This project was originally a sudoku solver that used single solution, single possibility, twins, and guess to solve puzzles. To add a new feature to this project a person is now able to solve the sudoku puzzle. To achieve this a command pattern was implemented. There are currently 4 commands: solve a cell, hint, undo, and a do nothing command.

These commands all inherit from the command class. The command class only has an execute method. There is no undo method in the command class because the only subclass that only had a need for an undo method was solve a cell. The reason why no other class needed undo was because there was no way to undo a hint and undo doesn't need one. In the future if more commands are added and there are multiple commands that need an undo method then the model should be changed.

When designing the client I considered having a client manager. ALthough this is applicable a manager was not implemented because the scale of this project made it unnecessary. The manager class would have allowed for easier management of a large group of clients however this project was only meant for a single player.

A single invoker could have been used for this project. The downsides to a single invoker are undo commands become complicated if there are multiple clients. The plus side is there is only a single invoker that needs to be in place. With an invoker belonging to a client it becomes easy to link undo commands with individual clients. If this project were to grow and have multiple clients having a single invoker that clients pass commands to would make sense. The invoker would take up less memory and it would decouple the invoker and the client. This would also mean something outside of the client could verify that the commands are valid.