Software Implementation

A Marina Berth Booking System

An assignment on object-oriented design and programming

Gregory Boyle - SID 1747451
Autumn Trimester, 2018/19

Contents

Report

Introduction

The specification asks for a console application to manage an inventory of boats at a marina. A menu should present the app's main options: 1) record a new booking 2) delete a record 3) display all records (and available marina space) and 4) exit the program. Recording a new booking involves user input, checks and calculations before the user can decide to proceed. If so, data should be stored in an object, and that object saved to a list of records, before the user is returned to the menu.

Design

The *MarinaAdmin* class controls the user interface – displays messages, captures user input and retrieves the results of other classes to the user.

The *Marina* class does the 'heavy-lifting'; the logic, based on the business rules outlined in the specification.

The *Boat* class simply encapsulates the idea of a boat and controls the accessibility of the boat's properties from other methods via data hiding.

Implementation

The five boat properties –boat length, draft, name, owner, and type – are saved as a class object.

The objects are stored in a <list> during runtime.

Deletions from the list are implemented.

The list is written to an XML file on terminating the program and read from the same file when starting the program.

While methods are public, data is kept private.

Evaluation

Features

Abstraction - The code is broken down into separate, discrete functions, each responsible for a different task.

Formatted output – The records are displayed using columns and rows, so the data is more visually appealing to the user.

Speed of use – When possible, operations are presented as options chosen by single numbers. This is easier for the user.

Clarity - The user interface is explicit about what type of input is expected.

Limitations

Restrictions on user input – As per spec, the boat length values are saved as integers. However, in combination with Math.Ceiling(), this means entries of 2.5 and 2.1 are rounded up to 3. This imposes a restriction where users cannot input lengths other than whole metres.

Certain data not saved – The app currently asks the user for the duration of stay in months and calculates price of berth, but once the user has accepted the offer, the duration of stay and price of berth are not saved, which seems like a missed opportunity to increase the functionality of the app.

These limitations were not addressed as they were outside the scope of the specification.

Improvements

To enable users to input boat lengths of 2.5m for example, instead of an int, the boat length property should be a double.

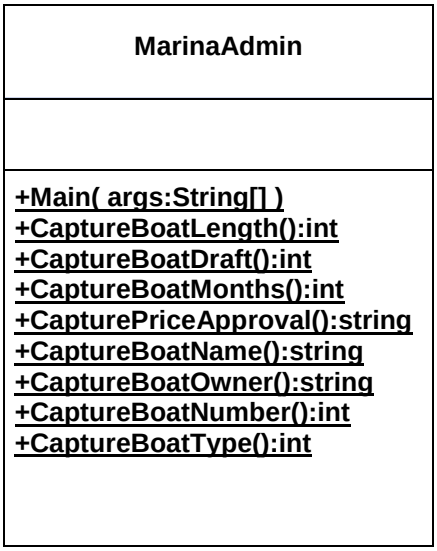
If the lengthOfStay, dateOfTermination and invoiceAmount were saved, the app could be more useful for payment tracking and renewal purposes.

Conclusion

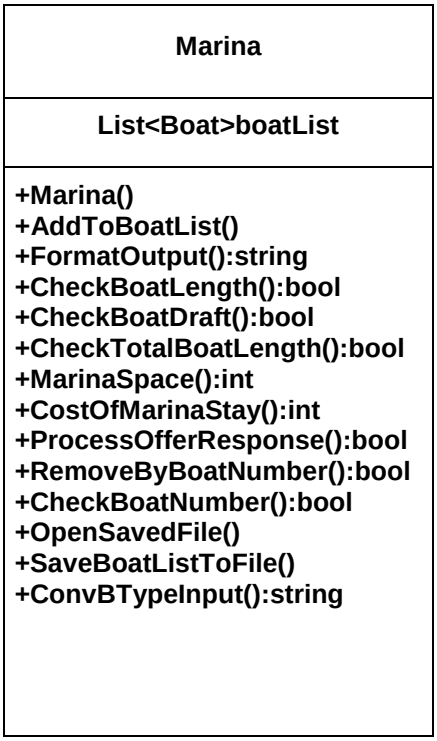
The application demonstrates the main functionality outlined in the specification, and its implementation follows an object-oriented approach.

Class diagram of the program

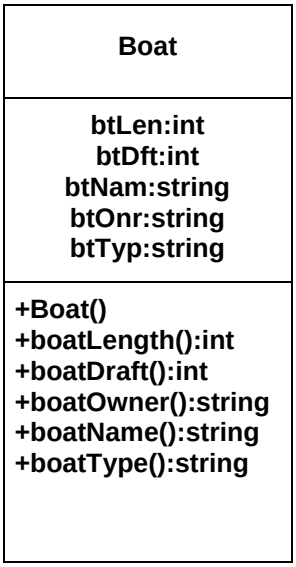
MarinaAdmin



Marina



Boat


Boat
btLen:int
btDft:int
btNam:string
btOnr:string
btTyp:string
+Boat()
+boatLength():int
+boatDraft():int
+boatOwner():string
+boatName():string
+boatType():string

Instructions on how to run the program

Open **MarinaAppV6.exe** located in:

USB STICK\Marina App V6\MarinaAppV6\MarinaAppV6\bin\Debug\

Choose from one of the main options: 1) Add Boat 2) Delete Boat 3) View Boats 4) Quit.

When quitting, the boat records will be saved to an XML file called **listofBoats.xml.**

Whenever the app is restarted, the XML file will be loaded, showing the boat records from the previous session.

A listing of the C# source code

See ‘**SI - Main Assignment - Source Code - G Boyle SID 1747451.pdf’** on the supplied USB stick.

An executable copy of the program

See **MarinaAppV6.exe** on the supplied USB stick.

Its location is:

USB STICK\Marina App V6\MarinaAppV6\MarinaAppV6\bin\Debug\MarinaAppV6.exe