**Debugger Lab**

Learning Outcomes

* Identify and fix bugs in pre-existing code using the Visual Studio Debugger

Overview

You are working on an application that keeps tracks of the books a user has read. The code for this lab can be cloned from the following URL: <https://git.rockfin.com/dev-build/Debugging_Lab_StarterCode.git>

When a user starts the application, they will be presented with an empty shelf of books:

Graphical user interface

Description automatically generated

When the user selects the *Select books you want to read here!* option they are presented with a random book from the collection.

Rate Books Page Elements

Graphical user interface, text, application, email

Description automatically generatedThis section explains the various interactive elements of the webpage.

**My Shelf** – Takes you to the list of books you have added to your shelf

**Rate Books** – Brings you back to the Rate Books page

**Add All Books to My Shelf** – Adds all 100 books in the collection to the user’s shelf

**Reject All Books** – Rejects all the books in the collection

**Read This Book** – Adds the displayed book to your shelf

**Don’t Read This Book** – Rejects the single book

The Bugs

Unfortunately, you may have noticed your application has thrown an exception and you cannot see these beautiful features. Well, you have identified the first bug; congrats!

Bug #1

The first bug occurrence can be reproduced by following these steps:

* After cloning the application, run it from Visual Studio
* On the index page select the *Select books you want to read here!* option
* The application throws an ArguementOutOfRangeException

**Objective:** Use the debugger to locate and fix the bug that is causing the exception.

Where did you locate the bug?

BookManager.cs, Line 80, int bookIndex = 101; This assigns a value to the index outside the 0-99 range of the book list.

Fix used: set the int bookIndex = 0;

What features of the debugger did you utilize to locate the bug?

Went and set a breakpoint to enter into the workflow where the asp-action first points to from the Index.cshtml view. I chose line 53 just inside the IActionResult RateBooks() method. From there I was able to watch the Locals values until I found a spot that the initial list index was outside of the expected range (0-99). Line 80 provided that point.

What was your fix for bug #1?

BookManager.cs, Line 80, int bookIndex = 101; This assigns a value to the index outside the 0-99 range of the book list.

Fix used: set the int bookIndex = 0;

Other solutions:

int bookIndex = random.Next(count);

Random rnd = new Random();  
int bookIndex = rnd.Next(0,99);

Bug #2

The second bug occurrence can be reproduced by following these steps:

* On the index page select the *Select books you want to read here!* option
* Select *Read This book (step 1 in image below)*
* Go to *My Shelf (step 2 in image below)*

Graphical user interface, text, website

Description automatically generated

* Click the image of the book you chose to read:
* Notice how many times it says you have read the book

Graphical user interface, text, application

Description automatically generated

* Click I have finished this book
* Now go back to the book again
* Check the number of read times
* The read amount is increasing beyond how many times you have actually read the book

**Objective:** Use the debugger to locate and fix the bug that is causing the read count to increase out of sync with the actual number of times you have read the book.

Where did you locate the bug?

In BookManager.cs Line 87: shelvedBook.TimesRead++;

What features of the debugger did you utilize to locate the bug?

Set a breakpoint

What was your fix for bug #2?

Comment out the BookManager.cs line 87. Don’t let the TimesRead++ to run before it should have.

Bug #3

The third bug occurrence can be reproduced by following these steps:

* On the index page select the *Select books you want to read here!* option
* Select *Read This book*
* Go to *My Shelf*
* Select the *I Finished This Book!* Option
* Click on the image of the book again
* Notice that only the date of when you finished the book is showing. The client would also like to view the time as well

Graphical user interface, text, application

Description automatically generated

**Objective:** Use the debugger to locate and add the time the book was finished with the date.

Where did you locate the bug?

Book.cs line 45

What features of the debugger did you utilize to locate the bug?

What was your fix for bug #3?

Conclusion

As a new developer, the debugger is one of the most valuable tools to have in your toolbox. It gives you a controlled way to step through the code line by line to (1) better understand how the code is behaving (2) to isolate different chunks of the code to identify any bugs or unexpected behavior.

Whenever you are presented with a new body of code or have noticed an unexpected behavior your first inclination should be to set a breakpoint and walk through it.