Nathan Nickerson Tasks

= DONE

= Started, not complete

= Not started

~~~~~ASSIGNED TASKS~~~~~

-Week 1

 Set up the applet.

Load an image into the applet.

Display a loaded image into the applet.

Be able to load an image with JAI and manipulate the pixels

within the image.

* Interesting RenderedOP documentation: <http://docs.oracle.com/cd/E17802_01/products/products/java-media/jai/forDevelopers/jai-apidocs/javax/media/jai/RenderedOp.html>
* Documentation for PlanarImage: <http://docs.oracle.com/cd/E17802_01/products/products/java-media/jai/forDevelopers/jai-apidocs/javax/media/jai/PlanarImage.html>
* JAI API Tutorial (PDF)

<https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CC8QFjAA&url=http%3A%2F%2Fseer.ufrgs.br%2Frita%2Farticle%2Fdownload%2Frita_v11_n1_p93-124%2F3555&ei=Y8zdUY2jDMGmigKCtoDwDQ&usg=AFQjCNEY9Ze1dtZ2x6gTj7C4vtt0QT3h1A&sig2=AgDbWpvjptsJgyzzCc_DOw>

* Remember this: Each pixel has bands. A band represents R (red), G (green), or B (blue).

Add scroll bars to the applet to see the whole image.

Start manipulating a loaded image.

Figure out what color values red eye has.

- I couldn’t find set in stone red eye RGB values.

- I played around with a color chooser and found RGB values

that would make sense for the average person. If a person

had an eye disease, the red eye color would not be red.

~~(175, 0, 0) for the darkest red.~~

~~(255, 0, 0) for the purest red.~~

~~(255, 75, 75) for the minimal red.~~

(Document under the documentation directory with red eye

sample data.)

Create a bounding box for the pixels that contain the red eye

values.

Find the eye in a picture.

* Best way proposed so far is to click the eye and have the user dynamically change the size of the eye or oval.

~~Create a bounding box for the eye.(The user now dynamically chooses where the eye is along with the size of the eye.)~~

* + ~~Make the bounding box a different color than the bounding box for red eye colors.~~

----------------------------Goal for week 1 ends here----------------------------

~~Get the original eye color. (Not needed because red eye focuses inside of the pupil.~~

* Check to see if it is possible there’s some color on the edge of the red.
* Check to see if the other eye is not red. If not, grab that color.

~~Change the redness in the eye to the original color.~~

* Cannot change the red eye to the original color because red eye only exists in the pupil.

Correct the redness in the eye.

----------------------------Stretch goal for week 1 end here--------------------

Week 2 –

Correct the redness in the eye.

Find the eye in a picture.

* + Best way proposed so far is to click the eye and have the user dynamically change the size of the eye or oval.
* user dynamically change the size of the eye or oval.

~~Create a bounding box for the eye.(The user now dynamically chooses where the eye is along with the size of the eye.)~~

* + ~~Make the bounding box a different color than the bounding box for red eye colors.~~

Be able to draw in the applet.

* Hold down the left click and drag with drawing happening as the pointer moves through the applet.

Custom Brushes

* Instead of drawing a pixel, have a preset of pixels to be drawn anywhere the pointer chooses.

Make the ability to create lines.

Be able to bend the line created.

* Accomplish something cool!

Week 3 –

Create a logo for Inspiram.

* Now the icon of the applet.
* Changed the applet name to Inspiram as well.

Create a point-to-point line.

* Point to point means click the start of the line to set the beginning of the line, and then the second and last click will be the end of the line.

Continuing on the point-to-point line, create a Bezier curve with interaction from the user.

----------------Stretch goals for week 3----------------

Scale a loaded image to either a bigger or smaller size.

* Keeping the aspect ratio while not exceeding the height or width of the resolution of the users monitor display.

Understand what the different types of Anti-aliasing are.

* Different types include:
  + Anti-aliasing filter
  + Spatial anti-aliasing
  + Super sample anti-aliasing
  + Multi sample anti-aliasing
  + Temporal anti-aliasing

------------End of stretch goals for week 3------------

-Week 4

Finish the point-to-point line

-The end result will create a line from the users chosen

beginning and end points.

Create text into a picture

* The text should be visually visible and put into the graphic not overlayed.

Understand what the different types of Anti-aliasing are.

* Different types include:
  + Anti-aliasing filter
  + Spatial anti-aliasing
  + Super sample anti-aliasing
  + Multi sample anti-aliasing
  + Temporal anti-aliasing

---------------Week 4 Stretch Goals---------------

Paste an image over the current image.

Use ctrl+v to paste the image

---------------End Week 4 Stretch Goals---------------

~~~~~END OF ASSIGNED TASKS~~~~~

~~~~~UNASSIGNED TASKS~~~~~

1. Be able to restart the project.
2. Be able to center text on a graphic.
3. Put multiple graphics into my graphic.
4. Multiple copies (ctrl+c copies) stored to choose from later.
5. Keep track of the history as changes are made to an image.
6. Let the user create custom shapes or custom logos.
7. Add shadows to a 2D picture.
8. Work on multiple areas on a picture?
9. Have the ability to zoom for precision.
10. Draw straight lines.
11. Be able to turn on a “snapping” effect.
12. Create anti-aliasing for image resizing or for a smoother graphic.
13. Different stroke types for painting.
14. Make any graphic sharper or put different effects onto it.
15. Save as multiple file types.

~~~~~END OF UNASSIGNED TASKS~~~~~

~~~~~Things to keep in mind~~~~~

~~~~~End of things to keep in mind~~~~~

~~~~~Meetings~~~~~

1st Meeting: July 10, 2013 – Need to make more realistic goals with some stretches to know if the work is good enough. Need to re-word vague tasks for example: “Dig into JAI” could be, “Be able to load an image with JAI and manipulate the pixels within the image.”

* Need to put project files into the code directory and use .gitignore to ignore any unwanted files to commit.
* No status color for this week.

2nd Meeting: July 17, 2013 – 15 to 17hrs put in this week. Put in more hours next week. Learn to either step over, or walk around brick wall. Putting more time could have changed what I delivered this week. Think towards user stories (Agile) instead of specifications (Waterfall). Waterfall commits too much towards this and has to be accomplished in that specific way. Think out different solutions and why not try them. Commit to drawing a line and the Bezier curve manipulating the line. Make something cool! Finish the red eye and do not deliver something that paint could do (Spray Can). Trying to recreate something another application already does is okay for Capstone, but more innovation is needed in industry, and recreation is not as okay there.

* Hours: 15-17hrs.
* Status: Yellow (Room for improvement, but have not lost hope)
* Technicality for accomplished: Moderate
* Professionalism: Dressed well and minimal stutters.

3rd Meeting: July 25, 2013 at 11:30a.m. – This week was rough and I put in 13-14hrs. I showed off my red eye correction, but that is all I had. We discussed features. Which features should have more work on to stand out. Specific features should have a “wow” factor, or be built with quality. Should the red eye correction I have built be continued to get rid of the ring of fire and be smoothed with a nice transition outward? Small tasks are a great way to boost confidence and become more motivated. The harder tasks will have a higher reward and make my program that much cooler. The red eye correction is an accomplishment. A feature that was built can only motivate for so long. Think of buying a banana. The banana is nice and yellow and is great until you either eat it or it becomes rotten. Then the banana should be replaced or substituted for an apple. The apple would be a new feature or problem solved to feel accomplishment and motivate me more. When dealing with code for a more GUI specific application, people often code a lot more in very few classes. My code is following that trend and should sometime be abstracted from the class into various classes. My backlog was not correct. Keeping the task in week one for red eye and turning it green was a mistake. The red eye task(s) should have been copied into the next week for a better evaluation of the progress. The mistake is a harder judgment of what I had actually completed. My week was all red, but it should have had at least some green. Each week should be a whole new week and remain specific or altered only for that week. A week is its own week. Always come prepared with the next week’s tasks. Do something cool still like the Bezier curve, but find more motivation from it.

* Hours: 13-14Hrs.
* Status: Red (I believe I didn’t put forth the time needed which really hurt my productivity.)
* Technicality for accomplished: For what I accomplished with what I had, I’d say moderate. Whole week tasks thrown together would be minimal.
* Professionalism: Average person dressed. I felt more emotional this time. Not sure if that was noticeable. For some reason emotions felt overpowering.

4th Meeting: July 31, 2013 –

* Hours: 13hrs
* Status:
* Technicality: Not too technical. Procrastination stopped me from reaching this point for the week.
* Professionalism: Dressed up a bit.

~~~~~End of Meetings~~~~~