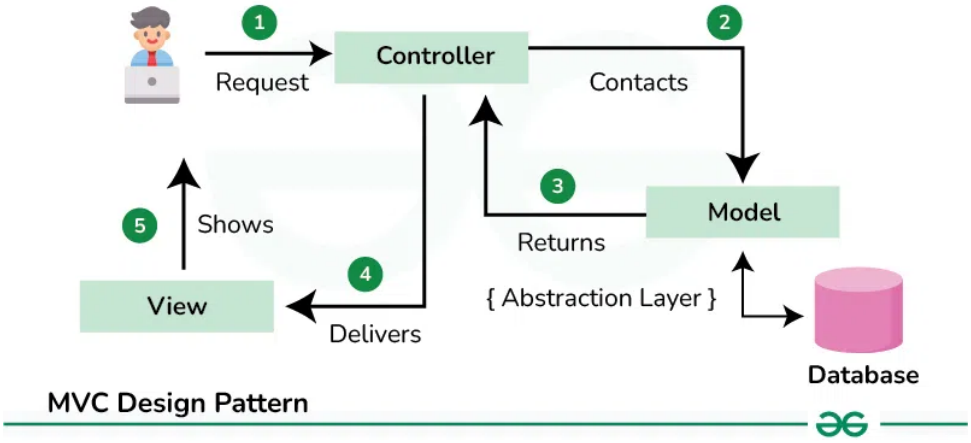
CSC2620 Object Oriented Design – Spring 2024
Unit 6 - MVC and Data Visualization

6.1 The Model View Controller Design Pattern

This is the classic GUI *design pattern*.



Example code on Classroom.

6.2 Practice

1. Create a new Java project.
2. Set up a Model-View-Controller pattern in your code by creating new classes for `Model` and `ViewController`.
3. Start with the `Model`
   (a) Grab the file `words.txt` from Classroom. It contains thousands of words in the English language.
   (b) Load it into a data structure/collection of your choosing in your project, making sure all data manipulation is handled by the `Model` class.
4. Build a GUI (a `ViewController`) that allows the user to view the entire list of words (probably should use a text area).
5. Add functionality to your GUI that allows the user to see all of the words that start with a particular letter or set of letters
   (a) Create a series of check boxes, one for each letter of the alphabet
   (b) When a box is checked, all letters that start with those words should be displayed.
   (c) Initialize all the boxes to be "unchecked"

6.3 Designing Good User Interfaces

These lecture notes were taken primarily from `https://sites.google.com/a/biola.edu/csci335/home/lecture-notes`.

- There are entire courses on UI/UX design. This unit will touch on some basic concepts at a very high level.
- UI Design - Designing user interfaces to efficiently optimize their ***usability***
- What is meant by the term usability?
  - It's a measure of the "user-friendliness" of an interface.
  - Accessible, comprehensible, intelligible, idiot proof, available, and ready
  - But it should also be helpful and overall not a hassle
- Shneiderman's "Eight Golden Rules of Interface Design" - These rules were obtained from the text Designing the User Interface by Ben Shneiderman. Shneiderman proposed this collection of principles that are derived heuristically from experience and applicable in most interactive systems after being properly refined, extended, and interpreted. To improve the usability of an application it is important to have a well designed interface. Shneiderman's "Eight Golden Rules of Interface Design" are a guide to good interaction design.

  1. Strive for consistency - Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent commands should be employed throughout.
  2. Enable frequent users to use shortcuts - As the frequency of use increases, so do the user's desires to reduce the number of interactions and to increase the pace of interaction. Abbreviations, function keys, hidden commands, and macro facilities are very helpful to an expert user.
  3. Offer informative feedback - For every operator action, there should be some system feedback. For frequent and minor actions, the response can be modest, while for infrequent and major actions, the response should be more substantial.
  4. Design dialog to yield closure - Sequences of actions should be organized into groups with a beginning, middle, and end. The informative feedback at the completion of a group of actions gives the operators the satisfaction of accomplishment, a sense of relief, the signal to drop contingency plans and options from their minds, and an indication that the way is clear to prepare for the next group of actions.
  5. Offer simple error handling - As much as possible, design the system so the user cannot make a serious error. If an error is made, the system should be able to detect the error and offer simple, comprehensible mechanisms for handling the error.
  6. Permit easy reversal of actions - This feature relieves anxiety, since the user knows that errors can be undone; it thus encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data entry, or a complete group of actions.
  7. Support internal locus of control - Experienced operators strongly desire the sense that they are in charge of the system and that the system responds to their actions. Design the system to make users the initiators of actions rather than the responders.
  8. Reduce short-term memory load - The limitation of human information processing in short-term memory requires that displays be kept simple, multiple page displays be consolidated, window-motion frequency be reduced, and sufficient training time be allotted for codes, mnemonics, and sequences of actions.

6.4 General Principles for Data Visualization

- Remember that the idea is to make your data as *easy to digest* as *quickly* as possible
- Really good list of best practices:
  `https://www.gooddata.com/blog/5-data-visualization-best-practices-0`

6.5 Data Visualization in Java

- We'll be using `JFreeChart` in this class for charts and graphs in Java. You can download the .jar files here:
  `https://sourceforge.net/projects/jfreechart/`
- Examples of utilizing JFreeChart in some slick UI design using nothing but Swing:
  `https://www.youtube.com/watch?v=OK2N-tibD9Q`
- `CSC2620_Unit6_JFreeChartExample`

6.6 Practice

1. Open your *previous practice solution*. Finish any functionality you were not able to complete previously.
2. Import the `JFreeChart` .jar files and make sure your program runs with it as part of the project.
3. Now, you will visualize some information about the list of words:
   - Using a `CardLayout`, create a section of the GUI (a panel) that allows a user to swap between charts/cards.
   - Create a chart that demonstrates the number of words in the list that begin with each letter of the alphabet.
   - Create a chart (on another card) that shows the lengths of the words (i.e. how many 2-letter words are there, how many 3-letter words are there, etc.). You can cap this at 15 or so (so how many words are 15+ letters long).