Inner Class, Interface

Sahirul Alim Tri Bawono

Overview

- Class and Object
- Structur Class in Java
- Inner Class
- Interface

Class and Object

- A class is a pattern or template for creating multiple objects with similar features. It defines the variables and methods of the class. It declares the capabilities of the class.
- An object is an instantiation of a class. That is, an object is composed of the memory allocated for the member variables of the class. Each object has its own set of member variables.

Structure Class in Java

- Package
- Import
- Class
  - Instance Variable
  - Method

Packages

- Packages are collections of classes with similar functionality. Classes are composed of methods that support the functionality of the class. This organization provides structure to applications. Classes will always be in a package and methods will always be in a class
- Example: com.company.customer

Import

- The importstatement indicates which packages and classes are used by the class. This allows the compiler to determine whether the package's members are used correctly.
- Example: import java.lang

Class

- Example:
- class classname {
- // define class level variables
- // define methods
- }

Instance variable

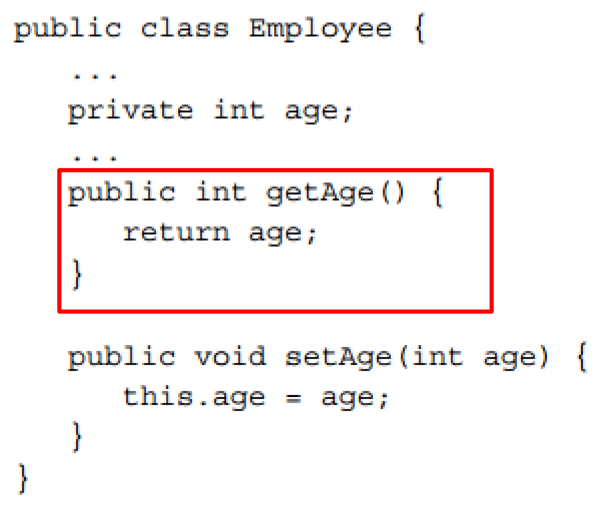
- When each object of a class maintains its own copy of an attribute, the field that represents the attribute is also known as an instance variable
- Example:
- public class Customer
- {
- private String name;
- private int accountNumber;
- private Locale locale;
- private BigDecimal balance;
- }

Method

- A method is a group of statements used to complete a specific task. A method has a return value, a name, a set of parameters, and a body. Parameters are passed to a method and are used to perform an action.
- Examples
- public Employee(String name, int zip, int age) {
- this.name = name;
- this.zip = zip;
- this.age = age;
- }

Accessor method

- An accessor method is one that reads or accesses a variable of a class



```
public class Employee {
   ...
   private int age;
   ...
   public int getAge() {
      return age;
   }

   public void setAge(int age) {
      this.age = age;
   }
}
```

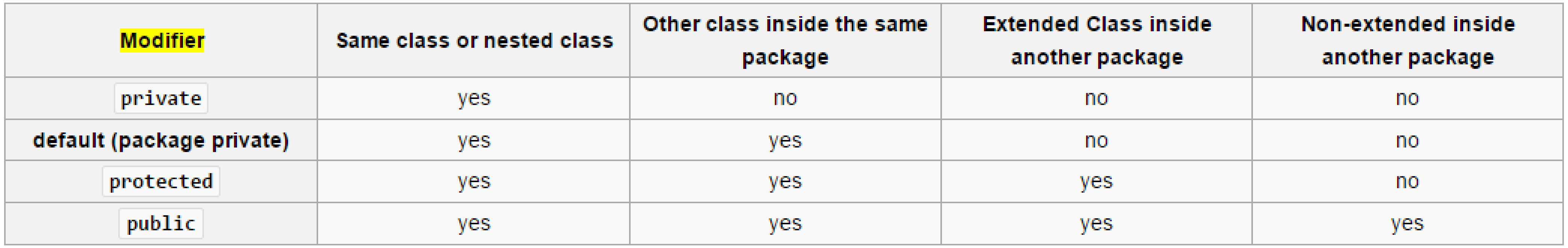
Mutator method

- A mutator method is one that modifies a variable of a class

```
public class Employee {
   ...
   private int age;
   ...
   public int getAge() {
      return age;
   }

   public void setAge(int age) {
      this.age = age;
   }
}
```

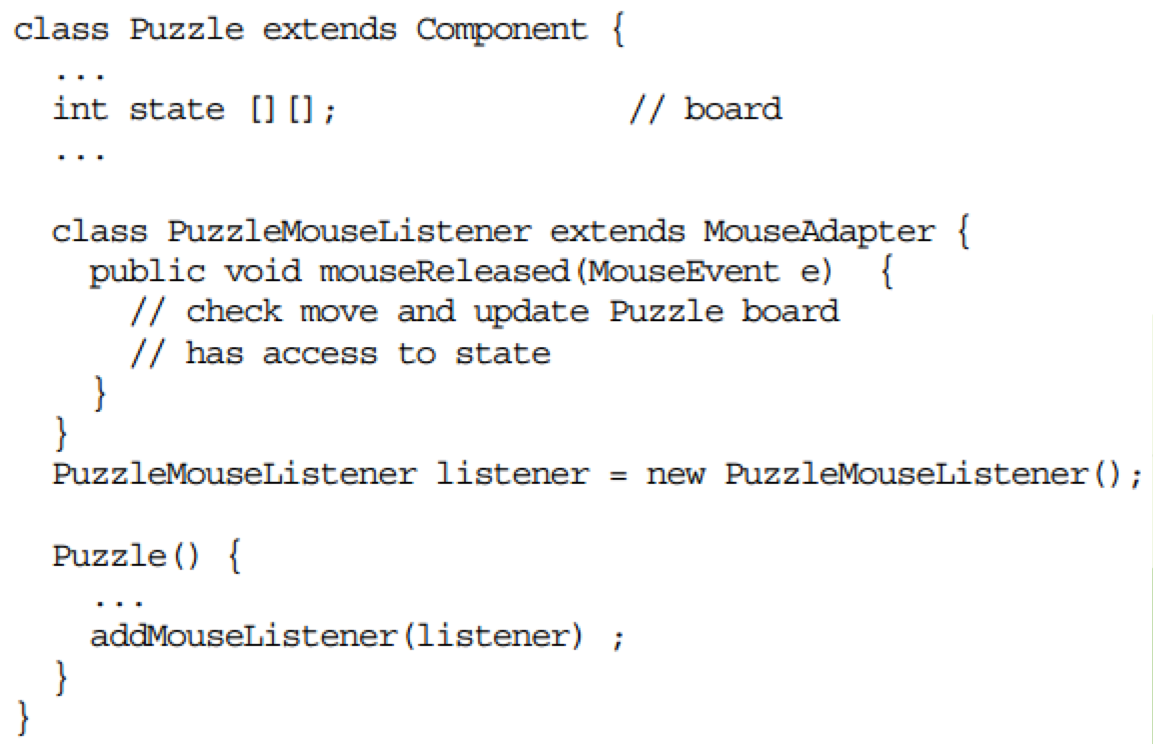
Access modifier



Modifier	Same class or nested class	Other class inside the same package	Extended Class inside another package	Non-extended inside another package
`private`	yes	no	no	no
default (package private)	yes	yes	no	no
`protected`	yes	yes	yes	no
`public`	yes	yes	yes	yes

Inner Class

- The revision of Java in JDK 1.1 allows for inner class, local class, and anonymous class definitions to solve this problem. While all classes in Java JDK 1.0 may only be declared at the top level, an inner class is one that is declared in another class.



```
class Puzzle extends Component {
  ...
  int state [][];                // board
  ...

  class PuzzleMouseListener extends MouseAdapter {
    public void mouseReleased(MouseEvent e)  {
      // check move and update Puzzle board
      // has access to state
    }
  }
  PuzzleMouseListener listener = new PuzzleMouseListener();

  Puzzle() {
    ...
    addMouseListener(listener) ;
  }
}
```

anonymous class

- When there is to be only one instance of an inner class, the creation of an anonymous class instance provides the syntactic sugar (**syntactic sugar** is syntax within a programming language that is designed to make things easier to read or to express) for more succinct code.

```
class Puzzle extends Component {
   ...
   int state [][];                 // board
   ...
   MouseAdapter listener = new MouseAdapter() {
     public void mouseReleased(MouseEvent e)  {
       // check move and update Puzzle board
       // has access to state
     }
   };
   Puzzle() {
     ...
     addMouseListener(listener) ;
   }
}
```

local classes

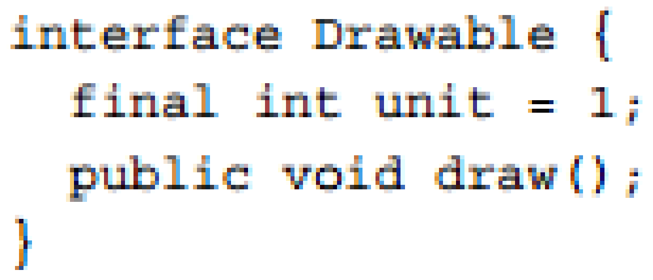
- While inner class definitions occur within an enclosing class (outer class), local classes are even more restricted in that they occur within blocks of method definitions.

```
class Puzzle extends Component {
   ...
   int state [][];                 // board
   ...

   Puzzle() {
      class PuzzleMouseAdapter {
         public void mouseReleased(MouseEvent e)  {
            // check move and update Puzzle board
            // has access to state
         }
      }
      ...
      addMouseListener(new PuzzleMouseAdapter());
   }
}
```

Interface

- An interface is similar to an abstract class. It is declared using the interface keyword and consists of only abstract methods and final variables. An abstract class normally has one or more abstract methods. An abstract method is the one that does not have an implementation.
- The following code defines an interface used to designate a class as capable of being drawn:



```
interface Drawable {
  final int unit = 1;
  public void draw();
}
```

Reference

- Reese, R. M.. 2012. Oracle Certified Associate, Java SE 7 Programmer Study Guide. Packt Publishing Birmingham
- Poo, D., et. All.Object Oriented Programming And Java .2008, Springer-Verlag. London.
- Wikipedia.com. Access 07-09-2015