

Build Your Own Java Constructor



Estimated time needed: 30 minutes

In this lab, you will learn how to create a special overloaded constructor for a class and create its instances in Java.

You are currently viewing this lab in a Cloud based Integrated Development Environment (Cloud IDE). It is a fully-online integrated development environment that is pre-installed with JDK 21, allowing you to code, develop, and learn in one location.

Learning Objectives

After completing this lab, you will be able to:

- Create an overloaded constructor
- Learn how to implement encapsulation using constructor overloading
- Explain how to create instances of the class using overloaded constructors
- Clone an object of the class

Adding an overloaded constructor

If your `Book.java` and `BookAccess.java` programs and classes from the previous exercise is still intact, you can go step 6 on this page. If not, follow steps 1 through 5 to create the classes.

1. Create a project directory by running the following command.

```
mkdir my_class_proj
```

2. Run the following code to create the directory structure.

```
mkdir -p my_class_proj/src
mkdir -p my_class_proj/classes
mkdir -p my_class_proj/test
cd my_class_proj
```

3. Create a file named `Book.java` inside the `src` directory.

```
touch /home/project/my_class_proj/src/Book.java
```

4. Select the following button to open the file for editing.

Open **Book.java** in IDE

5. Read each statement in the following program and focusing on how the program defines the attributes and functions of `Book` class.

Paste the following content in `Book.java`.

```

public class Book {
    private String title;
    private String author;
    private float price;
    public void setTitle(String title) {
        this.title = title;
    }
    public void setAuthor(String author) {
        this.author = author;
    }
    public void setPrice(float price) {
        this.price = price;
    }
    public String getTitle() {
        return this.title;
    }
    public String getAuthor() {
        return this.author;
    }
    public float getPrice() {
        return this.price;
    }
    public String toString() {
        return "Title - " + this.title + "\nAuthor - "
            + this.author + "\nPrice - " + String.format("%.2f", this.price);
    }
}

```

When you create an overloaded constructor, another constructor with the same name (using the class name) but different sets of parameters, you need to also create a default constructor.

6. Add the following code in `Book.java`, after the attributes.

```

public Book() {
    this.title = null;
    this.author = null;
}
public Book(String title, String author, float price) {
    this.title = title;
    this.author = author;
    this.price = price;
}

```

`public Book()` - This is the default constructor, which will create an instance but the values will have to be explicitly set with the setters.

`public Book(String title, String author, float price)` - This is the constructor of the class which takes three parameters for each of the attribute. The value of each of the attribute is set with the parameter passed.

7. Compile the java program, specifying the destination directory as the `classes` directory that you created.

```
javac -d classes src/Book.java
```

8. Set the `CLASSPATH` variable.

```
export CLASSPATH=$CLASSPATH:/home/project/my_class_proj/classes
```

9. Create a new file named `BookAccess.java`.

```
touch /home/project/my_class_proj/src/BookAccess.java
```

10. Select the following button to open the file for editing.

Open **BookAccess.java** in IDE

11. Read each statement in the following program. Evaluate how the program creates instances or objects of `Book` class.

Paste the following content in `BookAccess.java`.

```
public class BookAccess {  
    public static void main(String s[]) {  
        Book book1 = new Book("Atomic Habits", "James Clear", 30.00f);  
        Book book2 = new Book();  
        book2.setTitle("Sapiens");  
        book2.setAuthor("Yuval Noah Harari");  
        book2.setPrice(25.00f);  
        System.out.println("The first book object is ");  
        System.out.println(book1);  
        System.out.println("The second book object is ");  
        System.out.println(book2);  
    }  
}
```

12. Compile the Java program, specifying the destination directory as the `classes` directory that you created.

```
javac -d classes src/BookAccess.java
```

```
java BookAccess
```

You will see the following output:

```
The first book object is  
Title - Atomic Habits  
Author - James Clear  
Price - 30.00  
The second book object is  
Title - Sapiens  
Author - Yuval Noah Harari  
Price - 25.00
```

Implement Cloneable interface to allow cloning

An Object of a class is not by default cloneable. To create a clone of the object, you will need to create an object and explicitly use setters and getters. This technique is not ideal, especially when the class has too many attributes. Java provides the Cloneable interface for this purpose.

1. Select the following button to open Book.java for editing.

Open **Book.java** in IDE

2. Add the following code within the Book class in place of the existing code.

```
public class Book implements Cloneable {
    private String title;
    private String author;
    private float price;
    public Book() {
        this.title = null;
        this.author = null;
    }
    public Book(String title, String author, float price) {
        this.title = title;
        this.author = author;
        this.price = price;
    }
    // Overriding the clone() method
    @Override
    protected Object clone() throws CloneNotSupportedException {
        return super.clone();
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public void setAuthor(String author) {
        this.author = author;
    }
    public void setPrice(float price) {
        this.price = price;
    }
    public String getTitle() {
        return this.title;
    }
    public String getAuthor() {
        return this.author;
    }
    public float getPrice() {
        return this.price;
    }
    public String toString() {
        return "Title - " + this.title + "\nAuthor - "
            + this.author + "\nPrice - " + String.format("%.2f", this.price);
    }
}
```

public class Book implements Cloneable - This code explicitly says that the class will implement the clone method which will enable clone of instances or objects of this class.

protected Object clone() throws CloneNotSupportedException - This method class the super class clone method which would internally clone all the attributes of the object into a new object.

3. Compile the java program, specifying the destination directory as the classes directory that you created.

```
javac -d classes src/Book.java
```

4. Select the following button to open the file for editing.

Open **BookAccess.java** in IDE

5. Read each statement in the following program and evaluate how program creates the instances or objects of Book class.

Paste the following content in BookAccess.java.

```
public class BookAccess {  
    public static void main(String s[]) throws CloneNotSupportedException {  
        Book book1 = new Book("Atomic Habits", "James Clear", 30.00f);  
        Book book2 = new Book();  
        book2.setTitle("Sapiens");  
        book2.setAuthor("Yuval Noah Harari");  
        book2.setPrice(25.00f);  
        System.out.println("The first book object is ");  
        System.out.println(book1);  
        System.out.println("The second book object is ");  
        System.out.println(book2);  
        Book book3 = (Book)(book1.clone());  
        System.out.println("The third book cloned object is ");  
        System.out.println(book3);  
    }  
}
```

public static void main(String s[]) throws CloneNotSupportedException - Any method within which clone() method of an object is being invoked should handle CloneNotSupportedException. You will learn more about exception handling at a later point in time. Adding throws CloneNotSupportedException is like adding a warning to anticipate the exception.

Book book3 = (Book)(book1.clone()); - This creates a clone of the object book1. book3 and book1 are identical though they are two different objects.

6. Compile the java program, specifying the destination directory as the classes directory that you created.

```
javac -d classes src/BookAccess.java
```

```
java BookAccess
```

You will see the following output:

```
The first book object is  
Title - Atomic Habits  
Author - James Clear  
Price - 30.00  
The second book object is  
Title - Sapiens  
Author - Yuval Noah Harari  
Price - 25.00  
The third book cloned object is  
Title - Atomic Habits  
Author - James Clear  
Price - 30.00
```

Provide user menu

You will now change a command line app to manage books allowing the user to select the constructor.

1. Create a file named BooksMenu.java inside the src directory.

```
touch /home/project/my_class_proj/src/BooksMenu.java
```

2. Select the following button to open the file for editing.

Open **BooksMenu.java** in IDE

3. Read each statement in the following program and review how the program creates the Books array, how the program adds books to the array, and the program retrieves books. Paste the following content in BooksMenu.java.

```
import java.util.Scanner;
public class BooksMenu {
    private static Book getExpensiveBook(Book book1, Book book2) {
        if (book1.getPrice() < book2.getPrice()) {
            return book2;
        } else {
            return book1;
        }
    }
    public static void main(String s[]) {
        Scanner scanner = new Scanner(System.in);
        Book[] books = new Book[10];
        int bkIdx = 0;
        while(true) {
            System.out.println( "Press 1 to view books, 2 to add books, "+
                                "3 to compare prices of books, any other key to exit");
            String userAction = scanner.nextLine();
            if (userAction.equals("1")) {
                for(int i=0; i<books.length; i++) {
                    if(books[i] != null) {
                        System.out.println(books[i]);
                    }
                }
            } else if (userAction.equals("2")) {
                if(bkIdx == 10) {
                    System.out.println("10 books added already. Cannot add any more books!");
                    continue;
                }
                System.out.println("Which constructor do you want to use? Press 1 for default, "+
                                    "any other key for overloaded constructor");
                String constructor = scanner.nextLine();
                System.out.println("Enter book title");
                String tmpTitle = scanner.nextLine();
                System.out.println("Enter book author");
                String tmpAuthor = scanner.nextLine();
                System.out.println("Enter book price");
                float tmpPrice = Float.parseFloat(scanner.nextLine());
                if (constructor.equals("1")) {
                    Book bkTmp = new Book();
                    bkTmp.setTitle(tmpTitle);
                    bkTmp.setAuthor(tmpAuthor);
                    bkTmp.setPrice(tmpPrice);
                    books[bkIdx++] = bkTmp;
                } else {
                    books[bkIdx++] = new Book(tmpTitle, tmpAuthor, tmpPrice);
                }
            } else if (userAction.equals("3")) {
                System.out.println("Enter index of first book to compare");
                int book1Idx = Integer.parseInt(scanner.nextLine());
                System.out.println("Enter index of second book to compare");
                int book2Idx = Integer.parseInt(scanner.nextLine());
                if (books[book1Idx] != null && books[book2Idx] != null ) {
                    System.out.println("The details of expensive book is \n"+getExpensiveBook(books[book1Idx], books[book2Idx]));
                } else {
                    System.out.println("One of the books is null");
                }
            } else {
                break;
            }
        }
    }
}
```

```
import java.util.Scanner; - Allows the program to use the Scanner class for taking input from the user.
Scanner scanner - Used to read user input from the console.
Book[] books = new Book[10] - Creates an array of size 10 to store up to 10 Book objects. At a later point when you learn about collections, you can replace
the fixed size array with an expandable collection.
int bkIdx = 0 - Keeps track of how many books have been added to the books array.
```

while(true) - The program runs an infinite loop to continuously display the menu until the user chooses to exit.
The program prompts the user to select from the following options:

- 1 To view books.
- 2 To add books.
- 3 To get the most expensive of two books
- Any other key to exit.

If the user presses 1 the program iterates over the books array.
If a Book object is not null, the program prints its details.

If the user presses 2 the program checks if bkIdx has reached 10 (array is full). If so, it prevents further additions. Else it prompts the user for the choice of constructor the user wants to use.

If the user presses 1 default constructor is used. If the user presses any other key, the program uses the overloaded constructor. The the user is prompted for the book's:

- Title (tmpTitle)
- Author (tmpAuthor)
- Price (tmpPrice, converted from string to float using Float.parseFloat).

The code creates a new Book object with the choice of constructor and stores the object in the books array at the bkIdx position.
The code then increments bkIdx to prepare for the next book object.

4. Compile the java program, specifying the destination directory as the classes directory that you created.

```
javac -d classes src/BooksMenu.java
```

5. Now run the java program.

```
java BooksMenu
```

The output for this program depends on the user input. Next, view a sample of the of that output.

```
Press 1 to view books, 2 to add books, 3 to compare prices of books, any other key to exit
2
Which constructor do you want to use? Press 1 for default,any other key for overloaded constructor
1
Enter book title
Gone with the wind
Enter book author
Margaret Mitchell
Enter book price
45
Press 1 to view books, 2 to add books, 3 to compare prices of books, any other key to exit
2
Which constructor do you want to use? Press 1 for default,any other key for overloaded constructor
2
Enter book title
Inner Excellence
Enter book author
Kim Murphy
Enter book price
12
Press 1 to view books, 2 to add books, 3 to compare prices of books, any other key to exit
1
Title - Gone with the wind
Author - Margaret Mitchell
Price - 45.00
Title - Inner Excellence
```

Author - Kim Murphy

Price - 12.00

Press 1 to view books, 2 to add books, 3 to compare prices of books, any other key to exit

Practice Exercise

1. Implement equals method in Book class which will return true if the title, author and price are equal.
2. Add a method in BookAccess which will compare the equality of two books.
3. Add a menu option 2 within the add a new object menu, to clone one of the existing books instead of creating.
4. Add a menu item 3 to the main menu, to change the price of a book at a particular index position.
5. Create a book object and add it into index 0.
6. Clone the object and add it to index 1.
7. Compare the books in index 0 and 1. Add the code to print a statement saying the two books are the same.
8. Change the price of the object in index 1.
9. Compare the books in index 0 and 1. Add the code to print a statement saying the two books are **not** the same.

Hint: Use Scanner and Integer.parseInt to get the index number

- Click here for the sample code for Book.java
- Click here for the sample code for BookAccess.java
- Click here for sample output

Conclusion

In this lab, you learned how to create class and define its attributes and add appropriate methods and to create one or more objects of the class.

Author(s)

[Lavanya](#)

© IBM Corporation. All rights reserved.