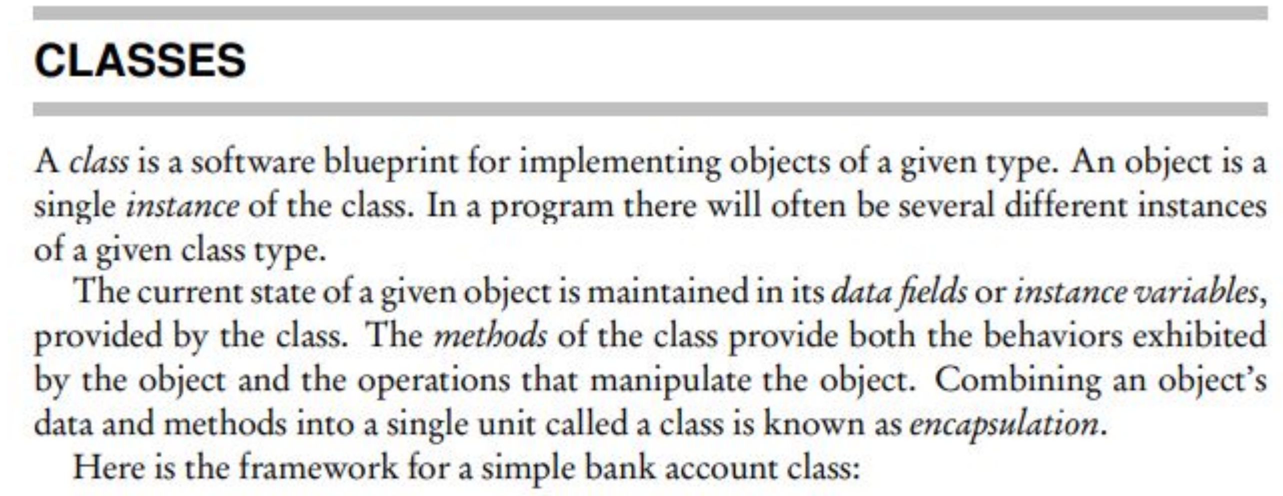
AP CSA Lecture 4

2022-10-29

Overview



CLASSES

A *class* is a software blueprint for implementing objects of a given type. An object is a single *instance* of the class. In a program there will often be several different instances of a given class type.

The current state of a given object is maintained in its *data fields* or *instance variables*, provided by the class. The *methods* of the class provide both the behaviors exhibited by the object and the operations that manipulate the object. Combining an object's data and methods into a single unit called a class is known as *encapsulation*.

Here is the framework for a simple bank account class:

Class:
Instance (Object):

Client Program:
You have main method in client program

Instance variable:
method:

Class

A class is a software blueprint for implementing objects of a given type

Eg:

Class: 麻雀

Object: 你家楼下的一只麻雀

Instance variable: 你家楼下这只麻雀的重量

Method: 飞，啄

Public and private

Class: (always use **public** in this class)

Public: can be used by code in other **package**

Private: cannot be used in code in other package (only used in folder that contains the class)

Method: (always use **public** in this class)

Public: The method is accessible by all classes

Private: The method can be used only in this class **and cannot be accessed by object**

Instance variable: (always use **private** in this class)

Public: The variable is accessible by all classes

Private: The variable can be used only in this class **and cannot be accessed by object**

Static variable

Static variable: variable that belongs to the whole class (impact every object)
Non-static variable (instance variable): variable that differs by object (unique to every object)
Usually static variable is public and instance variable is private

Example:

Static variable: 麻雀生存环境的空气质量
Non-static variable: 你家楼下那只麻雀的重量

Now, let’s define the class together

```
public class Bird {

        public static int air_quality = 9;
        Private weight = 3.3;


}
```

How to use variable? (look at Bird.java)

In class:

You can use public and private variable anywhere in the class

In client program:

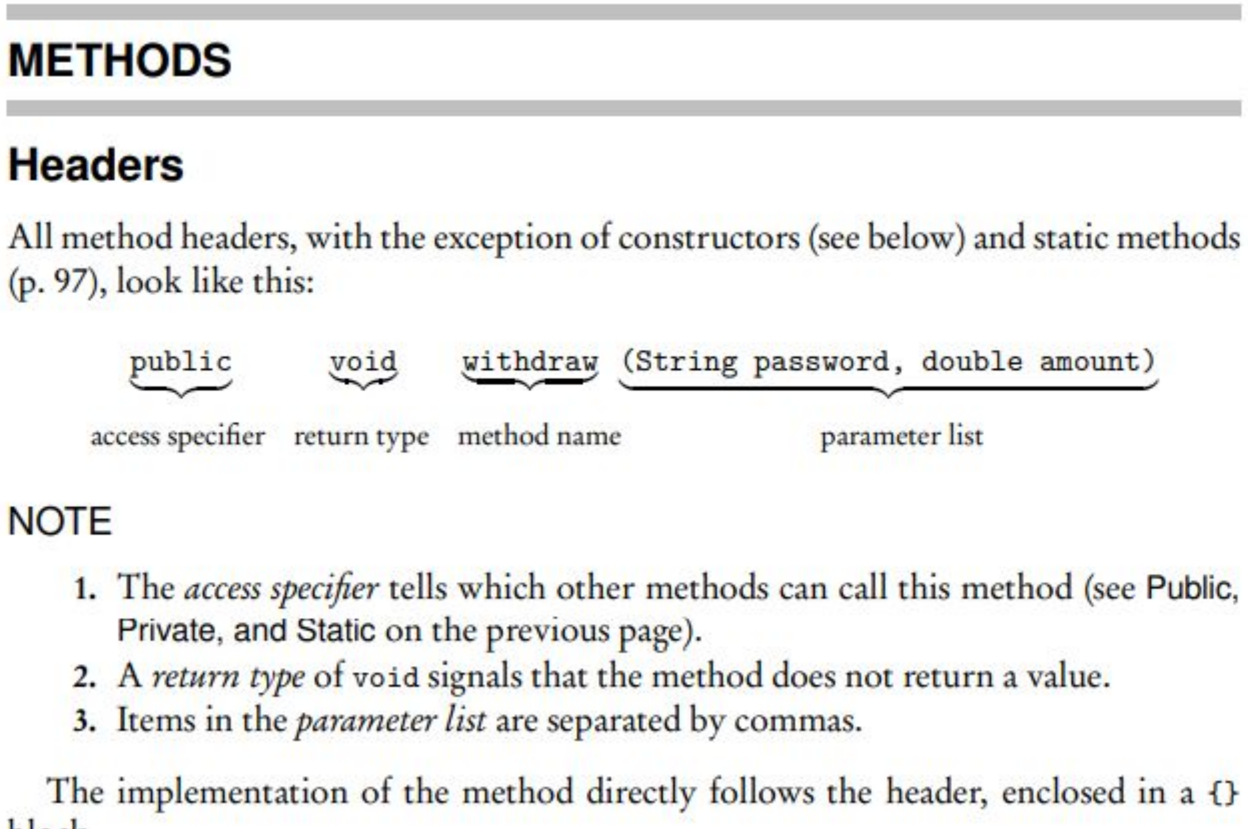
Suppose you create an object called little_bird in client program

You can access a public instance variable by using little_bird.variable

You can access a public static variable by using Bird.variable

Remember you cannot access a private variable in client program

Method



METHODS

Headers

All method headers, with the exception of constructors (see below) and static methods (p. 97), look like this:

$$\underbrace{\texttt{public}}_{\text{access specifier}}\quad\underbrace{\texttt{void}}_{\text{return type}}\quad\underbrace{\texttt{withdraw}}_{\text{method name}}\quad\underbrace{\texttt{(String password, double amount)}}_{\text{parameter list}}$$

NOTE

1. The *access specifier* tells which other methods can call this method (see Public, Private, and Static on the previous page).
2. A *return type* of `void` signals that the method does not return a value.
3. Items in the *parameter list* are separated by commas.

The implementation of the method directly follows the header, enclosed in a `{}`

Use method to allow object to access private variable

Look at Bird.java

How does the object use method

Look at Bird.java

Static method

A method that has static keyword is known as static method. In other words, **a method that belongs to a class rather than an instance of a class** is known as a static method. We can also create a static method by using the keyword static before the method name.

The main advantage of a static method is that we can call it without creating an object. It can access static data members and also change the value of it. It is used to create an instance method. It is invoked by using the class name. The best example of a static method is the main() method.

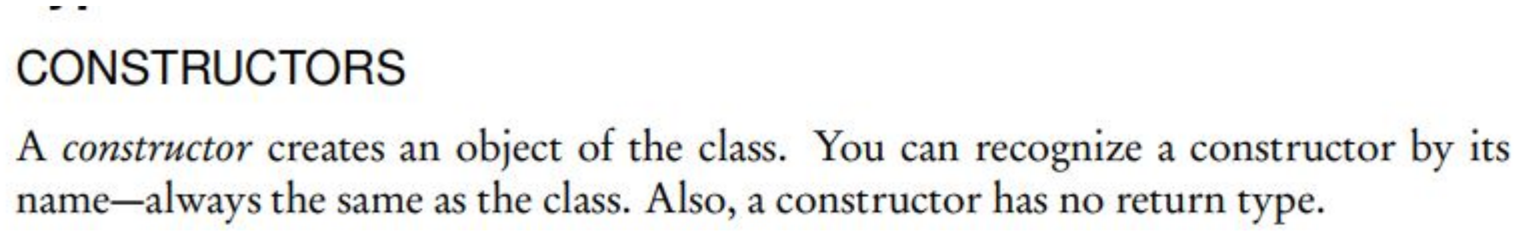
How to call a static method

Instance method vs Static method

- Instance method can access the instance methods and instance variables directly.
- Instance method can access static variables and static methods directly.
- Static methods can access the static variables and static methods directly.
- Static methods can’t access instance methods and instance variables directly. They must use reference to object.

Constructor

A kind of method that used to initialize instance variable of the class



CONSTRUCTORS

A *constructor* creates an object of the class. You can recognize a constructor by its name—always the same as the class. Also, a constructor has no return type.

```
public  Class_name() {

}
```