```
//Nicholas Peters
//EECS 498-007 F10
//September 21st, 2010

//Grayscale Wheel and Color Wheel Constants
int segments = 12;
int steps = 8;
float rotation = TWO_PI / segments / 2;
float radius;
float segmentWidth;
float interval = TWO_PI / segments;
color wheelStroke = color(75);
float wheelStrokeWeight = 3;

//Grayscale Wheel Constants
float grayscaleX = 125;
float grayscaleY = 600;

//Color Wheel Constants
float colorX = 595;
float colorY = 600;

//Pen Variables
float penX = 360;
float penY = 600;
color penColor = color(255);
int penFlag = 0;

//Other Constants
int type = 1;
int weight = 1;
color helpStroke = color(55);

void setup() {
  drawWindow();
}

void drawWindow() {
  //Window
  size(720, 800);
  background(255);
  frameRate(600);

  drawInterface();
}

void drawInterface() {

  //Toolbar
  noStroke();
  fill(color(0));
  rect(0, 480, 720, 320);

  //Grayscale Wheel
  radius = 110;
  segmentWidth = radius / steps;

    //Background
    smooth();
    ellipseMode(RADIUS);
    stroke(wheelStroke);
    strokeWeight(wheelStrokeWeight*2);
    fill(color(0));
    ellipse(grayscaleX, grayscaleY, radius, radius);

    //Wheel
    smooth();
    ellipseMode(RADIUS);
    noStroke();
    drawGrayscaleWheel();

  //Color Wheel
  radius = 110;
  segmentWidth = radius / steps;

    //Background
    smooth();
    ellipseMode(RADIUS);
    stroke(wheelStroke);
    strokeWeight(wheelStrokeWeight*2);
    fill(color(0));
    ellipse(colorX, colorY, radius, radius);

    //Wheel
    smooth();
    ellipseMode(RADIUS);
    noStroke();
    drawColorWheel();

  //Font
  PFont font;

  //Text
  String help;
  font = createFont("Arial", 11);
  textFont(font);

  help = "1: Pen | 2: Variable Pen | 3: Variable Rectangle | 4: Variable Rectable w/
Stroke | 5: Variable Ellipse | 6: Variable Ellipse w/ Stroke";
  fill(helpStroke);
  textAlign(CENTER);
  text(help, 0, 740, 690, 15);

  help = "+: Increase Weight | -: Decrease Weight | DELETE: Reset Window | RETURN:
Save Window (as window.png)";
  fill(helpStroke);
  textAlign(CENTER);
  text(help, 0, 755, 690, 15);

  help = "Nicholas Peters | EECS 498-007 F10 | September 21st, 2010";
  fill(helpStroke);
  textAlign(CENTER);
  text(help, 0, 780, 690, 15);

  //Pen Wheel
  radius = 110;

  smooth();
  ellipseMode(RADIUS);
  stroke(wheelStroke);
  strokeWeight(wheelStrokeWeight);
  fill(penColor);
  ellipse(penX, penY, 55, 55);

  String display;
  font = createFont("Arial", 32);
  textFont(font);

  display = "Weight: "+weight;
  fill(helpStroke);
  textAlign(CENTER);
  text(display, 235, 490, 250, 50);

}

void draw() {
  //Radius
  radius = 110;

  //Mouse Pressed
  if(mousePressed) {
    if(overWheel(grayscaleX, grayscaleY, radius) || overWheel(colorX, colorY, 
radius)) {
      if(penFlag == 0) {
        penColor = get(mouseX, mouseY);
      }
    }
    else {
      if (mouseY < 480) {
        switch(type) {
          case 1:
            invariableLine(mouseX, mouseY, pmouseX, pmouseY);
            break;
          case 2:
            variableLine(mouseX, mouseY, pmouseX, pmouseY);
            break;
          case 3:
            variableRect(mouseX, mouseY, pmouseX, pmouseY);
            break;
          case 4:
            variableRectWithStroke(mouseX, mouseY, pmouseX, pmouseY);
            break;
          case 5:
            variableEllipse(mouseX, mouseY, pmouseX, pmouseY);
            break;
          case 6:
            variableEllipseWithStroke(mouseX, mouseY, pmouseX, pmouseY);
            break;
          default:
            break;
        }

        penFlag = 1;
      }
    }
  }

  //Key Pressed
  if(keyPressed) {
    if(key == BACKSPACE || key == DELETE) {
      drawWindow();
      penColor = color(255);
      weight = 1;

    }
    if(key == ENTER || key == RETURN) {
      save("window.png");
    }
    if(key == '+' || key == '=') {
      if(weight < 50) {
        weight += 1;
      }
      else {
        weight = 50;
      }
    }
    if(key == '-' || key == '_') {
      if(weight > 1) {
        weight -= 1;
      }
      else {
        weight = 1;
      }
    }
    if(key == '1') {
      type = 1;
    }
    if(key == '2') {
      type = 2;
    }
    if(key == '3') {
      type = 3;
    }
    if(key == '4') {
      type = 4;
    }
    if(key == '5') {
      type = 5;
    }
    if(key == '6') {
      type = 6;
    }
  }

  drawInterface();
}

void drawGrayscaleWheel() {
  for (int i = 0; i < steps; i++) {
    color[] columns = {
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i),
      color(255-(255/steps)*i)
    };
    for (int j = 0; j < segments; j++) {
      fill(columns[j]);
      arc(grayscaleX, grayscaleY, radius, radius,
          interval*j+rotation, interval*(j+1)+rotation);
    }
    radius -= segmentWidth;
  }
}
```

```
void drawColorWheel() {
  for (int i = 0; i < steps; i++) {
    color[] columns = {
      color(255-(255/steps)*i, 255-(255/steps)*i, 0),
      color(255-(255/steps)*i, (255/1.5)-((255/1.5)/steps)*i, 0),
      color(255-(255/steps)*i, (255/2)-((255/2)/steps)*i, 0),
      color(255-(255/steps)*i, (255/2.5)-((255/2.5)/steps)*i, 0),
      color(255-(255/steps)*i, 0, 0),
      color(255-(255/steps)*i, 0, (255/2)-((255/2)/steps)*i),
      color(255-(255/steps)*i, 0, 255-(255/steps)*i),
      color((255/2)-((255/2)/steps)*i, 0, 255-(255/steps)*i),
      color(0, 0, 255-(255/steps)*i),
      color(0, 255-(255/steps)*i, (255/2.5)-((255/2.5)/steps)*i),
      color(0, 255-(255/steps)*i, 0),
      color((255/2)-((255/2)/steps)*i, 255-(255/steps)*i, 0)
    };
    for (int j = 0; j < segments; j++) {
      fill(columns[j]);
      arc(colorX, colorY, radius, radius,
          interval*j+rotation, interval*(j+1)+rotation);
    }
    radius -= segmentWidth;
  }
}

boolean overWheel(float wheelX, float wheelY, float wheelRadius) {
  float distanceX = wheelX - mouseX;
  float distanceY = wheelY - mouseY;
  if(sqrt(sq(distanceX) + sq(distanceY)) < wheelRadius) {
    return true;
  }
  else {
    return false;
  }
}

void invariableLine(int x, int y, int px, int py) {
  float speed = abs(x-px) + abs(y-py);
  stroke(penColor);
  strokeWeight(weight);
  line(x, y, px, py);
}

void variableLine(int x, int y, int px, int py) {
  float speed = abs(x-px) + abs(y-py);
  stroke(penColor);
  strokeWeight(speed);
  line(x, y, px+weight, py+weight);
}

void variableRect(int x, int y, int px, int py) {
  float speed = abs(x-px) + abs(y-py);
  noStroke();
  fill(penColor);
  rect(x, y, speed+weight, speed+weight);
}

void variableRectWithStroke(int x, int y, int px, int py) {
  float speed = abs(x-px) + abs(y-py);
  stroke(color(255));
  strokeWeight(1);
  fill(penColor);
  rect(x, y, speed+weight, speed+weight);
}

void variableEllipse(int x, int y, int px, int py) {
  float speed = abs(x-px) + abs(y-py);
  noStroke();
  fill(penColor);
  ellipse(x, y, speed+weight, speed+weight);
}

void variableEllipseWithStroke(int x, int y, int px, int py) {
  float speed = abs(x-px) + abs(y-py);
  stroke(color(255));
  strokeWeight(1);
  fill(penColor);
  ellipse(x, y, speed+weight, speed+weight);
}

void mouseReleased() {
  penFlag = 0;
}
```