

## Week 5 assignments

When creating the program code, you must apply the following basic principles:

- create a separate project for each assignment;
- use name 'assignment1', 'assignment2', etcetera for the projects;
- create one solution for each week containing the projects for that week;
- make sure the output of your programs are the same as the given screenshots;

## CodeGrade auto checks

Make sure all CodeGrade auto checks pass (10/10) for your assignments. The manual check will be done by the practical teacher.

Auto checks assignment 1-<sup>10</sup>/<sub>10</sub> **AT**

Manual check

Automatic checks for assignment 1

0

10

10

100

%

Submit

5.00

10 / 20

×

⌵

## Assignment 1 – Logic layer

Last week we created an (book reservation) solution with a Model layer and a DAL layer, and tested it with a Console application. In this assignment, we are going to add a Logic layer.

The logic layer contains Service classes. In the case of simple systems, the services are organised per model class. In this book reservation system, we have a BookService, a CustomerService and a ReservationService.

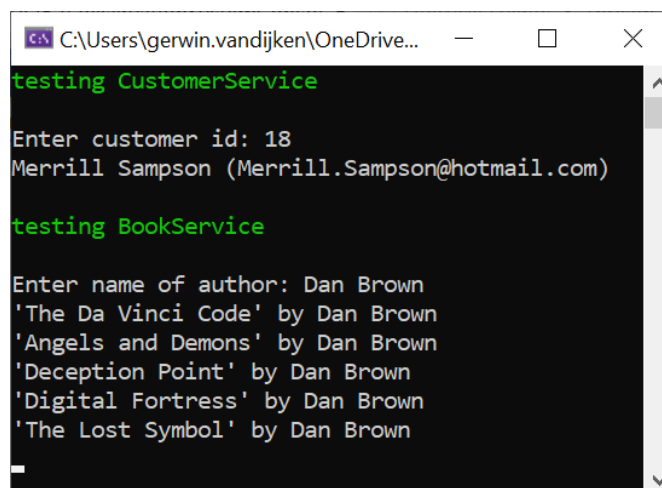
### The CustomerService class

- ☐ Create a new Class Library with name 'ReservationSystemLogic'.
- ☐ Add class CustomerService to this Logic layer.
- ☐ Add method GetAll() to the CustomerService class, that returns a list with (all) Customer objects. This method is using the GetAll method of a CustomerDAO object.
- ☐ Add method GetById(int customerId) to the CustomerService class, that returns a specific Customer object. This method is using the GetById method of a CustomerDAO object.

### The BookService class

- ☐ Create class BookService to the Logic layer.
- ☐ Add method GetAll() to the BookService class, that returns a list with (all) Book objects. This method is using the GetAll method of a BookDAO object.
- ☐ Add method GetById(int bookId) to the BookService class, that returns a specific Book object. This method is using the GetById method of a BookDAO object.
- ☐ Add method GetByAuthor(string authorName) to the BookService class, that returns a list of Book objects. This method is using a GetByAuthor method of a BookDAO object.

Test the methods of the CustomerService and BookService in a Console application. Show one specific customer (customerService.GetById) and show all books of one specific author (bookService.GetByAuthor).



```
C:\Users\gerwin.vandijken\OneDrive...  -  □  ✕

testing CustomerService

Enter customer id: 18
Merrill Sampson (Merrill.Sampson@hotmail.com)

testing BookService

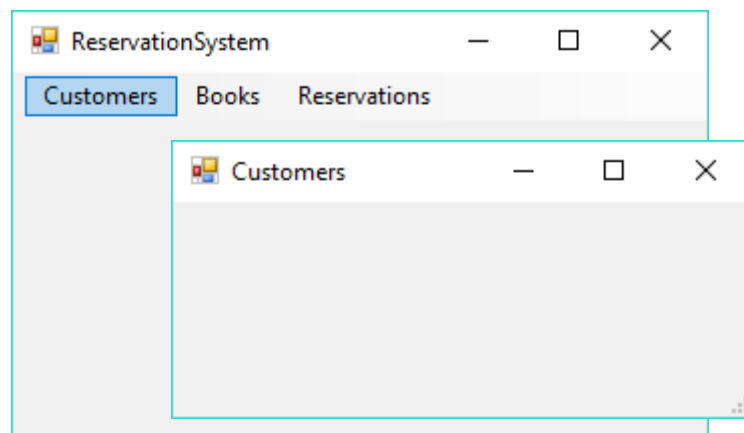
Enter name of author: Dan Brown
'The Da Vinci Code' by Dan Brown
'Angels and Demons' by Dan Brown
'Deception Point' by Dan Brown
'Digital Fortress' by Dan Brown
'The Lost Symbol' by Dan Brown
```

## Assignment 2 – Custom Forms **NOT MANDATORY!**

- ☐ Create a new 'UI' Windows Forms project for the Form classes. The form that is created by default is the main form (this is created in the 'Program.cs' file).
- ☐ Add a menu to the main form (MenuStrip control). Add three items to this menu: 'Customers', 'Books' and 'Reservations'.
- ☐ Create three new forms: CustomersForm, BooksForm and ReservationsForm (via Add, | New Item... | Windows Form).
- ☐ For each menu item, create an event handler by double-clicking on it in Design mode. The forms must be created via these three event handlers. This is possible using the following code:

```
CustomersForm form = new CustomersForm();  
form.ShowDialog();
```

- ☐ You can also show the forms using form.Show() instead of form.ShowDialog(). What is the difference?



## Assignment 3 – CustomersForm **NOT MANDATORY!**

We will now continue completing the CustomersForm form. As the form requires customer details, we will request these data via the CustomerService. The CustomerService, in its turn, will request these data from the CustomerDAO.

We will first complete the CustomersForm:

- ☐ Press <F7> to go to the code of the CustomersForm and ensure that the CustomersForm has an instance of the CustomerService:

```
public partial class CustomersForm : Form
{
    private CustomerService customerService = new CustomerService();
    ...
}
```

- ☐ Add a 'cmbCustomers' ComboBox control to the CustomersForm form. Enter all customers in the ComboBox using a separate (private) method 'DisplayCustomers'. This method retrieves the customers via the CustomerService object that was just created and then places the customers in the comboBox:

```
List<Customer> customers = customerService.GetAll();
...
```

- ☐ Select the first customer by setting the ComboBox property 'SelectedIndex' to 0.

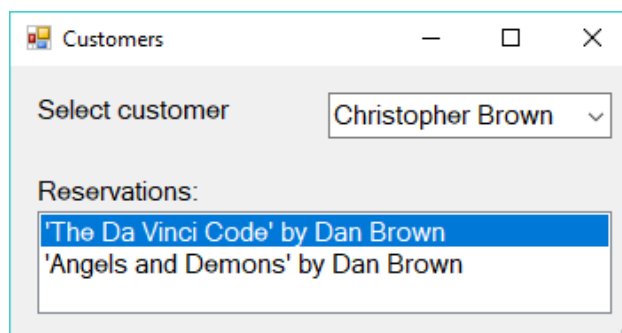
We will now add the reserved books of a selected customer to the CustomersForm:

- ☐ Create a class ReservationService in the Logic Layer and give this class a GetAllForCustomer method:

```
public class ReservationService
{
    private ReservationDAO reservationDAO = new ReservationDAO();

    public List<Book> GetAllForCustomer(Customer customer)
    {
        return reservationDAO.GetAllForCustomer(customer);
    }
}
```

- ☐ Place a ListBox control 'lstReservations' on the form in which the reservations of the selected customer are displayed. To do this, double-click on the ComBox control to create the Click event handler. In this event handler, all reservations associated with the selected customer are retrieved by calling reservationService.getAllForCustomer(customer); this method retrieves the books through the ReservationDAO.
- ☐ In the Click event handler, the retrieved books are added to the ListBox using lstReservations.Items.Add(...).

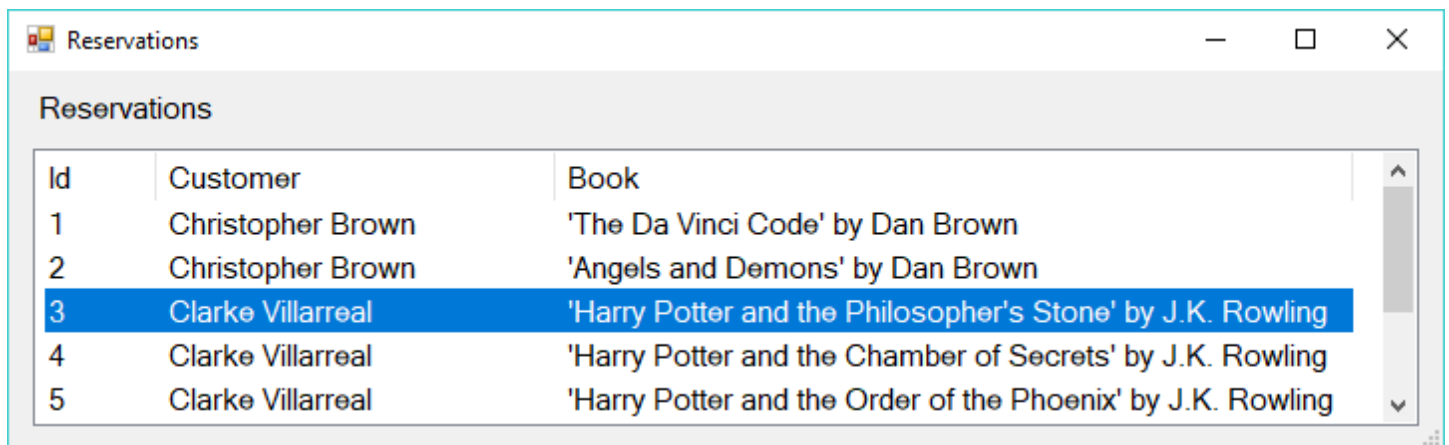


## Assignment 4 – ReservationsForm **NOT MANDATORY!**

In this assignment, we will continue completing the ReservationsForm form. Because we need data in this form as well, we need to add the required Service objects. In the case of the ReservationsForm, this is the ReservationService only.

- ☐ Add a 'lstReservations' ListView control to the ReservationsForm form. Set the View property to 'Details'. Add 3 columns to the ListView via the 'Columns' property (you can add columns using the 'Add' button).
- ☐ Enter all reservations from the reservationService in the ListView using a separate (private) method 'DisplayReservations'. You can add an item to a ListView control using the following code:

```
ListViewItem item = new ListViewItem(reservation.Id.ToString());  
item.SubItems.Add(reservation.Customer.FullName);  
item.SubItems.Add(reservation.Book.ToString());  
lvReservations.Items.Add(item);
```



## Assignment 5 – BooksForm **NOT MANDATORY!**

In this assignment, we will continue completing the BooksForm form. Because we need data in this form as well, we need to add the required Service objects. For the BooksForm, these are the BookService and the ReservationService.

- ☐ Add a 'cmbBooks' ComboBox control to the BooksForm form. Enter all books in the ComboBox using a separate (private) method 'DisplayBooks'. You can retrieve these books via the BookService, which in turn retrieves them from the BookDAO.
- ☐ Select the first book by setting the ComboBox property 'SelectedIndex' to 0.
- ☐ Place a ListBox control 'lstReservations' on the form in which the reservations of the selected book are displayed. To do this, double-click on the ComBox control to create the Click event handler. In this event handler, all reservations associated with the selected book are retrieved by calling `reservationService.getAllForBook(book)`; this method retrieves the books through the ReservationDAO.
- ☐ In the Click event handler, the retrieved customers are added to the ListBox using `lstReservations.Items.Add(...)`.

