Journal #1 - Star Wars

Initial idea

I was recently rewatching "Star Wars" and I thought I could recreate my own movie poster. The concept is rather simple. I wanted it to have glistening stars and varying looming planets. However, there is a catch to it. I wanted to make something that was unique to every viewer by creating a galaxy full of stars and planets. Thus, I would be using various random values for most of the attributes.

Progress

Processing is something new to me. While I had used java before, I never did creative programming. In order to learn about this new tool, I read the book “Getting Started with Processing”.

Stars

While I did not yet finish the book, creating the stars was easily done. I would be creating an arbitrary number of stars at random points. The stars in the bottom half of the poster will be fading away relative to their y-axis. The more their y value is, the less bright they will become.

Planets

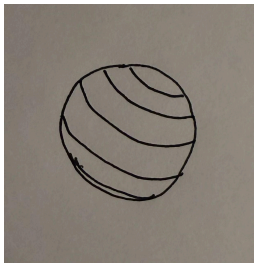
Exotic alien worlds are what makes star wars “Star Wars”. These unknown planets needed to stand out and make the viewer feel like they really are in a galaxy far, far away.
I created a class for the planets. It is designed so that every planet would have random size, color, and position.

Text

Now, all it needed was some outer galactic fonts to finish things up. I initiated the fonts according to the book and changed the mode so that the coordinates will be pointing at the center of the text. I made the texts so that it would be like the texts they use in the movies, where the text will get smaller and smaller as it gets close to the center.

Challenges I encountered

For the planets, I wanted to create arcs on the planets like this:



However, calculating the arcs' starting and ending point relative to their planet proved difficult. I managed to implement a drawArc() function, where it finds the y value of the arc using cosine function and the width of it using sine so that the starting and ending points will always be on the edge of the planet. Unfortunately, this function is buggy. I recommend you try it out.
After spending a long time on how I could implement this function better, I decided that since this is not a geometry class I will keep the planets filled with only one color. I will, however, plan to change the angle parameter to two parameters much like how arc is implemented in Processing, so that I don't have to manually calculate my angles.

The letters also proved somewhat tricky. While the original opening crawl had slanted text so that it is kind of like a trapezoid, mine does not. I believe I can somewhat manipulate the box that is around the text but that is something I will find out more in the future.

What I learned

I can confidently say that I learned:

- How to order different elements on top of eachother
- Use different strokes and colors
- Use basic shapes
- Change attributes of different elements
- Connect mathematics and art work

Source Code

Beware that you need the font I used. I will be including it in the folder.

```
import java.util.Random;
PFont font;

Random rand = new Random(); //Will be using this for random stars and planet coordinations

class Planet{
  float x;
  float y;
  int red;
  int green;
  int blue;
  float radius;

  Planet(){
    x = rand.nextInt(600) + 100;
    y = rand.nextInt(500) + 100;
    red = rand.nextInt(255);
    green = rand.nextInt(255);
    blue = rand.nextInt(255);
    radius = rand.nextInt(100) + 50;
  }
```

```
  Planet(float X, float Y, int Size, int r, int g, int b){
    x = X;
    y = Y;
    radius = Size;
    red = r;
    green = g;
    blue = b;
  }

  void display(){
    noStroke();
    fill(red, green, blue);
    ellipse(x, y, radius * 2, radius * 2);
  }


  void drawArc(float offset, float angle){
    strokeWeight(6);
    // Random stroke color
    stroke(rand.nextInt(255),rand.nextInt(255),rand.nextInt(255));
    // Calculate the width of the arc using radius
    float hyp = (float) Math.sin(angle) * radius;
    // Calculate the height of the arc using radius
    float h = (float) Math.cos(angle) * radius;
    noFill();
    arc(x, y-h, hyp*2, (h-offset)*2, 0.0, PI);
  }
}

// Creating randomized stars
void setStars(int lower, int higher){
  // Atleast "lower" stars but can be up to lower + higher
  int starCount = rand.nextInt(higher) + lower;

  for (int i=0; i < starCount; i++){
     stroke(255, 255);
     int x = rand.nextInt(width);
     int y = rand.nextInt(800);
     strokeWeight(rand.nextInt(4) + 3);
     // Selecting only stars that are in the bottom half
     if (500<y){
       float alpha =  Math.abs(y - 800);
       // Changing their alpha value relative to their y value, creating a gradient fade to black effect
       stroke(255, ((alpha *255)/300));
     }
     point(x, y);
  }
}
```

```
// Creating n number of randomized planets
void setPlanets(int n){
  for(int i=0; i<n; i++){
    Planet p1 = new Planet();
    p1.display();
    //p1.drawArc(50, PI/3);
    //p1.drawArc(80, PI/3.2);
    //p1.drawArc(100, PI/6);
  }
}

// Function used to write a text in the center
void typeText(String str, int size, int y){
  fill(247, 223, 5);
  textFont(font);
  textSize(size);
  textAlign(CENTER);
  text(str, width/2, y);
}

void setup() {
    size(800, 1000);
    background(0);
    setStars(400, 250);
    setPlanets(5);
    font = createFont("Starjedi.ttf", 32);
    typeText("Next summer", 10, 675);
    typeText("in theaters near you", 15, 690);
    typeText("Directed by Temuulen", 25, 720);
    typeText("Starring Benjamin Bacon", 33, 755);
    typeText("Star wars", 110, 860);
    typeText("Yes, disney made another one", 35, 900);
}
```

Final Outcome