# Getters and Setters Methods

Sungchul Lee

# Learning Object

- Getters and Setters
  - Access control using modifier
  - More secure
- Getters and Setters using Eclipse

# Getters and Setters

- How to set read only Object variable in the class?
  - Access Control using the modifier (private)

- If a class component (data member or method) is declared ***private***, client classes **cannot access** it
  - You can not read the value
  - You can not modify the value
- However, we want to read the object variable.
  - Use Getter and Setter

# Getters

- Getters and setters are the most basic methods in a class to manage the access control
  - Normally write getter and setter  for each attribute
- Getters are used to read attribute values.

```
public String getName () {
    return this.name;
}
```

Method which has Public modifier, can be accessed

- Special notes/tips
  - Normally, a getter has **no parameter, (but you can use it)**
  - The return type should be a **valid return type** (not void)

# Setters

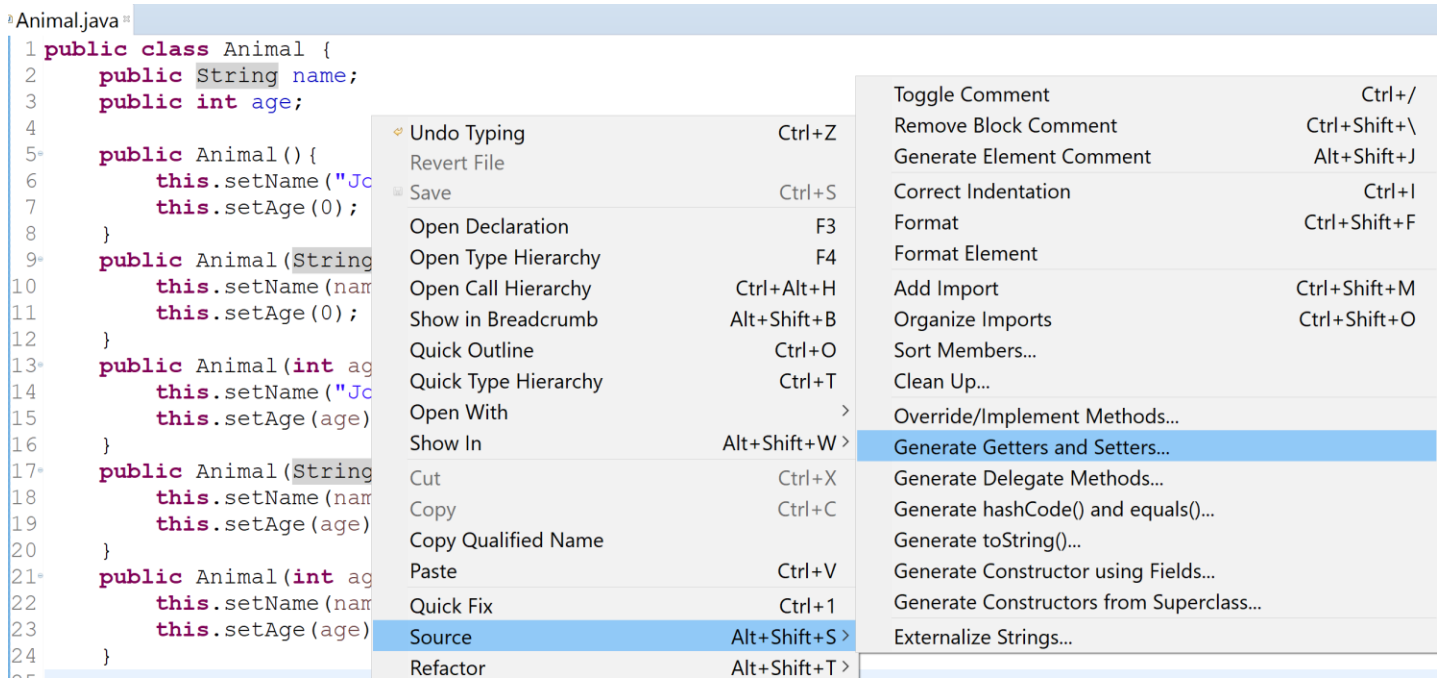
- Setters are used to initialize or modify the attribute values

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

- Special notes
  - Normally, a setter should have **parameter (s)**
  - The return type is **normally void**
    or Boolean for true/false for success message

# Getters and Setters using Eclipse

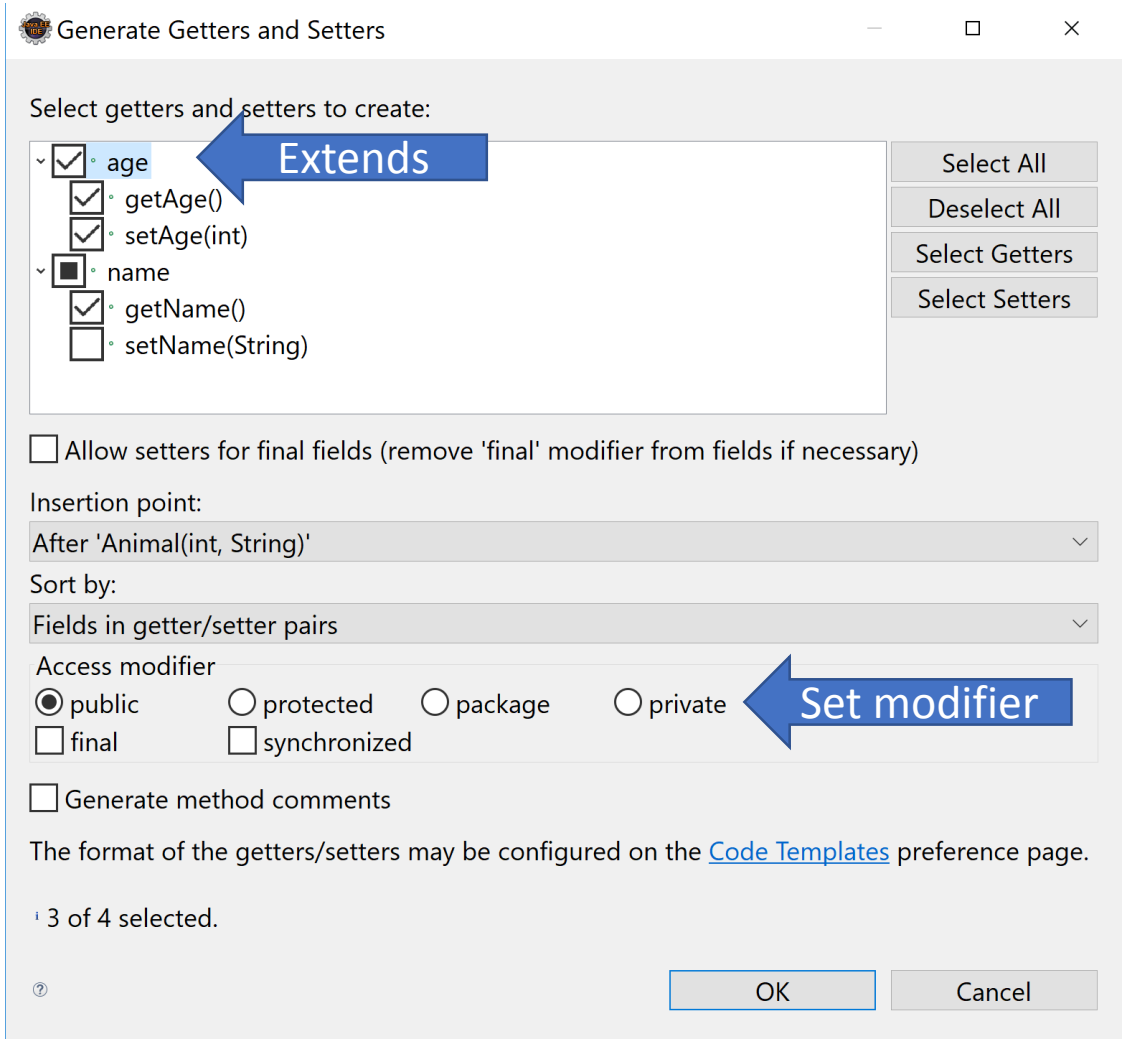
- Eclipse support to gererate getters and setters
  - After coding object variables and constructor
- How to use
  1. Mouse's right button Click on the line that you want to put the getter and setter
  2. Go to Source -> Generate Getters and Setters
  3. Click left button



# Getters and Setters using Eclipse – cont.

4. Extends the object variables
5. Select Getter and Setter
   - ❑Only getName() in name is selected
   - ❑Set private setName(String)
     - ❖Access control!!
6. Click “OK” button

```
public String getName() {
    return name;
}
private void setName(String name) {
    this.name = name;
}
```



# Practice

1. Make a new project (Reference: Create Project and Class File)
   - ❑Project name: Getter_Setter
2. Create two Class Files
   - ❑Class name: Animal
   - ❑Class name: Main
3. Coding:

# Practice - Coding

```
//Animal.java
public class Animal {
public String name; // Object Variable
public int age;

public Animal(){ // no paramter
        this.setName("Jone Doe");
        this.setAge(0);
}
public Animal(String name){//One string param
        this.setName(name);
        this.setAge(0);
}
public Animal(int age){//One int type param
        this.setName("Jone Doe");
        this.setAge(age);
}
public Animal(String name, int age){
        this.setName(name);
        this.setAge(age);
}
public Animal(int age, String name){
        this.setName(name);
        this.setAge(age);
}
public int getAge() {
return age;
}
public void setAge(int age) {
        this.age = age;
}
public String getName() {
        return name;
}
private void setName(String name) {
        this.name = name;
}
}
```

# Practice - Main

```
//Main.java
public class Main {
        public static void main(String[] args) {
        Animal cat = new Animal(); // no argument
        System.out.println("cat name:" + cat.name);
        System.out.println("cat age:" + cat.age);
        //cat.setName("New name for cat"); // Error
        cat.setAge(cat.getAge()+1); //next year +1 age
        System.out.println("Next year cat age:" + cat.age);

        Animal dog = new Animal("Pdog", 5);
        System.out.println("dog name:" + dog.name);
        System.out.println("dog age:" + dog.age);
        //cat.setName("New name for cat"); // Error
        dog.setAge(dog.getAge()+1); //next year +1 age
        System.out.println("Next year dog age:" + dog.age);
    }
}
```

# Practice – Code and Result

Main.java

```java
public class Main {
    public static void main(String[] args) {
        Animal cat = new Animal(); // no argument
        System.out.println("cat name:" + cat.name);
        System.out.println("cat age:" + cat.age);
        //cat.setName("New name for cat"); // Error
        cat.setAge(cat.getAge()+1); //next year +1 age
        System.out.println("Next year cat age:" + cat.age);

        // argument (String)
        Animal dog = new Animal("Pdog", 5);
        System.out.println("dog name:" + dog.name);
        System.out.println("dog age:" + dog.age);
        //cat.setName("New name for cat"); // Error
        dog.setAge(dog.getAge()+1); //next year +1 age
        System.out.println("Next year dog age:" + dog.age);

    }
}
```

Result

Problems Javadoc Declaration Console

<terminated> Main (3) [Java Application] C:\P

```
cat name:Jone Doe
cat age:0
Next year cat age:1
dog name:Pdog
dog age:5
Next year dog age:6
```

Animal.java

Getter_Setter/src/Animal.java

```java
public class Animal {
    public int age;

    public Animal(){
        this.setName("Jone Doe");
        this.setAge(0);
    }
    public Animal(String name){
        this.setName(name);
        this.setAge(0);
    }
    public Animal(int age){
        this.setName("Jone Doe");
        this.setAge(age);
    }
    public Animal(String name, int age){
        this.setName(name);
        this.setAge(age);
    }
    public Animal(int age, String name){
        this.setName(name);
        this.setAge(age);
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    public String getName() {
        return name;
    }
    private void setName(String name) {
        this.name = name;
    }
}
```

# Summary

- Object variable access control
- Getters and Setters
  - Access control using modifier
  - Getters method
    - Access object variable
      - Return
  - Setters method
    - Set object variable

```
25      public int getAge() {
26          return age;
27      }
28      public void setAge(int age) {
29          this.age = age;
30      }
31      public String getName() {
32          return name;
33      }
34      private void setName(String name) {
35          this.name = name;
36      }
37  }
```