Jared Chiang: Project Portfolio Page for EzWatchlist

This document serves to show the various contributions that I have made in the development of EzWatchlist in a concise and understandable manner.

More About The Project

EzWatchlist is a desktop application that allows forgetful users to keep track of movies and tv series that they want to watch or have watched. It was developed by a group of 5 students from the National University of Singapore taking the Software Engineering module including myself under the premise that we had to either optimize or morph an existing AddressBook application. We chose the latter. Another constraint was ensuring that the application has a command-line interface, which means that all commands should be executable through typing. The picture below shows the various components of EzWatchlist.

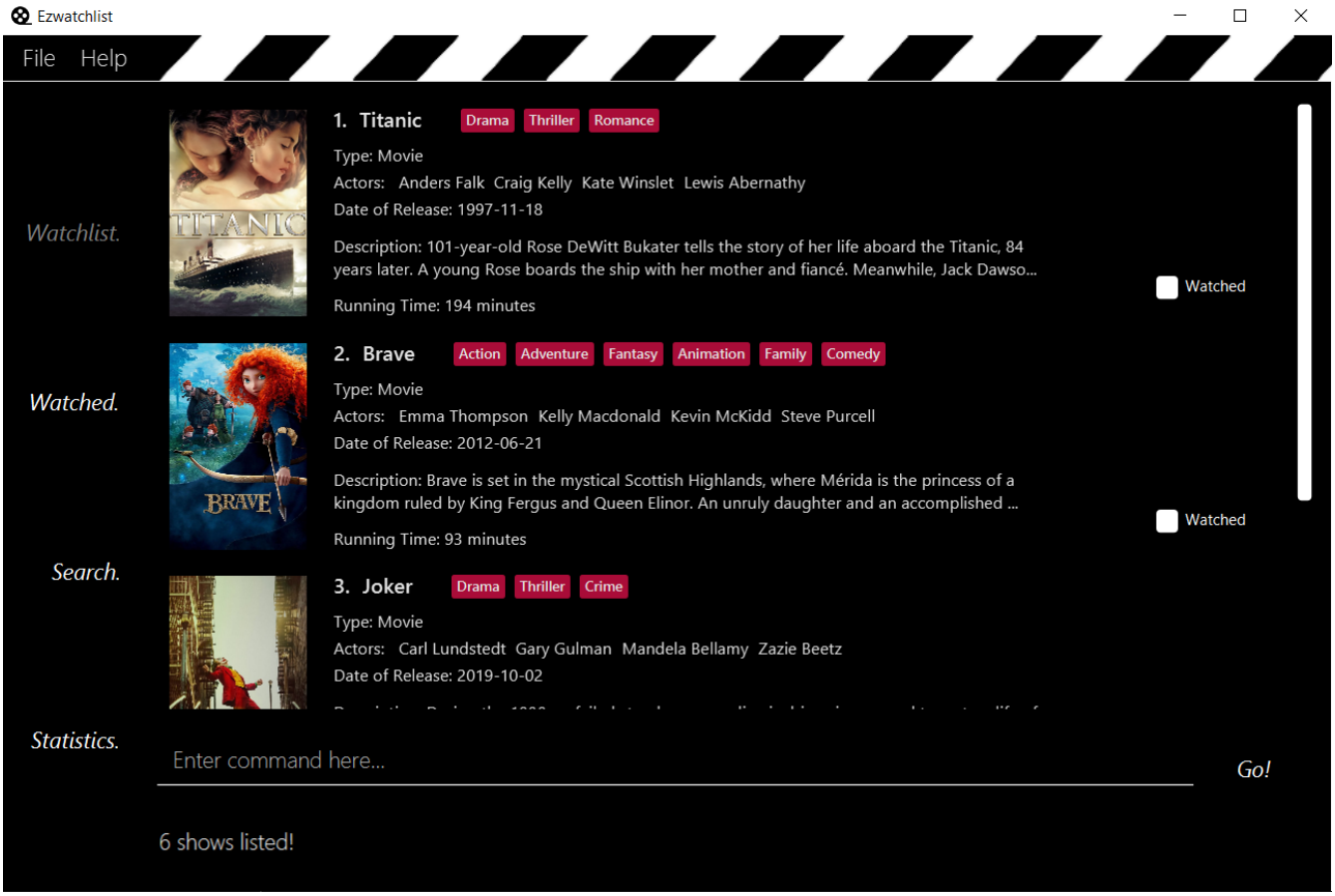

Figure 1. Graphical User Interface of EzWacthlist

Main Features

- Adding, editing and deleting shows
- Keeping track of shows and episodes that are watched

- Online search and syncing of shows
- Statistics about user's watchlist

NOTE	Highlighted words indicate commands or components of the application. Eg. `watch` command, `WatchCommand` class

Summary of Contributions

This section provides an overview of the contributions that I have made to the development of EzWatchlist. It also showcases some of my coding and documentation abilities, and the challenges I faced during the development of EzWatchlist.

Role

I was tasked with ensuring integration of the features of EzWatchlist. This means that I had to ensure that the commands work with one another well and perform their expected functions in the respective pages. This was a challenge since our application uses 4 different tabs, each with their unique functions and list of shows, so the implementation of the commands would vary from tab to tab.

Major Contributions

Implemented the `watch` feature

What it does

The `watch` feature allows users to update their watch list if a show has been watched. It also helps users to keep track of where they are in a TV series in terms of seasons and episodes.

Justification

The `watch` feature enables users to keep track of shows that they have and have not watched so that they can plan out what they want to watch in the future.

Highlights

This enhancement resulted in a need to change the storage structure of the shows to include the episodes and tv seasons. Its implementation was a challenge since it required structural change of the show objects, splitting them into movies and tv shows. The addition of the clickable checkbox to mark or unmark shows also increased the difficulty of the implementation.

Minor Contributions

- Separated the the watchlist into different sections (not watched and watched).
- Changed the storage structure of the shows to save them as either movies or tv shows. This feature was implemented with the intention of possibly retrieving season and episode details

future versions.

Code Contributions

View my code contributions here.

Other Contributions

NOTE	**#Numbers** represent pull requests to the mentioned contributions. These links show the code that has been contributed to the team project

Project Management

- Provided the team with information about the tasks that need to be completed
- Assigned tasks 1, 2

Enhancements to Features:

- Improved `add` command by reducing amount of information needed to be entered #138
- Changed storage structure to save shows as movies or tv shows #117
- Writing of various tests for EzWatchlist to reduce the possibility of bugs when using EzWatchlist #231, #237

Documentation:

- Improved the structure of the user guide #15, #214
- Updated the developer guide #241

Community:

- Reviewed team pull requests #74, #81, #84
- Reported bugs and suggestions for other teams in the class (examples: 1, 2, 3)

Tools:

- Set up Travis Continuous Integration

Contributions to User Guide

This section serves to showcase my contributions to the User Guide, as well as my ability to write documentation for users in a concise and understandable manner. It also shows my ability to use asciidoc and markup language for formatting.

Mark/Unmark as watched : `watch`

To mark an unwatched show in the watchlist as watched, use the command format listed below.

Format: `watch INDEX [e/EPISODE_NUMBER] [s/SEASON_NUMBER]`

Example Usage:

1. You want to mark "The Office" in the watchlist page as watched. Navigate to the watchlist by clicking on the watchlist tab or hitting the keyboard shortcut 1.

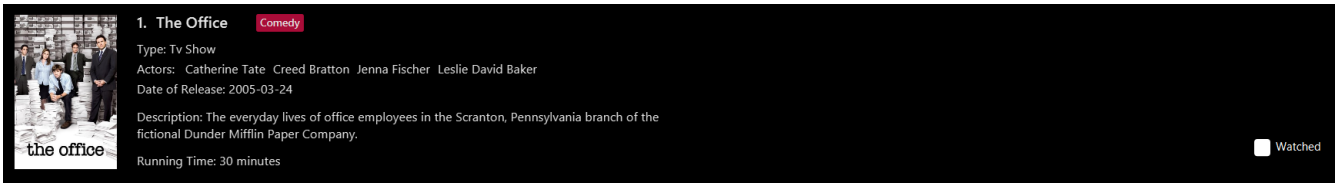

Figure 2. Viewing an unmarked show

2. Enter `watch 1` into the command box in the watchlist tab.

Figure 3. Entering the watch command

3. You may now view the show under the watched tab by clicking the watched tab or hitting the keyboard shortcut 2.

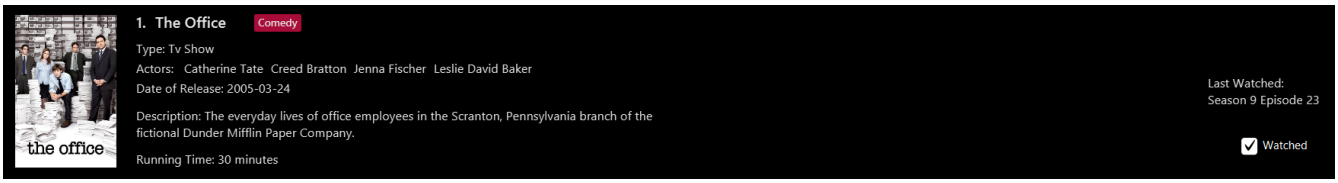

Figure 4. Viewing the newly marked show

Alternatively, you may click on the watched checkbox to toggle between whether a show is watched as indicated by the red arrow in the image below.

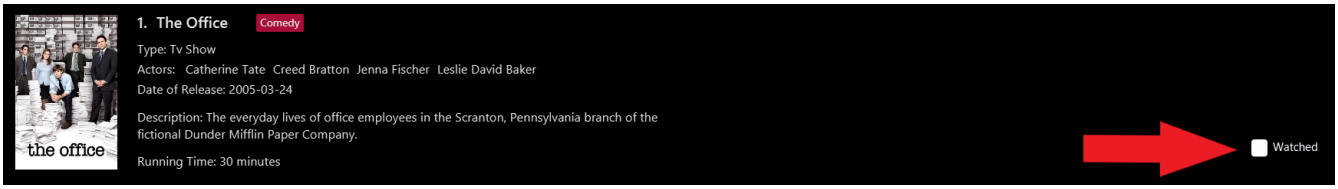

Figure 5. Clicking the checkbox to mark a show as watched

- The `index` refers to the index number shown in the displayed watchlist. The index **must be a positive integer** 1, 2, 3, …
- Any number of the optional fields may be provided.
- Having only the index of the show will mark/unmark the show as watched.
- Having the index and the episode number of the show will update the cumulative number of episodes of the show that are watched.
- Having the index and the season number of the show will update the cumulative number of seasons of the show that are watched.
- Having the index, season number and the episode number of the show will update the last watched episode to be the indicated episode of the indicated season of the show.

TIP Using the `watch` command on an already watched show will un-mark the show as watched.

Examples:

- `watch 1`
  Marks/un-marks the first show of the list as watched.
- `watch 2 e/20`
  Marks the first 20 episodes of the second show of the list as watched.
- `watch 2 s/5`
  Marks all episodes of the first 5 seasons of the second show as watched.
- `watch 3 s/5 e/2`
  Marks all episodes up to and including the second episode of the fifth season of the third show in the list as watched.

Contributions to Developer Guide

This sections showcases my contribution to the EzWatchlist Developer Guide. It exhibits my ability to use diagrams and technical terms to inform other developers of how the features were implemented and the possibility for enhancements.

[Feature] Mark/unmark as watched feature

The watch feature allows users to mark or unmark shows as watched. It also allows users to keep track of the latest episode of a TV series that they have watched.

Implementation

The mark/unmark as watched mechanism is facilitated by `WatchCommand` which can be found under the commands package. It extends `Command` and uses the `WatchCommandParser` to process the command entered by the user.

Given below is an example usage scenario and how the mark/unmark as watched mechanism works at each step.

Step 1. The user launches the application, and executes `watch 1 s/2 e/3` command to update the latest watched episode of the first show in the list.

Step 2. Entering the command calls `WatchListParser#parseCommand()`, which in turn returns a new `WatchCommandParser` and the `WatchCommandParser#parse()` command is called.

Step 3. A new `WatchCommand` is created, with the index of the show being parsed as a field of the `WatchCommand`. A new `WatchShowDescriptor` is also created to relay the episode number and season number to the `WatchCommand` object.

Step 4. The `WatchCommand#execute()` method is called, referencing the current `model`, and the show that is in the current `FilteredShowList` is referenced based off the current `model`.

NOTE	If the `index` is out of bounds, a new `CommandException` is thrown.

Step 5. A copy of the show is created through the use of `WatchCommand#createEditedShow()`, with the new total number of seasons and episodes updated if there are any changes. A new `isWatched` value of the show is also determined based on the number of episodes that are watched.

The following activity diagram below summarizes the calculation of the number of episodes watched:

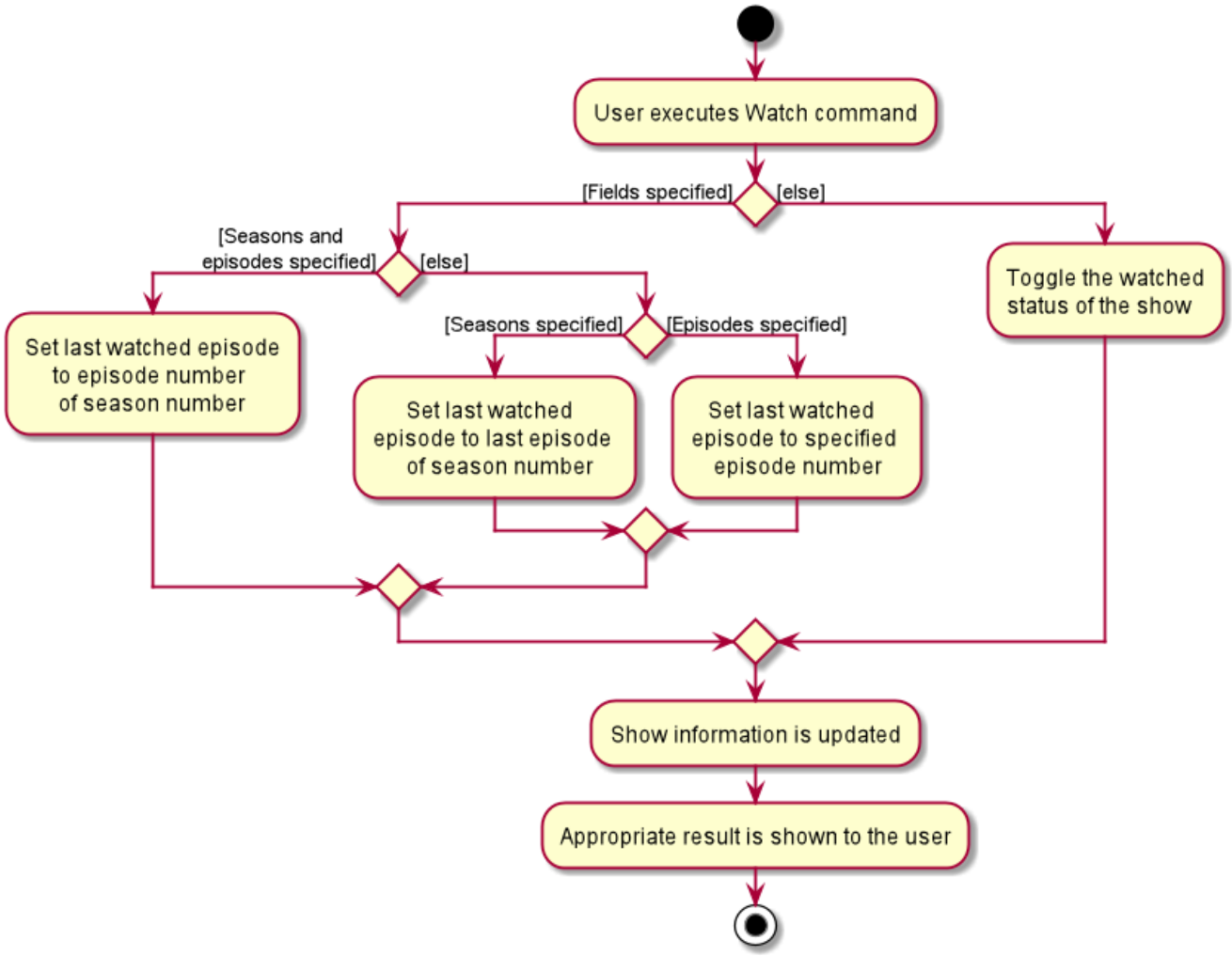

Figure 6. WatchActivityDiagram showing how episodes are calculated

Step 6. The show in the current show list is updated to the newly created copy with the updated watched status and latest episode watched, and a `CommandResult` with the new watched status of the show is created.

The following sequence diagram shows how the watch operation works:

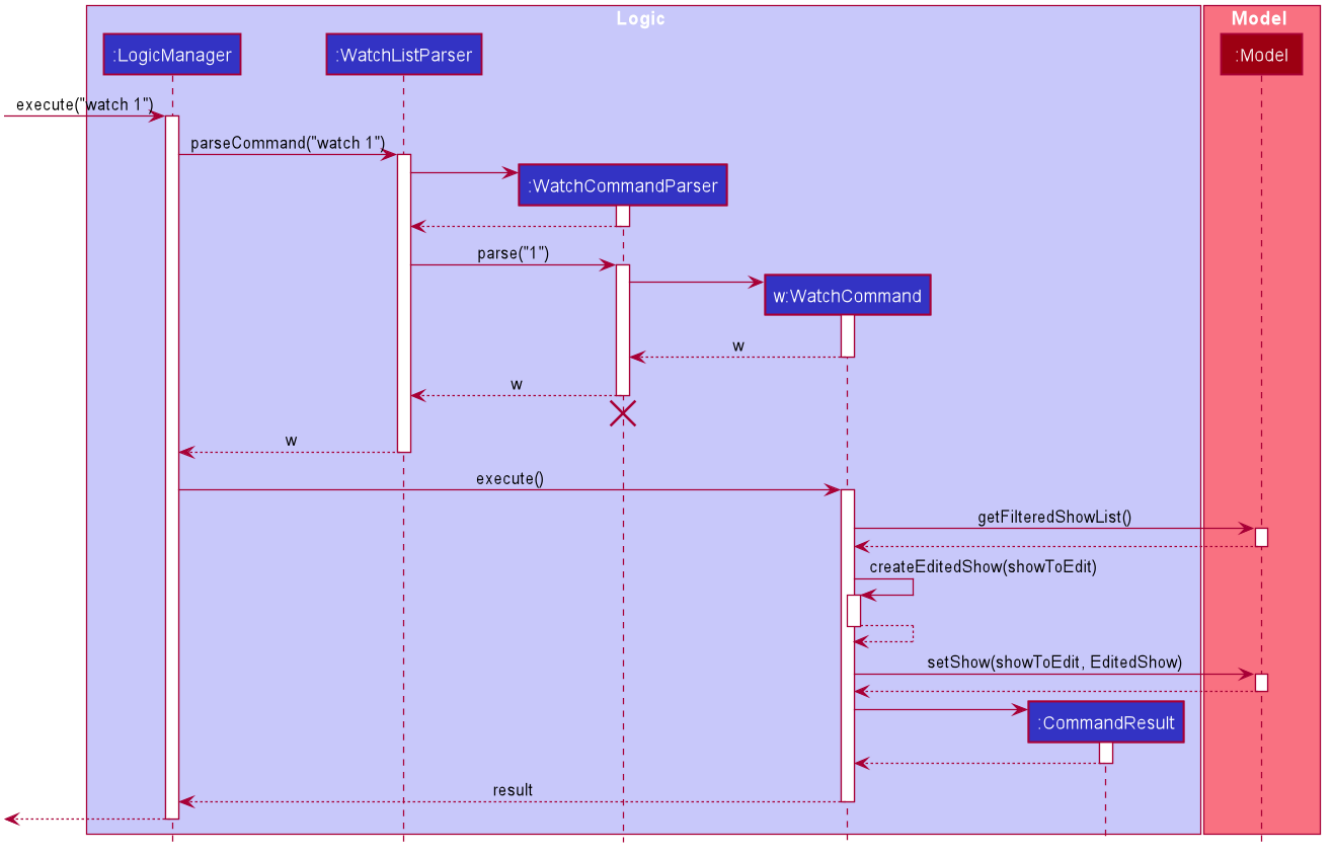

Figure 7. WatchSequenceDiagram showing flow of the watch command

Design Considerations

Aspect: Creating a new WatchCommand instead of an altered EditCommand

- **Alternative 1 (current choice):** Creating a new `WatchCommand` class for changing the 'watch' status of a show.
    - Pros: Enables for greater cohesion since there is a specific command for editing the 'watch' status of a show.
    - Cons: Requires longer code, and the code is also repetitive since its implementation is similar to that of the `EditCommand`.
- **Alternative 2:** Use the `WatchCommandParser` to create a new `EditCommand` object that edits the watch status of the show.
    - Pros: Less code repetition and shorter code in general.
    - Cons: This will mean that there is less cohesion of the code and greater dependencies since more classes depend on the `EditCommand` class.