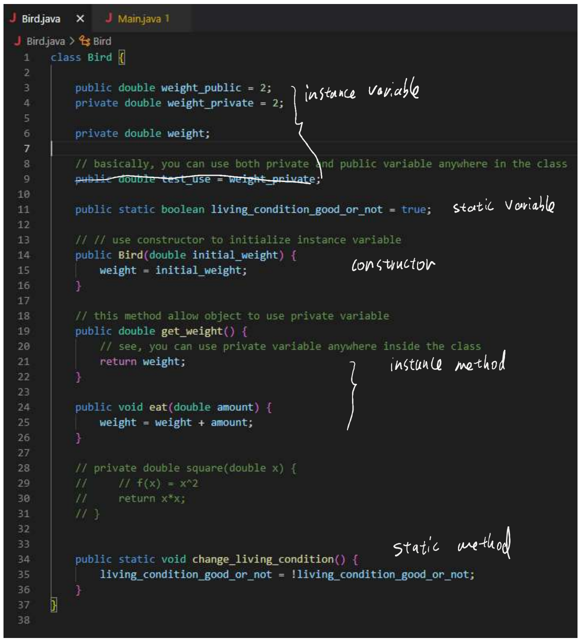
Class and object review session part1

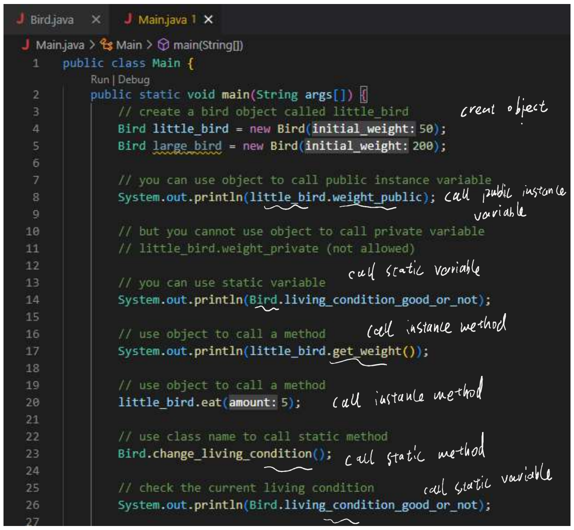
Friday, December 9, 2022 2:39 PM

We will focus on the following example class to review the material

As you can see, we usually design a class in a java file and create the object and test the class in another file. The test file is usually called client program. There must be a **static void main** in the test file. Your code will only run in a **main** method.



```java
class Bird {

    public double weight_public = 2;
    private double weight_private = 2;

    private double weight;

    // basically, you can use both private and public variable anywhere in the class
    ~~public double test_use = weight_private~~;

    public static boolean living_condition_good_or_not = true;

    // // use constructor to initialize instance variable
    public Bird(double initial_weight) {
        weight = initial_weight;
    }

    // this method allow object to use private variable
    public double get_weight() {
        // see, you can use private variable anywhere inside the class
        return weight;
    }

    public void eat(double amount) {
        weight = weight + amount;
    }

    // private double square(double x) {
    //     // f(x) = x^2
    //     return x*x;
    // }


    public static void change_living_condition() {
        living_condition_good_or_not = !living_condition_good_or_not;
    }
}
```



```java
public class Main {
    public static void main(String args[]) {
        // create a bird object called little_bird
        Bird little_bird = new Bird(initial_weight: 50);
        Bird large_bird = new Bird(initial_weight: 200);

        // you can use object to call public instance variable
        System.out.println(little_bird.weight_public);

        // but you cannot use object to call private variable
        // little_bird.weight_private (not allowed)

        // you can use static variable
        System.out.println(Bird.living_condition_good_or_not);

        // use object to call a method
        System.out.println(little_bird.get_weight());

        // use object to call a method
        little_bird.eat(amount: 5);

        // use class name to call static method
        Bird.change_living_condition();

        // check the current living condition
        System.out.println(Bird.living_condition_good_or_not);
```

Before move on, Let me first remind you what a class may contain

Bird class

① Instance variable

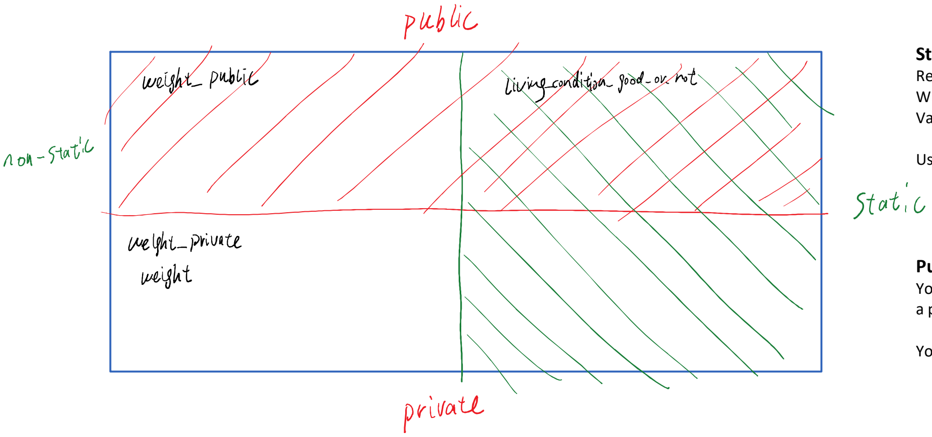
② Static variable

③ Instance method

④ Static method

⑤ Construction

Let me also remind you what is the difference between public and private, and static and non-static



	non-static	Static
public	weight_public	living_condition_good_or_not
private	weight_private weight	

Static and instance:

Recall that static variable belongs to the whole class,
Where the value will influence the whole class. Non-static variable (instance)
Variable belongs to a certain object.

Usually, we will have static variable to be public and instance variable to be private

Public and private:

You can access a public variable (method) in your client program, but you cannot access a private variable (method) in your client program.

You can freely use public or private variable inside the class.

How to create object. 调用 (call) method or variable in a client program.

Remember what I said, if you want to create an object, you need to create a separate java file (client program)

You need to use the class to call public static variable or method
You need to create an object and use object to call public instance variable or method

When you want to call a method, you may encounter the following situation:

1. Call a static method in a client program
   a. In this case, you just use **Class_name.method_name(any parameters)**
2. Call an instance method in a client program
   a. In this case, you just use **object_name.method_name(any parameters)**
3. Call a static method inside the class
   a. In this case, you just use **method_name(any parameters)**
4. Call an instance method inside the class
   a. In this case, you just use **this.method_name(any parameters)** (this 可以省略)

When you want to call a variable, you may encounter the following situation:

1. Call a public static variable in a client program
   a. In this case, you just use **Class_name.variable_name**
2. Call a public instance variable in a client program
   a. In this case, you just use **object_name.variable_name**
3. Call a static variable inside the class
   a. In this case, you just use **variable_name**
4. Call an instance variable inside the class
   a. In this case, you just use **this.variable_name** (this 可以省略)

Let's shortly review method here as well.

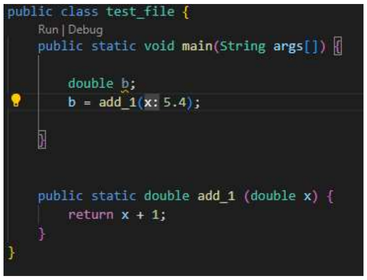
public/private (static) void/double/int/boolean/Class method_name (method parameter) {

}

You can see that you need to specify 5 parts in order to define a method:

1. Is your method public or private (usually public)
2. Is you method static or non-static
3. What does your method return
   a. Int/double/boolean: return primitive type
   b. String: return a String
   c. Class (name of some class): return an object of that class type
   d. Void: return nothing, you should not have a return statement in your method
4. What is your method name
5. What parameter does the method have

Some people have difficulty understanding method parameter, let me illustrate with an example here



```
public class test_file {
    Run | Debug
    public static void main(String args[]) {

        double b;
        b = add_1(x: 5.4);

    }


    public static double add_1 (double x) {
        return x + 1;
    }
}
```

Suppose you have a class called test_file, there is a "main" method and a "add_1" method in your class
Remember, you can call static method in a static method (Again, you can use static method in both instance or static method, but you cannot use instance method in a static method. We have mentioned this point several times)

The method "add_1" has the following property:

1. It is public
2. It is static
3. It returns a double
4. Its name is "add_1"
5. It takes a double parameter x

The method basically add 1 to the input parameter and returns it.