Book Store Project Brief

This brief suggests you build a book store project. Then, in the lectures we'll build it too to show you how I'd do it. In addition, after completing the project we'll extend it by making it store data first in files and then in a database. Each lecture will build upon previous lectures, and each lecture builds upon you completing this project on your own first!

By tackling it on your own, the subsequent explanations are going to be that much clearer.

Your task

Your task, should you choose to accept it…

Is to create a console-based book store system that allows users to enter and retrieve book details, as well as mark books as read (meaning they've finished reading them already), and delete existing books.

Like the previous project, we'll be using an in-memory database (i.e. a Python list—which we're calling a database because it stores data). Store books in this list, and interact with them as the user requires.

What are books?

Books will, just like movies in the last project, be dictionaries. You can define the structure of a book to be anything you like, but this is my proposed structure:

```
{
    'name': BOOK_NAME,
    'author': BOOK_AUTHOR,
    'read': False  # or True
}
```

Marking books as read

The property `read` of the book is a boolean, so all we have to do to mark a book as read is to loop through all the books and setting the flag to `True` if the name matches what the user typed.

Deleting books

Deleting books is something that might look complicated, but it can be really straightforward if we think of the Python constructs we've already seen. You can use a list comprehension to re-create the books list **without** the book that the user typed.

For example, let's say you have a list of three books, and the user wants to delete `“Peter Pan”` .

You could have a list comprehension that adds each book to a new list if the name is not equal to `“Peter Pan”`. Like this:

```
def delete_book(name):
    books = [book for book in books if book['name'] != name]

# somewhere else in your application...
delete_book('Peter Pan')
```

Bonus points

You'll get bonus points for saving your books to a file and loading them from a file too!

You can do this in two ways:

- Have an option in the menu to save the current list of books, and another option to load the list of books that you've previously saved; or
- Every time a book is added, read, changed, or deleted, make this change in a file instead of a list.

I'm sure you can do it! If you're unsure, tackle it like the last project, and then try to work files in there afterwards.

By programming yourself and working through these exercises, you're going to grasp everything in programming really quickly.

Happy coding!

— Jose and the Teclado team