Design Choices

SYSC 3110 Milestone 4: Documentation

Group Members:
Dorothy Tran 101141902
Evan Smedley 101148695
Harsimran Kanwar 101143556
Shuvaethy Neill 101143478

MonopolyModel Class

- Added the following new private helper methods to clean up the roll method
    - checkIfAI method to check if the current player is an instance of AI. Instead of having the same if statement nested in the if statements already placed in roll, it is much simpler and cleaner to have a one line method call to do so
    - handleNonDoubleTurnRoll method to handle the case where the player's roll was not a double and they are an AI. Similar to the reason listed for the new method above, this was previously a repetitive if statement nested in the if statements already placed in roll. Thus to clean up the code, it was extracted into a helper function so that it could be a single method call
    - payRent method to handle all instances where a player lands on an owned property and rent payment is owed. We saw this as a sub-functionality and extracted it to be a helper function so that it can be called in roll method instead
- Moved house and hotel logic from MonopolyFrame class to MonopolyModel class
    - New "BUILDING" state and "BUILD" constant used in play method to comply with new "Build Houses/Hotel" button
    - Created new helper method updatePropertyLists that keeps track of the colouredProperties a player has and if they own any full colour sets
    - Created a new method buildHouseHotel that is called when the "Build Houses/Hotel" button is pressed. It will check if a user is eligible to start building (checks if they own any full colour sets) and will notify the frame to proceed with the property colour set selection
- Implemented saveSerialize and importSerialize methods to save/load game data
    - saveSerialize saves the data stored in the model, specifically the players, their positions, the board pieces and the current player's turn. The data is saved as a text file under the name MonopolyGame.txt.
    - importSerialize imports previously stored game data and updates the current model. This method calls the previously stored MonopolyGame.txt file. In the case, there is no previously stored data, the program will inform the user the file does not exist, print the stacktrace, and then exit.

Player Class

- Implemented two new list attributes that are used in the house and hotel building logic
    - Implemented new HashMap attribute to keep tracks of the coloured properties a player owns (similar idea to the property cards in a physical game)
    - Implemented new Arraylist attribute to keep track of the complete property colour sets a player owns
- Implemented boolean canBuild attribute to determine the action of clicking the “Build Houses/Hotel” button is valid for the current player

MonopolyAIPlayer Class

- Moved hardcoded AI strategy logic into the MonopolyAIPlayer class and created method getRollDecision to handle that

MonopolyModelTest Class

- Implemented setUp method to eliminate repetitive instantiations in each test methods
- Added test cases to test the save\load and international features.

MonopolyFrame Class

- Created new menu items for “Save Game” and “Quit” and grouped them together under JMenu “Menu”
- Replaced old “Quit” button with “Build Houses/Hotel” to keep all game play command buttons in the same area

MonopolyController Class

- Added XML importing functionality using SAX Parser in this class
- This class now extends DefaultHandler (for responding to XML parser events)
- This class now has another dialog box that will get the user input for which version to use (which XML file to parse)