CSc 690 Project 2.5
Image Browser with Auditory Icons and Annotations

Due 5pm Friday 9/29/2017
(standard 48 hour grace period for 75% of credit)
(10% of your grade)

This is an individual project. Work on your own.

Part a: Refactoring (15/100)

If necessary, reorganize/refactor your Image Browser source code according to these guidelines:

1. You code should follow the Model-View-Controller or Model-View design pattern. Hence, there should be one class for the data model, and one or two separate classes for the view and controller. See lecture slides for details on MVC or MV design patterns.
2. Each file should contain one class only!
3. Document each file according to the documentation guidelines from the Project 1 handout.

Part b: Specifying main display window width (10/100)

Allow the user to specify the width of the main display window with a command line parameter. Suppose the Python file containing the main class is Main.py. To specify the window width = W pixels (an integer), a user would type on the command line:

```
python3 Main.py W
```

The display window would still have aspect ratio 4 x 3. W may be between 600 and 1200. Your thumbnails and images should be scaled accordingly in the display window; larger windows would display larger thumbnails and full-screen images.

Part c: Auditory Icons (15/100)

Extend your image browser from Project 1.5 to support *auditory icons*, i.e., sounds that accompany user interface events.

Choose a short sound that will play when the user clicks on an image on the image bar, or types an arrow key, either in mode 0 or mode 1. Play the sound only when the keystroke results in a change in the display.

Choose a longer sound that will play when the user types the < or > key.

Part d: Text Annotations (60/100)

Each image has a list of text tags, stored in a file separate from the image. (You may create a separate tag file for each image, or combine all the image tags in the same file.) The tags are displayed below the current image in full-screen image display mode (Mode 1).

Add an editable QLineEdit and an ***Add Tag*** button to the display. When the user types text in the QLineEdit and clicks the *Add Tag* button, the tag will be added to the list of tags for the current image.

Add a ***Save All Tags*** button to the display. When the user clicks the *Save Tags* button, the updated list of tags will be saved. When the image browser is invoked again, the updated tags will be attached to each image.

You may design a custom format for the file(s) of tags. Just make sure tags can be added to each image, saved and displayed correctly.

A demo video for this project will be available shortly.

Programming style and documentation: Each source file should have a header documenting clearly the filename, author, date, with a description of the contents.

Submission: Submit a single zip file using the iLearn submission link.