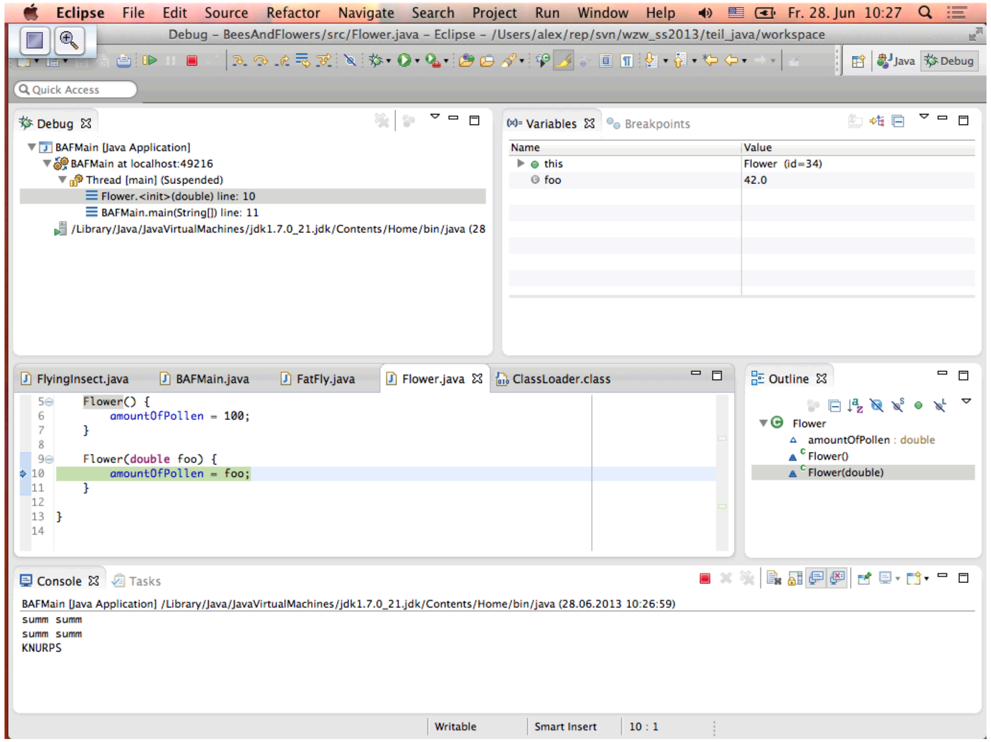


```
BAFMain [Java Application]
  BAFMain at localhost:49216
    Thread [main] (Suspended)
      Flower.<init>(double) line: 9
      BAFMain.main(String[]) line: 11
  /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28
```

Name	Value
this	Flower (id=34)
foo	42.0

```
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }
```

```
BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 10:26:59)
summ summ
summ summ
KNURPS
```


Eclipse File Edit Source Refactor Navigate Search Project Run Window Help
Fr. 28. Jun 10:27
Quick Access
Java
Debug
Variables
Breakpoints
Outline
Tasks

Debug – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace

```
BAFMain [Java Application]
  BAFMain at localhost:49216
    Thread [main] (Suspended)
      Flower.<init>(double) line: 10
      BAFMain.main(String[]) line: 11
    /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28
```

Name	Value
this	Flower (id=34)
foo	42.0

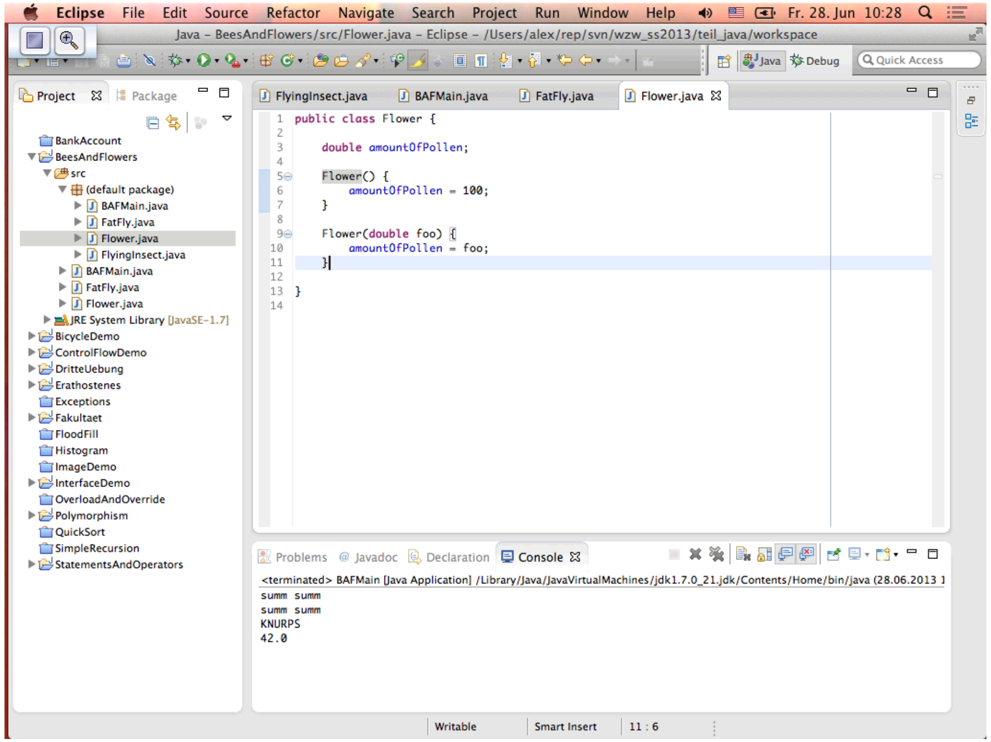
FlyingInsect.java | BAFMain.java | FatFly.java | Flower.java | ClassLoader.class

```java
    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

}
```

```
Flower
  amountOfPollen : double
  Flower()
  Flower(double)
```

```
BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 10:26:59)
summ summ
summ summ
KNURPS
```

Writable | Smart Insert | 10 : 1



Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace

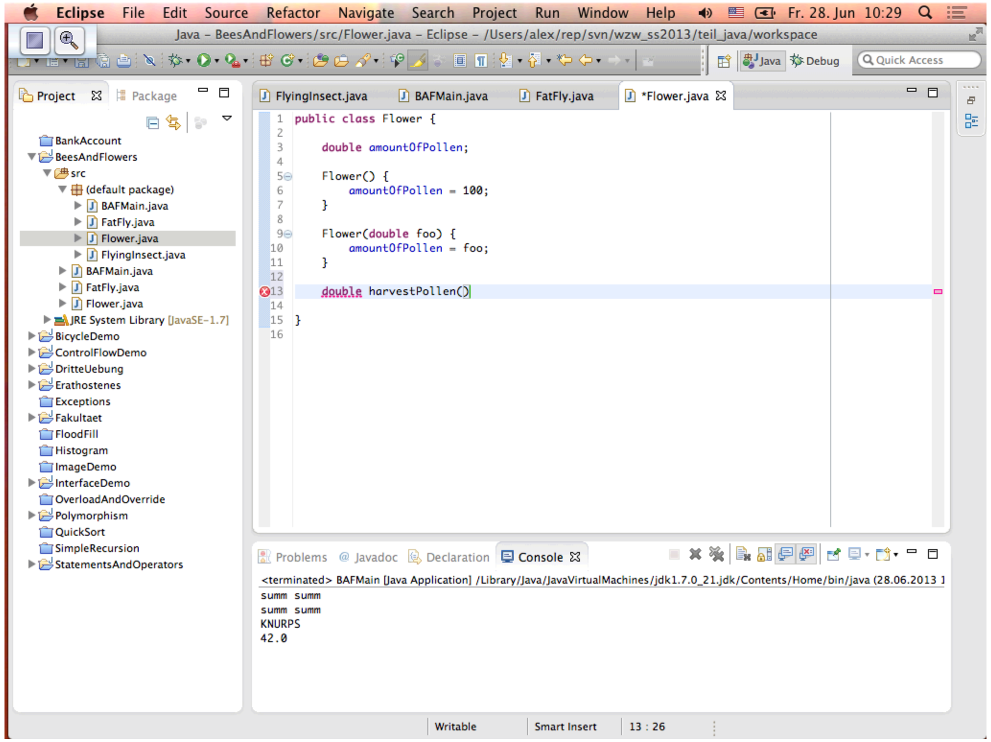
```
BankAccount
BeesAndFlowers
  src
    (default package)
      BAFMain.java
      FatFly.java
      Flower.java
      FlyingInsect.java
    BAFMain.java
    FatFly.java
    Flower.java
  JRE System Library [JavaSE-1.7]
BicycleDemo
ControlFlowDemo
DritteUebung
Erathostenes
Exceptions
Fakultaet
FloodFill
Histogram
ImageDemo
InterfaceDemo
OverloadAndOverride
Polymorphism
QuickSort
SimpleRecursion
StatementsAndOperators
```

FlyingInsect.java | BAFMain.java | FatFly.java | Flower.java

```java
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```

Writable | Smart Insert | 11 : 6



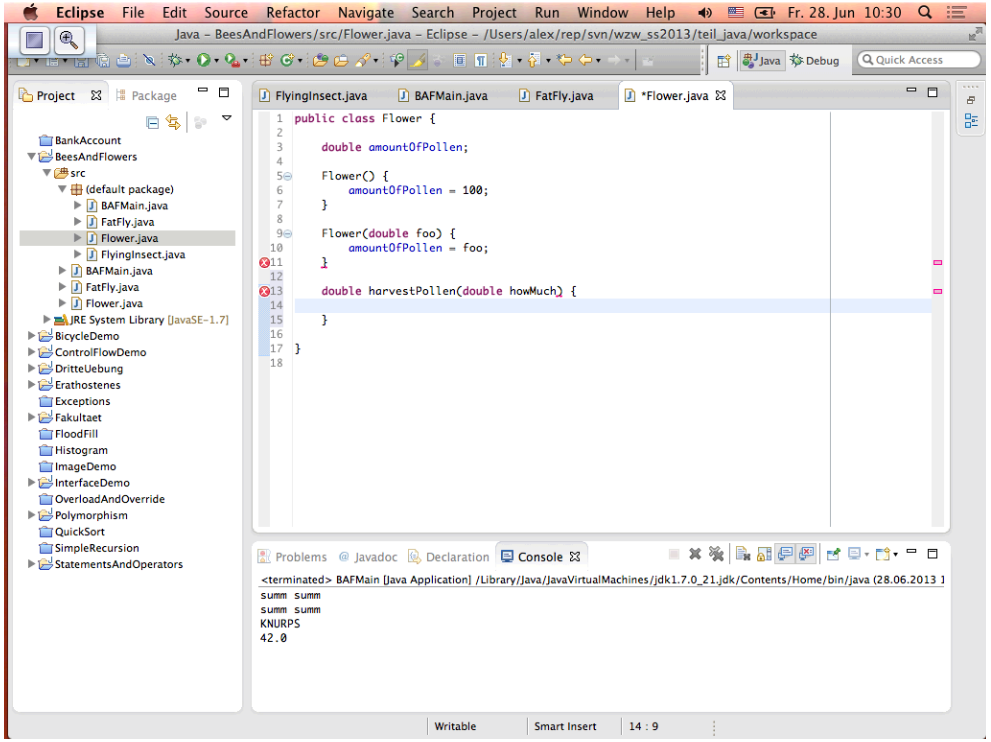
Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace

FlyingInsect.java | BAFMain.java | FatFly.java | *Flower.java

```java
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

    double harvestPollen()

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```

Writable | Smart Insert | 13 : 26



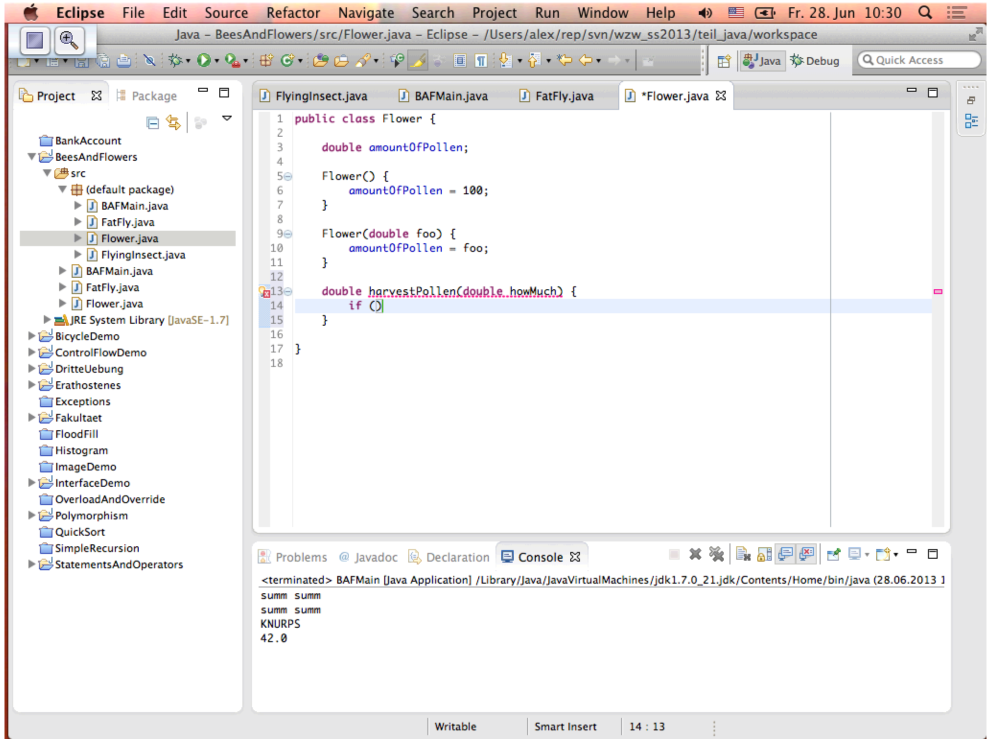
Java – BeesAndFlowers/src/Flower.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace

FlyingInsect.java | BAFMain.java | FatFly.java | *Flower.java

```java
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

    double harvestPollen(double howMuch) {

    }

}
```

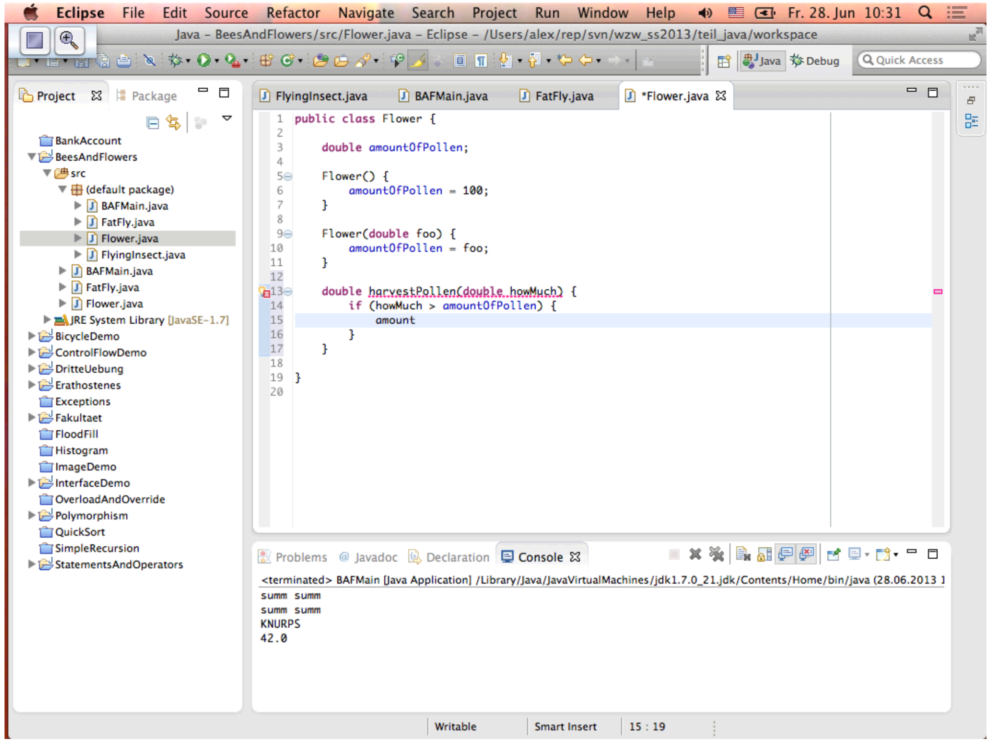
```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```

Writable | Smart Insert | 14 : 9



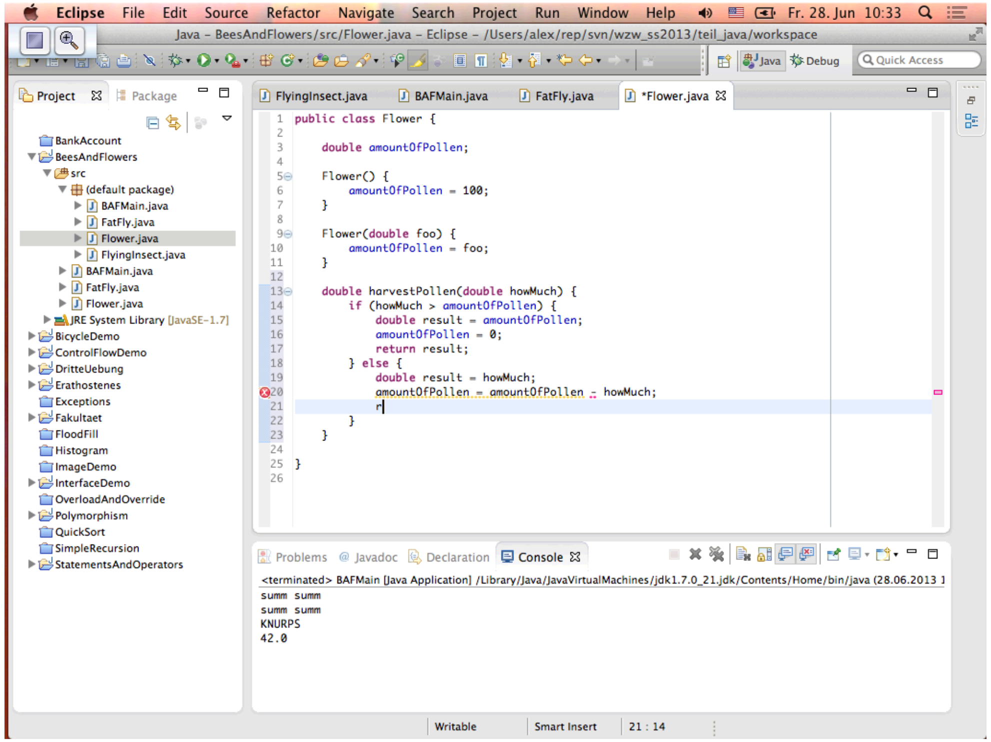
```java
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

    double harvestPollen(double howMuch) {
        if ()
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```



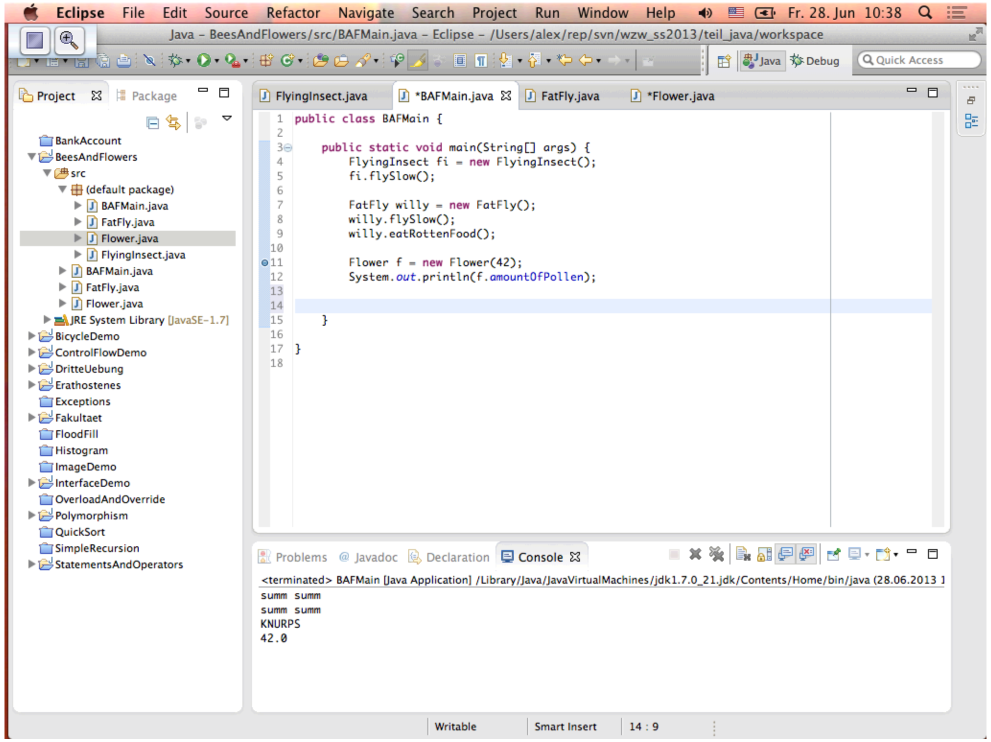
```java
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

    double harvestPollen(double howMuch) {
        if (howMuch > amountOfPollen) {
            amount
        }
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```



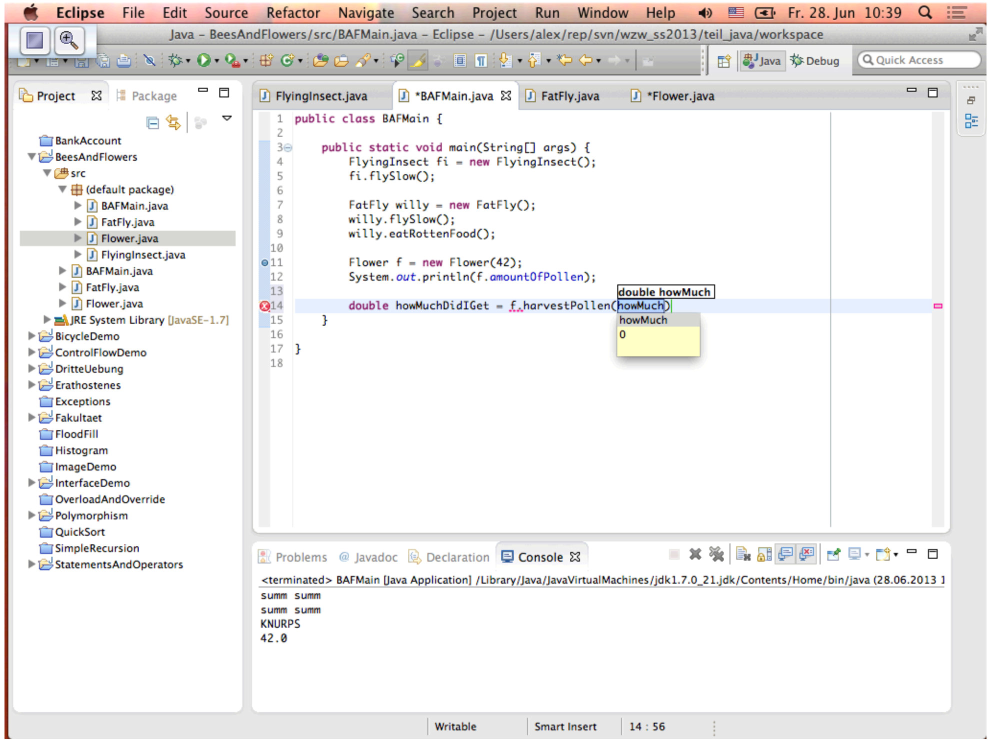
```java
public class Flower {

    double amountOfPollen;

    Flower() {
        amountOfPollen = 100;
    }

    Flower(double foo) {
        amountOfPollen = foo;
    }

    double harvestPollen(double howMuch) {
        if (howMuch > amountOfPollen) {
            double result = amountOfPollen;
            amountOfPollen = 0;
            return result;
        } else {
            double result = howMuch;
            amountOfPollen = amountOfPollen - howMuch;
            r
        }
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```



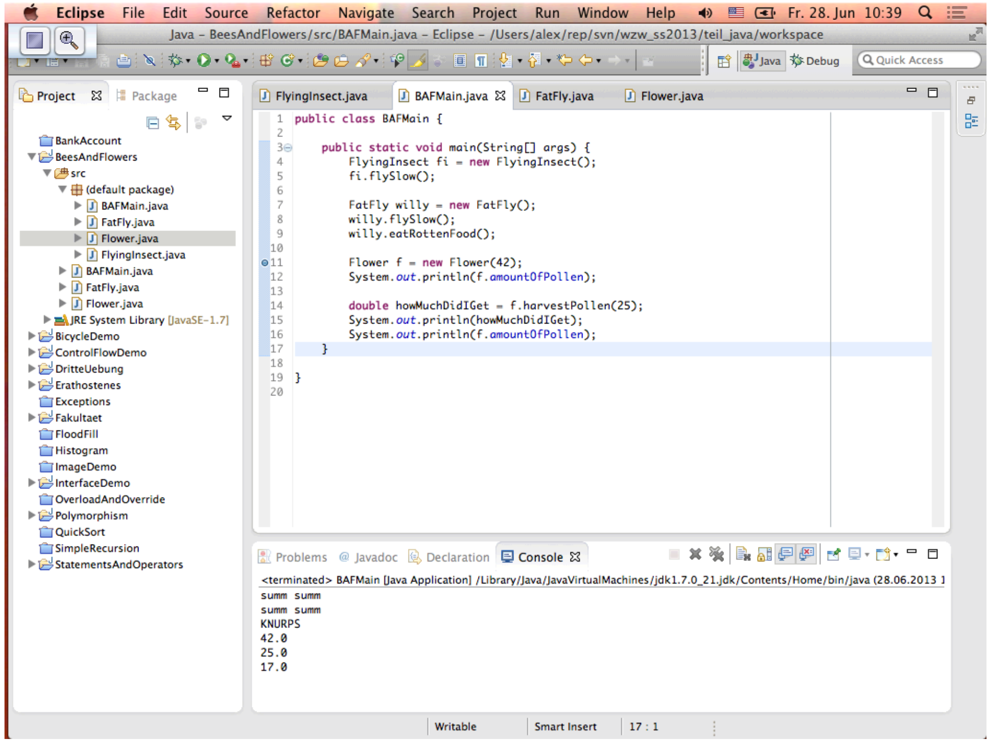
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);


    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```



```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(howMuch)
    }

}
```

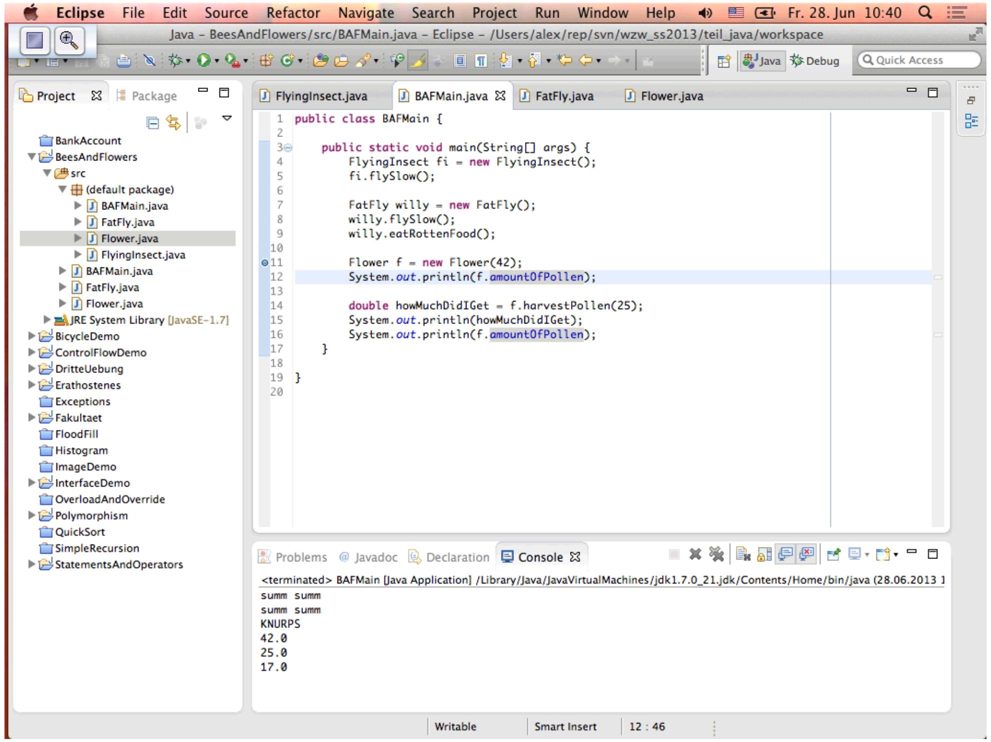
```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
```



Java – BeesAndFlowers/src/BAFMain.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
FlyingInsect.java
BAFMain.java
FatFly.java
Flower.java
Writable
Smart Insert
17 : 1

```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);
    }

}
```

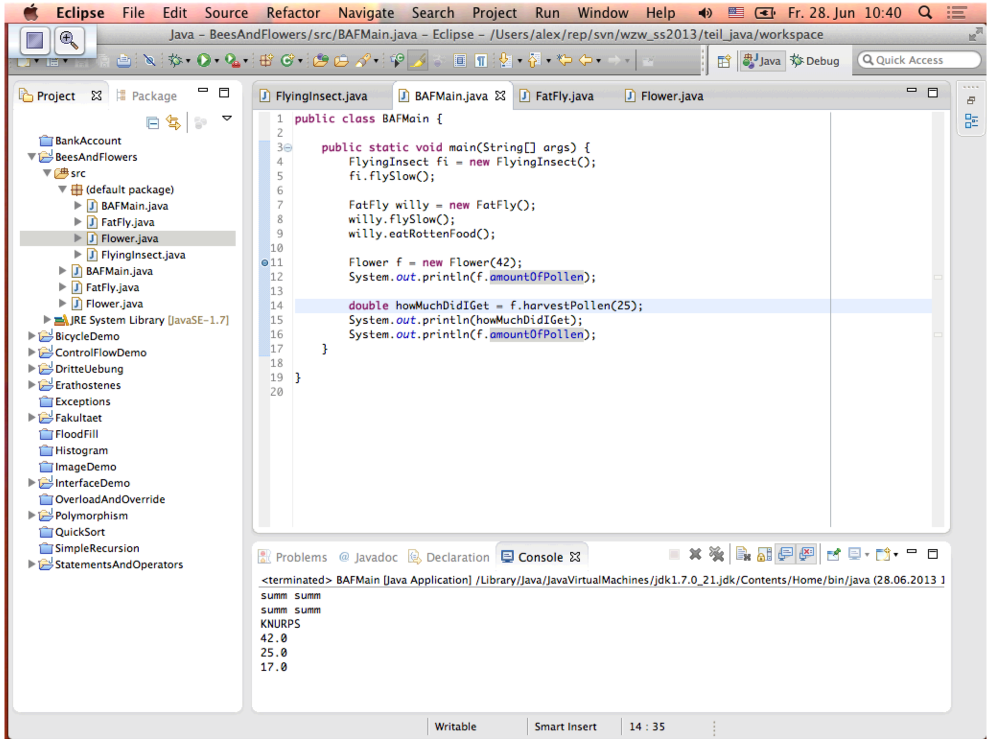
```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
25.0
17.0
```



Java – BeesAndFlowers/src/BAFMain.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
FlyingInsect.java
BAFMain.java
FatFly.java
Flower.java
Writable
Smart Insert
12 : 46

```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);
    }

}
```

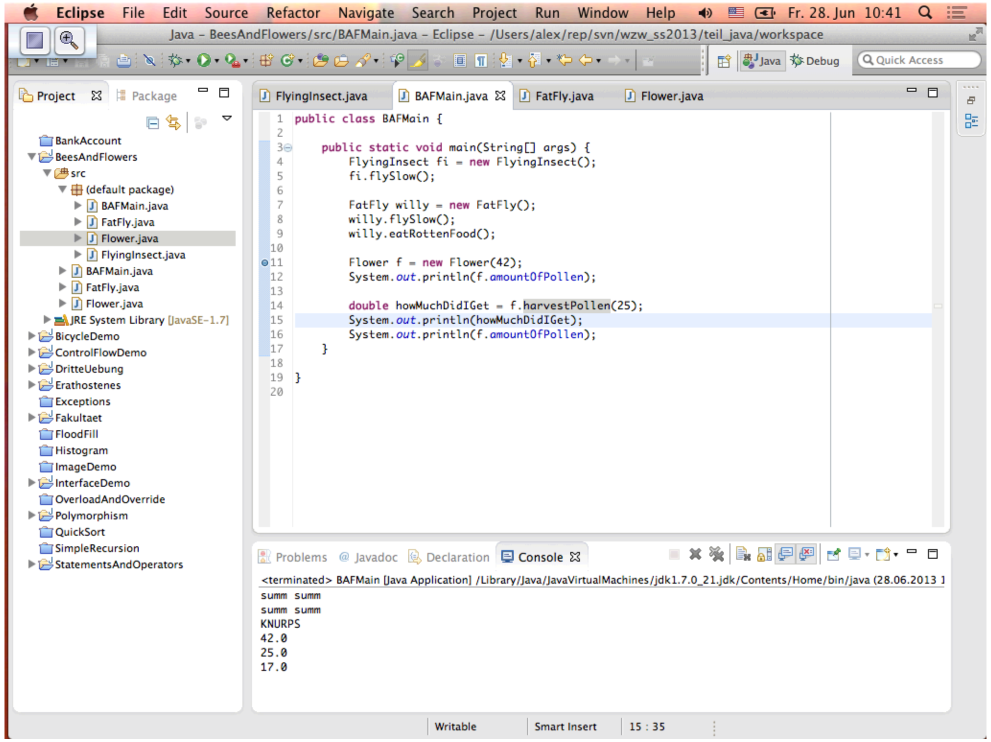
```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
25.0
17.0
```



Java – BeesAndFlowers/src/BAFMain.java – Eclipse – /Users/alex/rep/svn/wzw_ss2013/teil_java/workspace
FlyingInsect.java
BAFMain.java
FatFly.java
Flower.java
Writable
Smart Insert
14 : 35

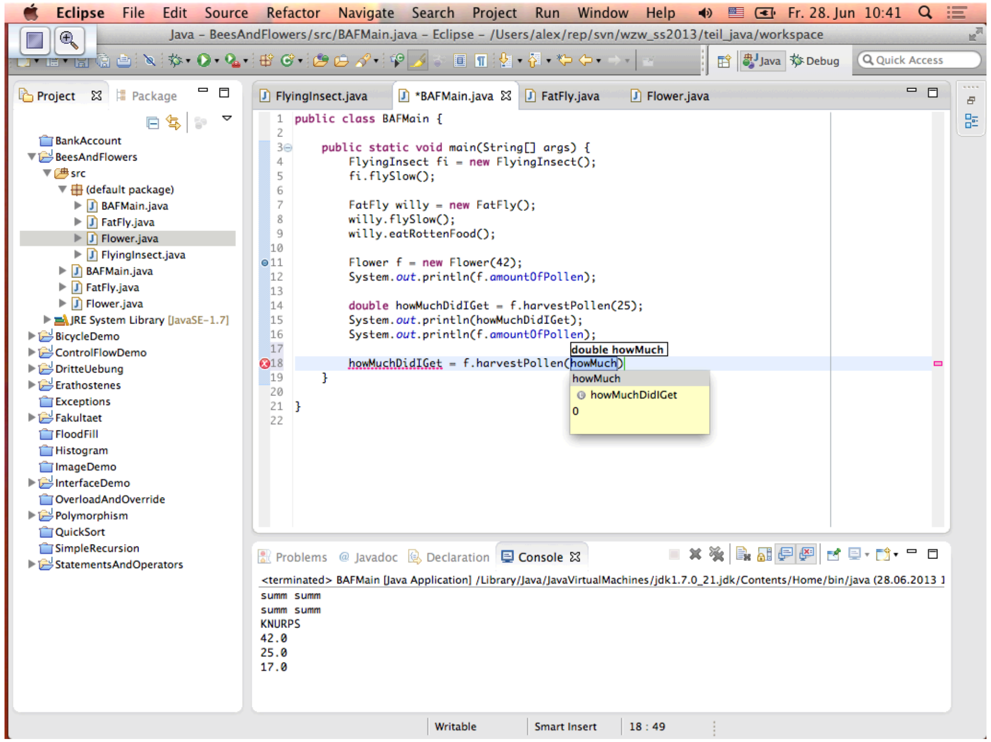
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
25.0
17.0
```



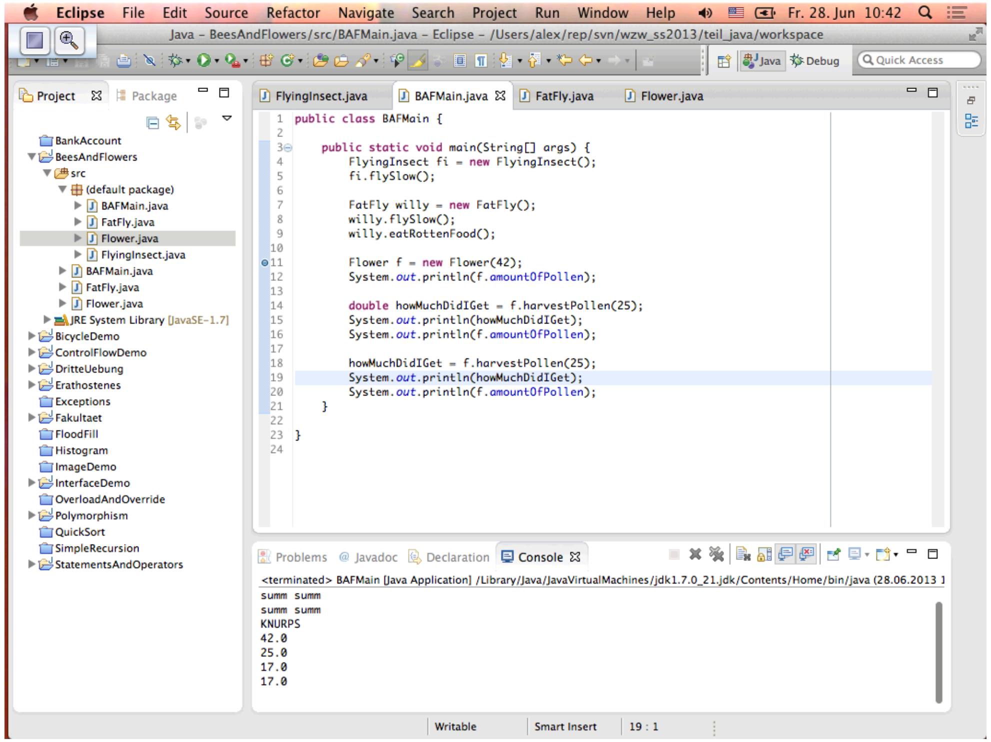
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
25.0
17.0
```



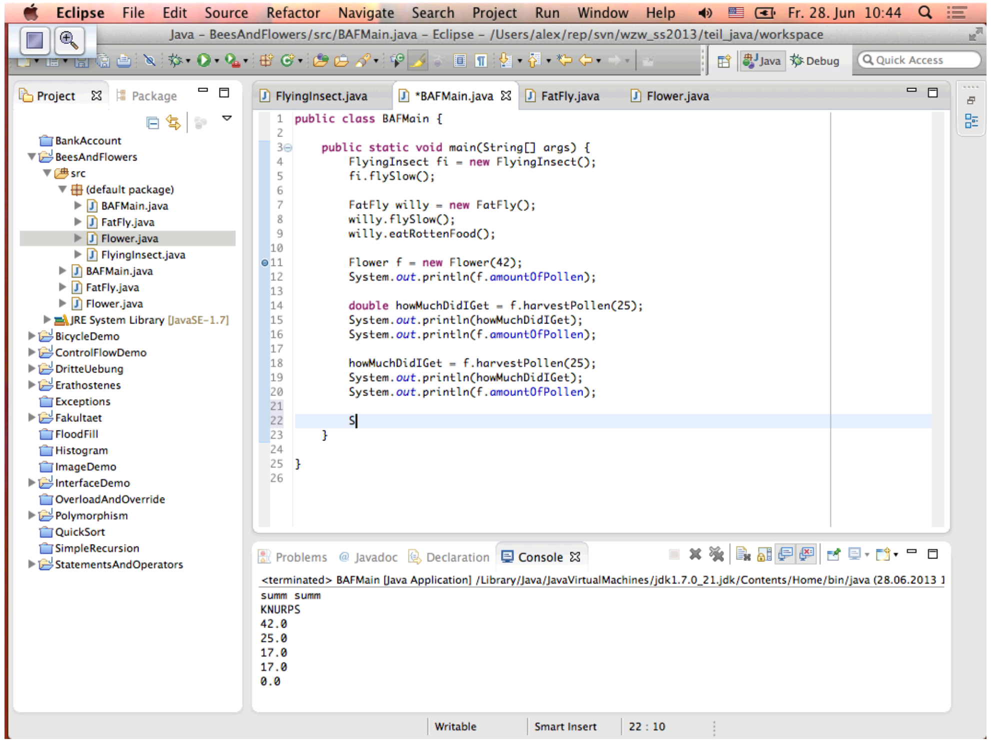
```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);

        howMuchDidIGet = f.harvestPollen(howMuch)
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
25.0
17.0
```



```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);

        howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
summ summ
KNURPS
42.0
25.0
17.0
17.0
```



```java
public class BAFMain {

    public static void main(String[] args) {
        FlyingInsect fi = new FlyingInsect();
        fi.flySlow();

        FatFly willy = new FatFly();
        willy.flySlow();
        willy.eatRottenFood();

        Flower f = new Flower(42);
        System.out.println(f.amountOfPollen);

        double howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);

        howMuchDidIGet = f.harvestPollen(25);
        System.out.println(howMuchDidIGet);
        System.out.println(f.amountOfPollen);

        S
    }

}
```

```
<terminated> BAFMain [Java Application] /Library/Java/JavaVirtualMachines/jdk1.7.0_21.jdk/Contents/Home/bin/java (28.06.2013 1
summ summ
KNURPS
42.0
25.0
17.0
17.0
0.0
```