Xiaohan Zhang

Programming with Processing

What is Processing?

- http://processing.org/
- Language for interaction and visuals
- Java applet/application, JS port
- Get it at processing.org/download/

While we wait…

- Exhibition at processing.org/exhibition/
  - Platonic Solids
  - BallDroppings
- Some demos at openprocessing.org/
  - http://openprocessing.org/visuals/?visualID=1210

First Steps…

- void setup() {

      size(800, 600);
      background(128);
      smooth();

  }
- void draw() {}
- Press control+R
- Escape to quit

Polygons man, how do they work?

- ellipse(x, y, diameterX, diameterY);
- line(x1, y1, x2, y2);
- rect(x, y, width, height);
- point(x, y);

Help -> Reference

Colors man, HOW DO THEY WORK?!

- color(r, g, b); color(r, g, b, alpha);
- color(gray); color(gray, alpha);
- stroke(color);
  - noStroke();
- fill(color);
  - noFill();

Help -> Reference

VARIABLES?!

- mouseX, mouseY, pmouseX, pmouseY
- mousePressed, mouseButton
- keyPressed, key, keyCode
- width, height, frameCount

Help -> Reference

EVENTS!!!!!!!!!!!!!!!!!!!

- void keyPressed() {}
- void keyReleased() {}
- void keyTyped() {}
- void mousePressed() {}
- void mouseDragged() {}
- void mouseReleased() {}
- void mouseClicked() {}
- void mouseMoved)() {}

HOOOLYYYY SHIIITTTTTTT

- millis()
- map(x, inLow, inHigh, outLow, outHigh)
- dist(x1, y1, x2, y2)
- noise(x [, y[, z[, w]]])
- colorMode(HSB or RGB);
- strokeWeight(pixels);

New Horizons

- new sketch
  - smooth();
- strokeWeight(5);
- point(mouseX, mouseY);
  - in mouseDragged()

Classes for the Masses

```
public class Point {
    float x, y;
    public Point(float x, float y) {
        this.x = x;
        this.y = y;
    }

    public void draw() {
        point(x, y);
    }
}
registerDraw(new Point(mouseX, mouseY));
```

I got da jitterz

```
public void draw() {
    jitter();
    point(x, y);
}
void jitter() {
    x += random(-2, 2); y += random(-2, 2);
}

fill(255, 2); rect(0, 0, width, height);
```

Moths to the flame

```
public void draw() {
    jitter();
    attract();
    point(x, y);
}
void attract() {
  float dx = mouseX - x,
        dy = mouseY - y,
        dist = dist(0, 0, dx, dy);
  x += dx / dist;
  y += dy / dist;
}
```

Derivatives and shit

```
float dx, dy;
public void draw() {
    jitter();
    attract();
    x += dx;
    y += dy;
    point(x, y);
}
```

```
void attract() {
   float dx = mouseX - x,
         dy = mouseY - y,
         dist = dist(0, 0, dx, dy);
   this.dx += dx / dist;
   this.dy += dy / dist;
 }
```

All the colors of the rainbow!

```
public void draw() {
    jitter();
    attract();
    float xn = x, yn = y;
    x += dx;
    y += dy;
    colorMode(HSB);
    stroke(map(atan2(dy, dx), -PI, PI, 0, 255), 255,
    dist(0, 0, dx, dy) * 50);
    line(x, y, xn, yn);
}
```

At your command

```
void mouseDragged() {
  if(mouseButton == RIGHT)
    registerDraw(new Point(mouseX, mouseY));
}

public void draw() {
  jitter();
  if(mousePressed && mouseButton == LEFT)
    attract();
  float xn = x, yn = y;
  x += dx *= .98f;
  y += dy *= .98f;
  colorMode(HSB);
  stroke(map(atan2(dy, dx), -PI, PI, 0, 255), 255, dist(0, 0, dx, dy) * 50);
  line(x, y, xn, yn);
}
```

Do it ourselves

```
List<Point> points = new ArrayList();

void draw() {
  fill(255, 2);
  rect(0, 0, width, height);
  for(Point l : points) {
    l.draw();
  }
}

void mouseDragged() {
     points.add(new Point(mouseX, mouseY));
}
```

Fade to black

```
void draw() {
  background(0);
  for(Point l : points) {
    l.draw();
  }
}
```

Know your neighbors

```
Set<Point> within(float rad) {
  Set<Point> s = new HashSet();
  for( Point pt : points) {
    if(pt == this) continue;
    if(dist(x, y, pt.x, pt.y) < rad) s.add(pt);
  }
  return s;
}
```

… put this inside draw

```
  stroke(255);
  for(Point p : within(10)) {
    line(x, y, p.x, p.y);
  }
```