Alana Huszar

Assignment 4

**Eight Golden Rules:**

1. *Strive for consistency*  
   I kept the location of the help/about links in the same place on each screen. I also
2. *Enable frequent users to use shortcuts.*  
   I will allow the use of tab to get from one search field to the next, as well as allow enter to be used in place of clicking the go button.
3. *Offer informative feedback.*  
   After clicking go, the user will either get a list of games, or will get a message saying no games match their search.
4. *Design dialog to yield closure.*  
   After updating their collection, a user will get a message informing them that their changes have been saved.
5. *Offer simple error handling.*  
   If the user attempt to enter invalid search terms, we will catch it and ask that they change their search.
6. *Permit easy reversal of actions.*  
   If the user is not presented with a list they like, they will be able to refine their search, which will bring them back to the previous screen without clearing what they have already entered.
7. *Support internal locus of control.*

Users will be able to select exactly what they want to search, and will only be shown results when they ask for them.

1. *Reduce short-term memory load.*

There are very few screens as part of this application, and informative descriptions are given for each search option.

**Techniques for Unit and Integration Testing:**As I create classes, I will test them as part of the unit testing process. As more modules are completed and tested, I will slowly integrate them together, and test them with the use of a driver (bottom-up testing). I am doing this because I have never really designed or created stubs before, so I think it will be more efficient to use drivers and test from the bottom of my code upwards. I will create the class for game objects, and then run that through several tests to see if it works properly. I will then test the collection class, as it contains games, and then I will continue to the user class. As I ascend the class hierarchy, I will cluster them together for my integration testing. I’ll do this throughout the project’s life, so that I can find problems early, and fix them without getting too stressed.

**Debugging/Testing:**I will use both debugging and testing tools. For debugging, I will be using Eclipse. I am using eclipse because it is available in the mac lab, and the wiki-tutorials reference it. I looked into other php debugging tools, but decided it would be wisest to use one that has a tutorial on the wiki, and that others know how to use. If I have time, I would like to perform security testing, but I am unsure at this time which tool will be most useful, as I do not know if I will be able to implement a user/password system by the end of the semester. I will also perform system testing as stated above. There are several programs for system testing php code, like phpUnit and codeception. I may look into these, or just use my own test methods, keeping track of all tests performed and their results.

|  |  |  |  |
| --- | --- | --- | --- |
| **Functionality Tested** | **Inputs** | **Expected Output** | **Actual Output** |
| Creation and usage of Game object | Name, Age, Time, Type, Players, Genre | Correct name, age, time, etc. |  |
| Handling of game object with null values | Name | Will only show up for queries without any search parameters |  |
| Searching | Desired restrictions | Games that satisfy the restrictions (and none that do not) |  |
| Invalid input into search field | “one” for number of players | An informative message asking user to change their formatting |  |
| Valid input, but no matching games | Restrictions that will result in no matching games | An informative message telling user that no games match, and that they should try another search |  |
| Results from other users | Desired restrictions | Games that satisfy the user’s restrictions, but without any overlap with the user’s collection |  |
| Refine search | Search restrictions, back button | Page should still show the restrictions the user originally entered, rather than reverting to the default static screen |  |
| Compare collections | Two different collections, with some similar games | List of all games that are not similar |  |
| Compare collections | Two different collections, with all the same games | An empty list of games |  |
| Search fields | Attempt to select/change all of the search options | Responsive buttons, and results matching the altered fields |  |
| Keyboard shortcuts | Tab, enter | Tab should allow the user to go from one option to the next, enter should allow the user to go to results without clicking go |  |
| Add a game | Game added to collection | Collection remains the same, except with the additional game |  |
| Remove a game | Game removed from collection | Collection remains the same, except without the removed game |  |
| Multiple search categories are selected | A mix of age, time, length, etc. parameters are searched upon | Games that satisfy ALL the parameters are shown, and no others |  |
| Age range | A range of ages is entered (i.e. 8-13) | Games that have an age interval containing the range are returned (i.e. a game with 6+ would be returned) |  |