Peer review comments Feiyu Xiong

General comment

a. Multiple controllers and view classes are making confusion. For view classes, since many other view classes are referring back to base view class for writing in command line, we believe the operations that are performed in other view classes could be done in one 'baseview' class.

b. Nice programme, but sometimes input requires 'enter' and sometimes does not, which made a confusion while we were testing it.

Class diagram

a. Association between boat and HandleMembers is not shown in the class diagram (Ln 97 of HandleMembers)

b. Dependency of boat and member list view is missing (line 76)

Model

a. Inside **[Member.cs]**, usage of boat id is discouraged; refer to discussion on Slack. But if it is still need to be generated we wonder if creating a random number between 0 and 100 is unique enough as it limits a number of boats a member can has. If a member id is connected to each boat, we wonder if it is necessary to use the boat id.

b. Similarly, **[MemberList.cs]**'s RegisterMember is somehow limiting the amount of member to 500.

c. The update member method Inside **[MemberList.cs]** is referring a member using integer id. It could be nice if it is taking Member class as an argument, like you did in delete member, and edit the member instead of creating a new.

d. We do not understand that update boat is registering a boat instead of updating an existing one in the **[Member.cs]**. Also, inside the **[ModelDAL.cs]** we do not quite understand why it is updating boat instead of adding.

View

e. Usage of static field of **[base view]** class should be replaced by handing over its baseview class as an argument/parameter in other classes. For example, line number 45 of the member list view, get chosen member id method could take base view as a parameter of the method to use it [1].

f. We believe having one view class is enough instead of having multiple view classes for a one programme. [HandleMember.cs]

g. Programme does not execute if there is no members.txt file. You might want to handle that case also.

h. For verbose list and compact list, Boolean compressed list was used and this reduces other reviewer's readability quite significantly.

Summary

Is the architecture ok?

It's okay but it can be improved. It is using model-view-controller pattern, and model and view part is separated. However, creating too many view classes two controllers are a bit unnecessary.

Is the requirement of a unique member id correctly done?

It is correctly done if and only if number of members are lesser than 500 and boats are lesser than 100 for each respective member.

What is the quality of the implementation/source code?

Usage of Boolean in a method could have be avoided since there are other two methods which has more intuitive naming. Updating and registering boat / member are implemented in a same way: to create a new instance. We wonder if it could be updating an existing class instead of overwriting it. Usage of static class of the base view is a huge minus; it could be avoided.

What is the quality of the design? Is it Object Oriented?

Yes, it is object oriented in some extends by usage of member class and boat class in other view/controller classes. It could be handled in batter way if objects were passed over instead of referring to ids in some methods.

As a developer would the diagrams help you and why/why not?

Yes, it helps as it is representing right relations with correct notations. However, usage of multiple view classes and dependencies among them would make other developers confused to understand the class diagram.

What are the strong points of the design/implementation, what do you think is really good and why?

Model is not depending on its view.

What are the weaknesses of the design/implementation, what do you think should be changed and why?

Usage of multiple view classes, which could have been merged.

P.S. We believe the submission passes grade 2 criteria.

Bibliography

[1] T. Butler, ”Tom Butler's programming blog,” 12 Auguest 2012. [Online]. Available: https://r.je/static-methods-bad-practice.html. [Använd 11 October 2016].