Further Programming Assignment 2 Report

s3878874 / Yu-Cheng Lai / Wednesday 14:30

In the implementation of this canvas GUI application, MVC (Model-Visual-Controller) design pattern is the main structure design approach. In the model component, there stores user data and functions/methods that will be used to retrieve, store, or update any user data. In the visual component, there are FXML files that are responsible for all the GUIs for the user, and they are all stored inside a resources folder. As for the controller component, each of the controllers is responsible for one FXML file. The controller classes decides all the functionalities that the user can interact with on the GUI. If the functionalities require any user data from the model component, the model object will be created in that corresponding controller class.

This implementation of this application also follows the SOLID principles. First of all, to show that this application has single responsibility in all its classes, each class serves only one purpose, and each of them have low coupling with each other. For example, all controller classes focus only on their own interface FXML file, and all visual components are only there to serve as a visual interface and no other purposes, while the model components are only responsible for all the functionalities that are related to backend database such as user data. Secondly, this application can be easily extended if there are any new changes and closed to modification if there's any changes needed to be made, and this falls under the open-closed principle in SOLID principles. For example, to add a new interface, it requires a new controller, and maybe the implementation of the model component functions. However, these extensions do not require any modifications in the existing code and all the extension is quite simple. Thirdly, Liskov substitution principle can also be found in this application thanks to the JavaFX library. In this application, all Shape class's subclasses including Circles and Rectangles, can all be easily substitute without any severe problems. The well-designed library saves us a worry on this principle. Second to last, the application also takes the interface segregation principle into account since there are no useless methods and all components are not highly coupled with each other. When all interfaces are separated like this application, we do not have to think about the recompilation of other interfaces whenever there's a change in the code. Last but not least, the dependency inversion principle is also considered in this implementation of this application. Again thanks to the well-designed library and low-coupling, no abstract classes are depending on any details and no high level modules need to depend on the lower level ones.

Further than that, Singleton design structure is also used while implementing JavaFX. In all the controllers, whenever the controller class is initiated, there's only one running instance, and this instance does not terminate until the whole program ends. In conclusion, this GUI canvas application follows the SOLID principles while the MVC component structure and singleton structures are implemented.