DAT171 - Computer assignment 3 (2019-02-25)

The objective is to make use of the library from the previous assignment in a graphical user interface for a Texas Hold ‘em game. The design should follow the Model-View-Controller concept. It should be possible to play the game until losing or winning. There needs to be at least 2 players.

The first step is to create a suitable layout for buttons, players, displaying card data etc.

1. Make a sketch (on paper) of how you want to display your Poker game. Add sections for all the data you wish to display, for example, each players hand, betting amount, the cards on the table, bet and fold buttons, as well as buttons for starting or quitting a game. Think through how you want this game to work on paper first instead of fixing things later.
   Tip: Use only one setup of input buttons (bet, fold etc.), representing the action for the active player.
2. Find suitable QT-layouts to construct your sketch. Draw these on paper as well, in order to get a good idea of the hierarchies of the layouts and in what order to construct the widgets. You **must** make a (reusable) player view widget.
3. Write the code which draws up your layouts (in a single window) and add all the necessary buttons etc. Use subclasses and overload for each region of the screen (e.g. input control, player information, card areas).
   The buttons don’t need to do anything yet.

See Lecture 11 for some examples of layouts, and for links to even more tutorials and examples on the available layouts. Primarily, you will probably want to use the HBox and VBox layouts, but feel free to investigate other layouts (like the grid layout). The purpose here is to learn to use widgets and layouts, so please make a simple “boxy” design.

The second step is to model the game state, and connect it to the GU code:

1. Make a class (or classes) that describe the *game state*. You don’t need to support more than 2 players, but some simple betting with win/lose conditions are required. Think of a game state as the information needed to save the game (who the active player is, which cards are on the table, the pot, etc.). Make methods for betting/folding etc. that manipulates this game state. E.g. a class “TexasHoldEm” with the methods “fold()”, “call()”, “bet(amount)” could be suitable. You **must** also create a class representing a player.
2. Connect the GUI widgets to the model, and add update functions in the view which displays the current state of the model (player cards, money, pot, player turn, etc.). You **must** use signals.
3. Make the game playable, with a definite win/lose state (a popup message is sufficient for notifying losses/victories). It is recommended to make a signal for alerting the GUI that the game has ended. You should be able to play several rounds of poker, until a player runs out of money.

See the card_view.py example for the split between model and view. The card_view.py script is available on the course homepage, which runs a small example. You are free to use any of the code from the card view widget, modify it to fit your game (or not use it at all).
Do **not** embed game logic in the GUI-code! You can do basic stuff like checking user input for correctness in the GUI-code, but keep a clear separation of the graphical interface and the underlying model (game state). Keep the GUI-code in a separate file.
You **must** make a custom widget for the main window to bundle layouts inside; see last example in lecture 11.
When handing in the assignment, please attach all files needed to run your program. Name the script that should be *started* **pokergame.py**