Java Fundamentals
Chapter 9 Practice Notes

Virtual Pair Programmers
Let the course come to you

In this chapter we are going to create a class to model a book. The class will have fields that store information about each book, a constructor method which allows us to set the initial value of each field, and methods to retrieve the values of each field.

The code we will write will be a simple representation of a book - we will simply store some key information, such as the book's title and author.

Task Overview

Stage 1

Create a new class, called Book, with the following fields:

Title - this will store the book's title - an example might be "The History of Java"

Author - this will be the name of the person who wrote the book - an example might be "Matt Greencroft"

ID - this will be a unique identification number for each book. An example might be 1203947. In a real library, this number might well be printed somewhere on the book, in the form of a barcode. It is then scanned by the librarian when a customer wishes to borrow it.

ISBN - this stands for International Standard Book Number - it is a unique identifier for each book, which is assigned by the book's publisher.

Stage 2

Create a constructor for the Book class, which sets the initial value of each of its fields.

Stage 3

Create get methods for the Title, Author and ISBN number fields.

Background Information

Because there can be many books which have the same title, the ISBN number will make sure that each book is unique. In fact, the ISBN number normally identifies different editions of a book. But the key is that all copies of a particular edition of a particular book will have the same ISBN number.

In our library we could have more than 1 copy of a particular book - if this happens they will have the same ISBN but unique IDs.

Be sure to watch the video!

After we have created the class, we then go on to create some books, and print out details about those books to the console. There are some Java syntax issues explained in this part of the video, so keep watching, even if you're comfortable with how to create the class!

See page 2 for a step-by-step walk-through

Java Fundamentals
Chapter 9 Practice Notes

This page contains supplementary information to support you in the task. Try and work out what is needed first, and then use this page as a hint if you get stuck. Don't forget there is a full work through on the video, and sample completed code in the Practicals and Code folder.

Task Detail

Stage 1a - create the new class

Right click on the project name, and select new class from the drop-down menu.

In the New Class dialog box, give the class a name - you should call the class Book (with a capital letter B - remember that Java is case sensitive).

Still in the New Class dialog box, make sure that the `public static void main(String[]args)` option is not selected - this is only needed for classes that are to be runnable.

Stage 1b - add fields to the new class

Fields will be defined using the following syntax:

```
private type name;
```

here `private` means that the field will only be visible inside the class (which is generally what you will want - it stops code from outside the class changing the data within the class)

`type` is the variable type, such as int, string or double

and `name` is the name of the variable, such as "title", or "author". Variable names should normally be entered in lower case.

Think about what variable type each field should be...

HINT: title, author and ISBN will all store characters, but ID will be whole numbers only.

Stage 2 - add class constructor

The constructor is the method which is run when the class is instantiated (when a new instance of the class is created).

The format for a constructor is:

```
public ClassName(type variableName,
       type variableName,... )
```

here `public` means that the constructor will be visible outside the class (which is vital - otherwise you can't create the class!)

each of the `type`s are variable types, such as int, string or double

and each of the `variableName`s are the name of the variables.

It's common to set initial values of some or all of the class's fields in it's constructor and that's what we are going to do here.

So for each of the different fields in the class, we will want to create a variable in the constructor's signature.

In the body of the constructor, we need to assign each of the variable field values. We do this using the following syntax:

```
this.fieldName = variableName;
```

Here `fieldName` is the name of the class field that you set in stage 1b, and `variableName` is the name of the variable as you defined it in the constructor's signature.

The keyword `this` is needed if both the class field and constructor variable have the same name... and if they do then `this` means the class's version.

HINT: don't forget that the constructor's code needs to be within curly brackets { } .

Stage 3 - add get methods

Because our class's fields are private they can't be accessed from outside the class, so we need to write methods to allow other code to find out the values of each of these fields.

The usual way to do this is to write 1 method for each variable using the following syntax:

```
public type getVariableName() {
       return variableName;
}
```

here `public` means that the method will be accessible from outside the class

The `type` is the variable type that this method returns, such as int, string or double

`getVariableName` is the method name - it's the variable name preceeded by the word get - you can call the method anythign you like but this is the usual convention. Note that it's normal to put the first letter of the variable name in upper class.

and the `return` line tells java what value to pass back to the caller when the method runs.

Be sure to watch the video!

After we have created the class, we then go on to create some books, and print out details about those books to the console. There are some Java syntax issues explained in this part of the video, so keep watching, even if you're comfortable with how to create the class!