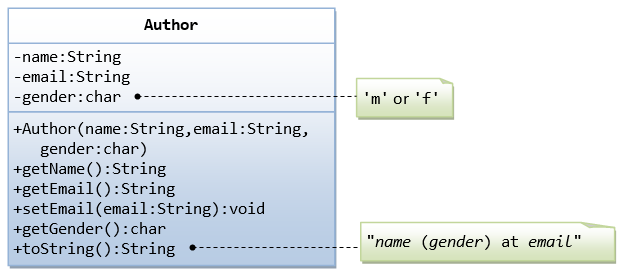
1. **Composition**

The Author and Book Classes

Let's start with the Author class



A class called Author is designed as shown in the class diagram. It contains:

* Three private member variables: name (String), email (String), and gender (char of either 'm' or 'f' - you might also use a boolean variable called isMale having value of true or false).
* A constructor to initialize the name, email and gender with the given values.

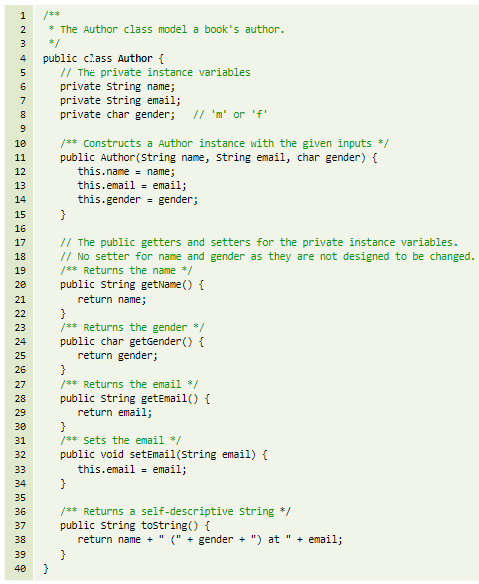
(There is no default constructor, as there is no default value for name, email and gender.)

* Public getters/setters: getName(), getEmail(), setEmail(), and getGender().

(There are no setters for name and gender, as these properties are not designed to be changed.)

* A toString() method that returns "name (gender) at email", e.g., "Tan Ah Teck (m) at [ahTeck@somewhere.com](mailto:ahTeck@somewhere.com)".

The Author Class (Author.java)

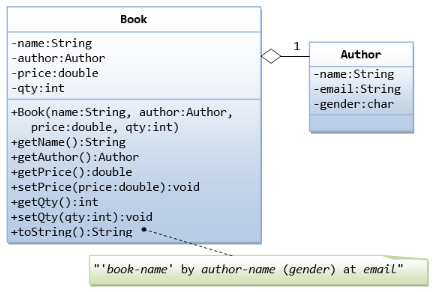


A Test Driver for the Author Class (TestAuthor.java)

Ảnh có chứa văn bản

Mô tả được tạo tự động

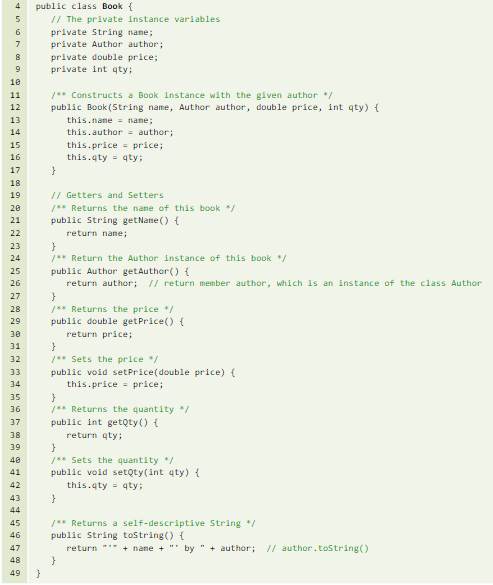
A Book is written by one Author - Using an "Object" Member Variable



Let's design a Book class. Assume that a book is written by one (and exactly one) author. The Book class (as shown in the class diagram) contains the following members:

* Four private member variables: name (String), author (an instance of the Author class we have just created, assuming that each book has exactly one author), price (double), and qty (int).
* The public getters and setters: getName(), getAuthor(), getPrice(), setPrice(), getQty(), setQty().
* A toString() that returns "'book-name' by author-name (gender) at email". You could reuse the Author's toString() method, which returns "author-name (gender) at email".

The Book Class (Book.java)



A Test Driver Program for the Book Class (TestBook.java)

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Inheritance

The Circle and Cylinder Classes

Ảnh có chứa văn bản

Mô tả được tạo tự động

In this example, we derive a subclass called Cylinder from the superclass Circle, which we have created in the previous chapter. It is important to note that we reuse the class Circle. Reusability is one of the most important properties of OOP. (Why reinvent the wheels?) The class Cylinder inherits all the member variables (radius and color) and methods (getRadius(), getArea(), among others) from its superclass Circle. It further defines a variable called height, two public methods - getHeight() and getVolume() and its own constructors, as shown:

Circle.java

Ảnh có chứa văn bản

Mô tả được tạo tự động

Cylinder.java

Ảnh có chứa văn bản

Mô tả được tạo tự động

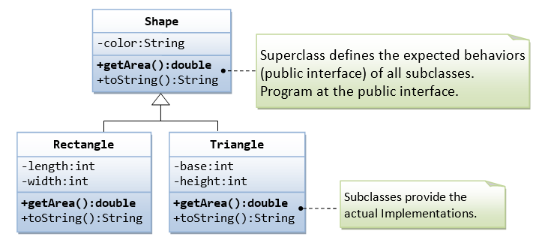
A Test Drive for the Cylinder Class (TestCylinder.java)

Ảnh có chứa văn bản

Mô tả được tạo tự động

1. Polymorphism

Shape and its Subclasses



Superclass Shape.java

Ảnh có chứa văn bản

Mô tả được tạo tự động

Take note that we have a problem writing the getArea() method in the Shape class, because the area cannot be computed unless the actual shape is known. We shall print an error message for the time being. In the later section, I shall show you how to resolve this problem. We can then derive subclasses, such as Triangle and Rectangle, from the superclass Shape.

Subclass Rectangle.java

Ảnh có chứa văn bản

Mô tả được tạo tự động

Subclass Triangle.java

Ảnh có chứa văn bản

Mô tả được tạo tự động

The subclasses override the getArea() method inherited from the superclass, and provide the proper implementations for getArea().

A Test Driver (TestShape.java)

Ảnh có chứa văn bản

Mô tả được tạo tự động