Sudoku Solver Java Project Documentation

By:-

Shadan Hussain(2020A7PS0134P)

Atharwa Pandey(2020A7PS1686P)

- ## Encapsulate what Varies

  Encapsulation is followed partially. In SolveSudoku class, integer N and M follow encapsulation because they are private and thus they can only be accessed using class functions. Board and oldtext 2d arrays do not follow encapsulation because access type is default.

- ## Favour composition over Inheritance

  Yes, this is followed. The Board_GUI class contains a SolveSudoku object as its data member, thus there is no inheritance used but Board_GUI contains SolveSudoku in its composition.

- ## Classes should be open for extension and closed for modification

  Classes are not very freely open for extension but they are closed for modification. It is possible to freely add more buttons and features but it will require some considerable amount of change in the code.

- **Depend on abstraction, do not depend on concrete classes**

No, since we are creating objects of the classes in the program , we require concrete classes. There is no abstraction as all the classes used in the program have their objects created somewhere or another.

Design Pattern

The code is resembling Command Design Pattern.

In object-oriented programming, the command pattern is a behavioural design pattern in which an object is used to encapsulate all information needed to perform an action or trigger an event at a later time. This information includes the method name, the object that owns the method and values for the method parameters.

Four terms always associated with the command pattern are command, receiver, invoker and client.

In our program : -

Command :- Command object is associated with buttons used in the program.

Receiver :- The receiver object is the SolveSudoku object since it receives data in its constructor and stores the required output. The text fields also acts as receiver objects as they store the user data.

Invoker :- The buttons are also acting as invokers since they are creating the SolveSudoku objects.

Client :- Board_GUI is resembled as client object because it enables the interaction between the buttons, text fields and the SolveSudoku objects.