# Creative Coding

Professor Danne Woo

creativecode.dannewoo.com

ARTS 249

Spring 2020

Thursday 2:00 PM – 5:50 PM

I-Building 213

# Motion and Interaction

Week 08: Kinetic Forms

Week 09: Drawing and Interaction (Mouse)

Week 10: Interaction and Image Import (Keyboard and Events)

Week 11: 3D

Week 12: Audio

Week 13: Computer Vision

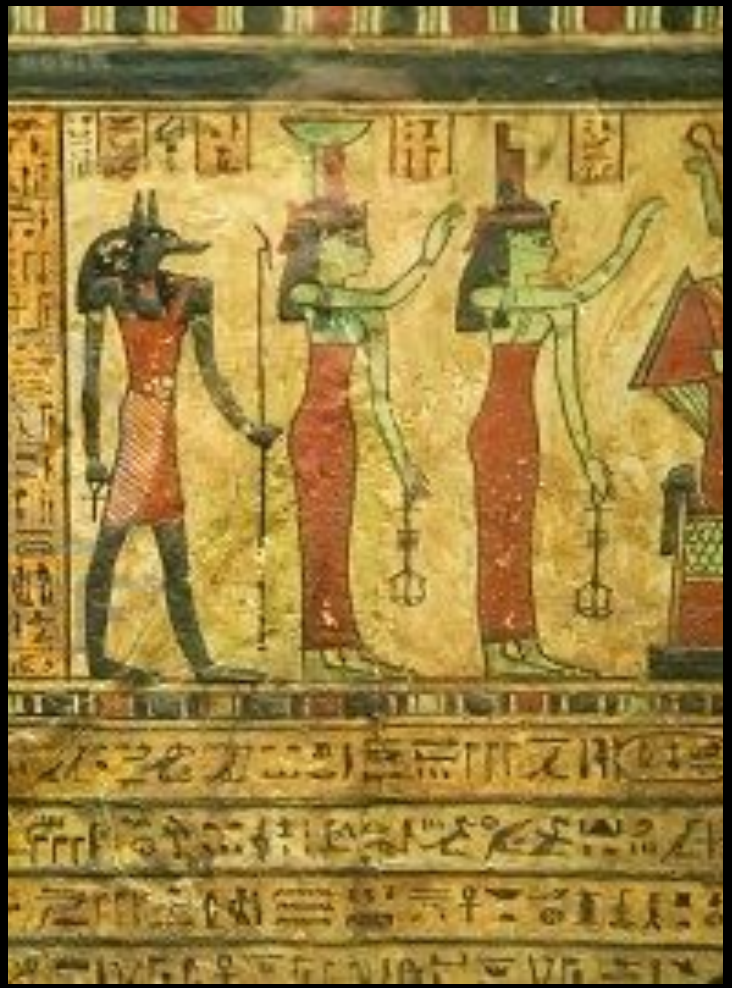
Week 14: Final Presentation

# History of Drawing

El Castillo Cave

# History of Drawing

Egyptian Hieroglyphics

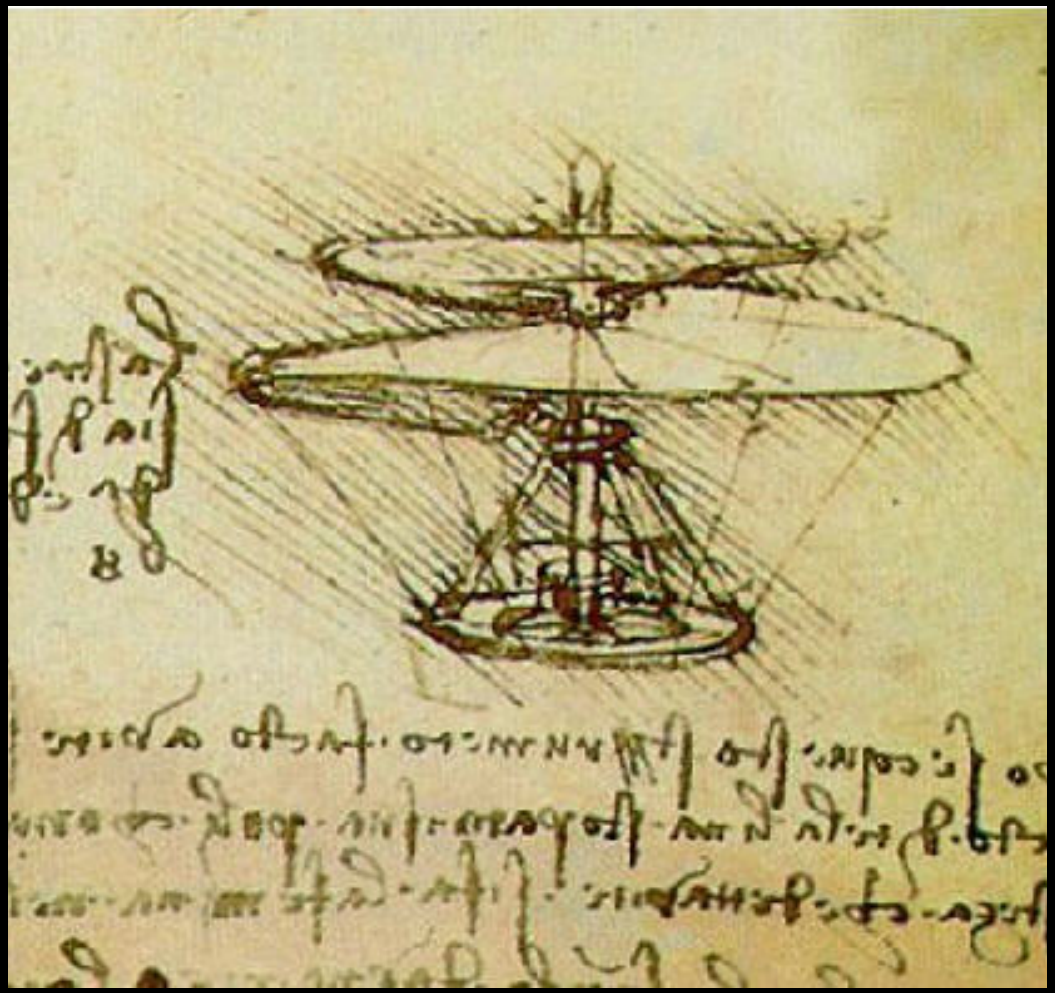


# History of Drawing

Middle Ages - Illuminated Manuscripts

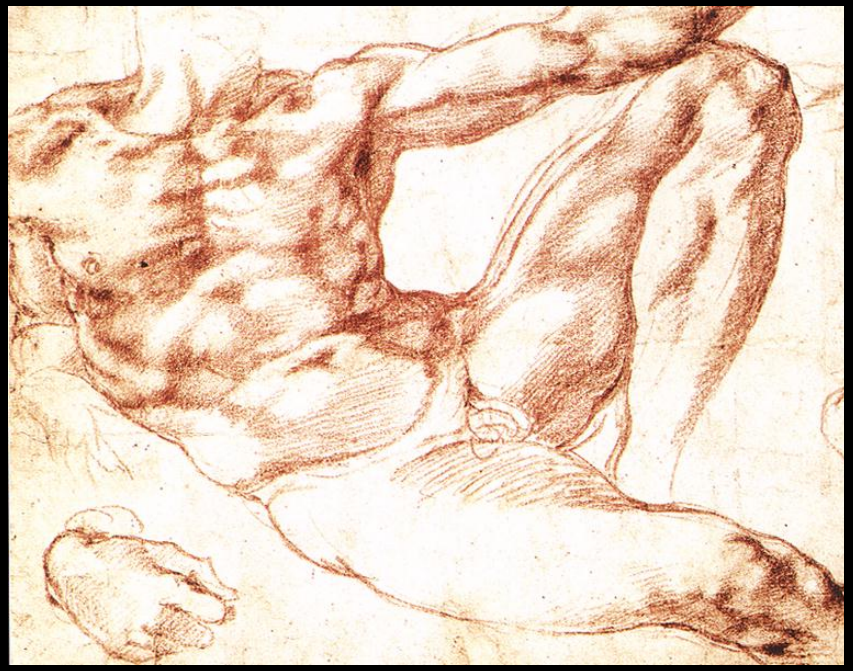
# History of Drawing

Renaissance – Da Vinci Sketches



# History of Drawing

Renaissance – Michelangelo Sistine Chapel Sketches



# History of Drawing

Renaissance - Chalk, Charcoal and Pen & Ink

# History of Drawing

Baroque and Rococo - 1600 and 1700s

# History of Drawing

Graphite Pencil - 1800 and 1900s

# History of Drawing

Edgar Degas - 1800 and 1900s

# History of Drawing

1800 and 1900s

# History of Drawing

Personal Computers, 1977

# History of Drawing

Apple II, Lisa and WIMP, 1984

# History of Drawing

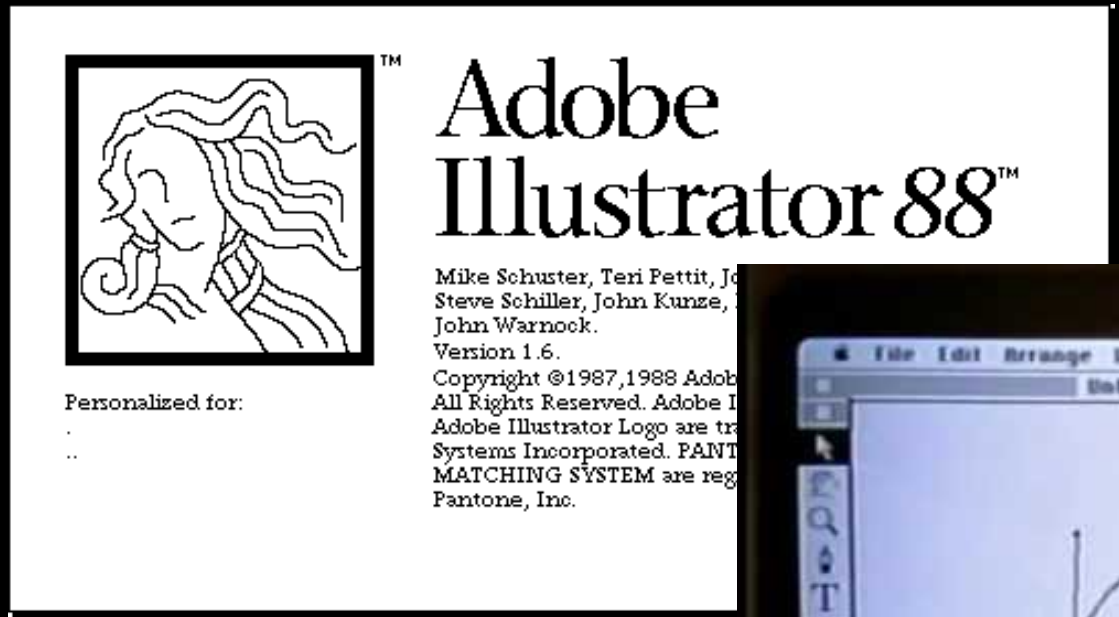
QuickDraw and MacPaint, early 80s

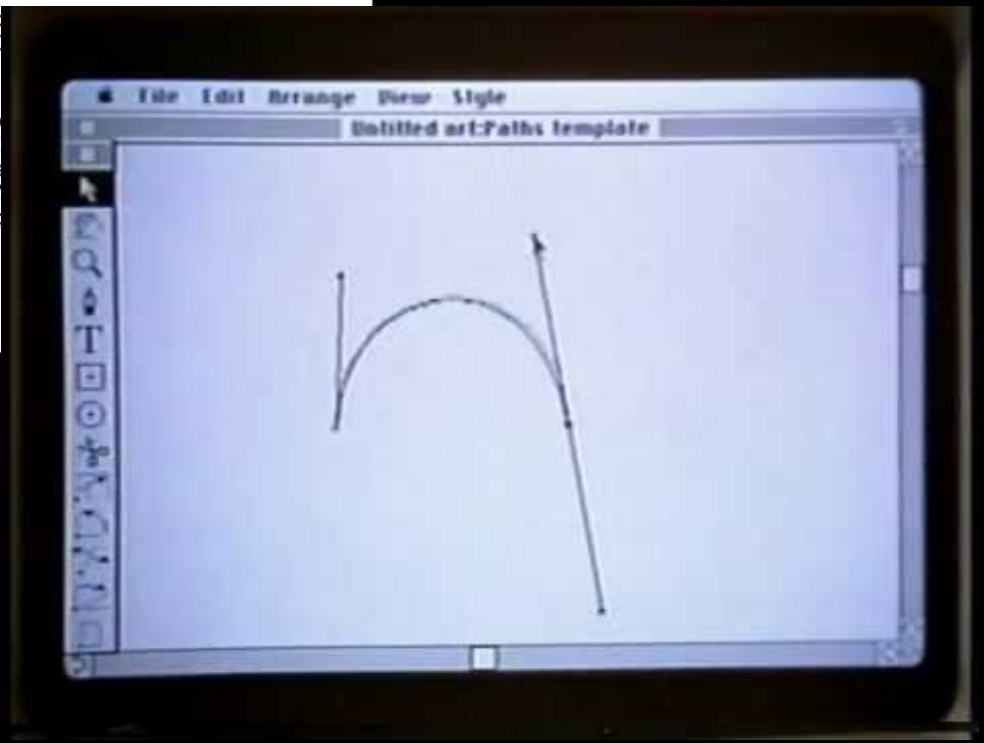
# History of Drawing

Apple and Artists

# History of Drawing

Adobe Illustrator



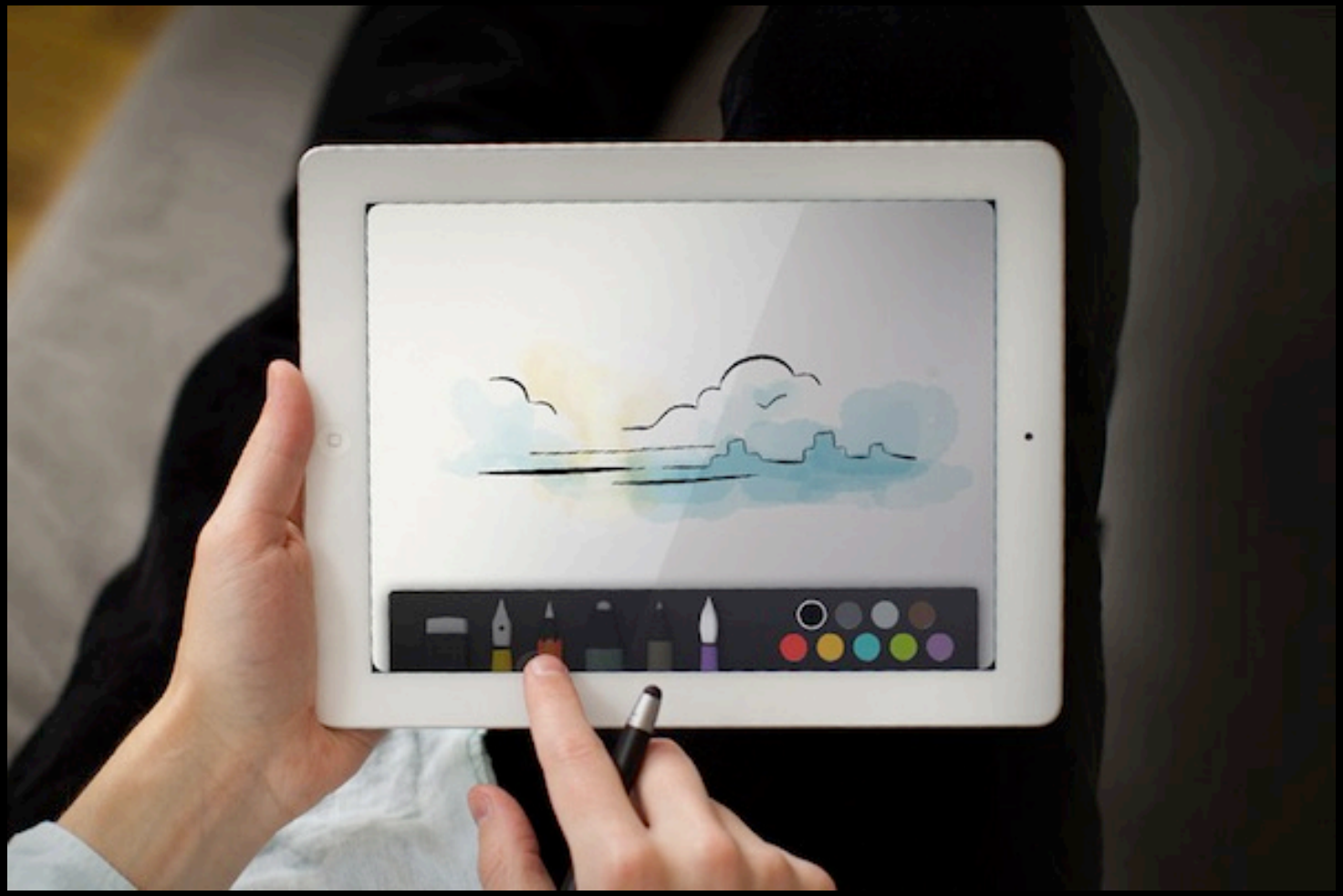
# History of Drawing

Adobe Illustrator


https://vimeo.com/95415863

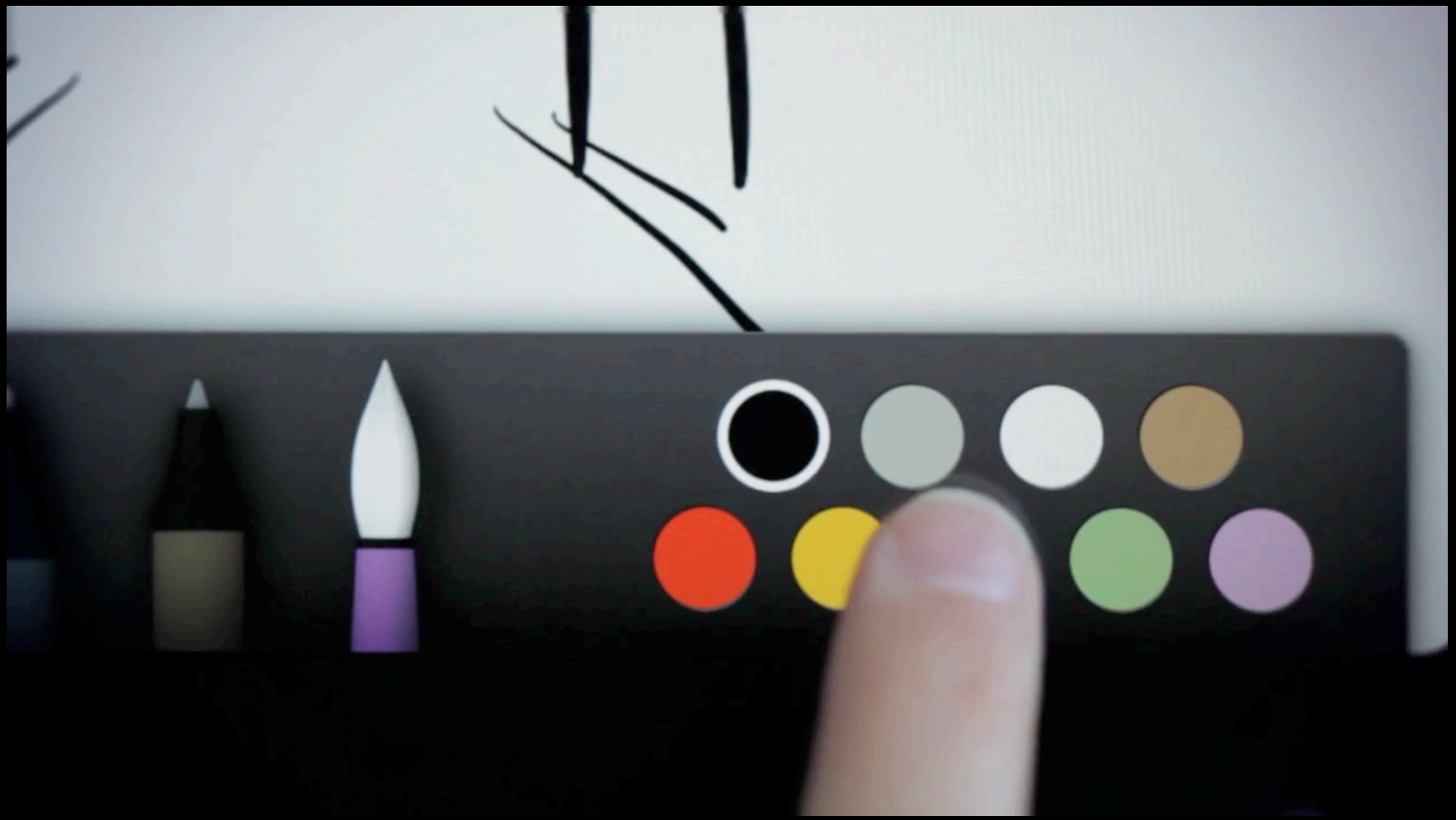
# History of Drawing

Tablets



# History of Drawing

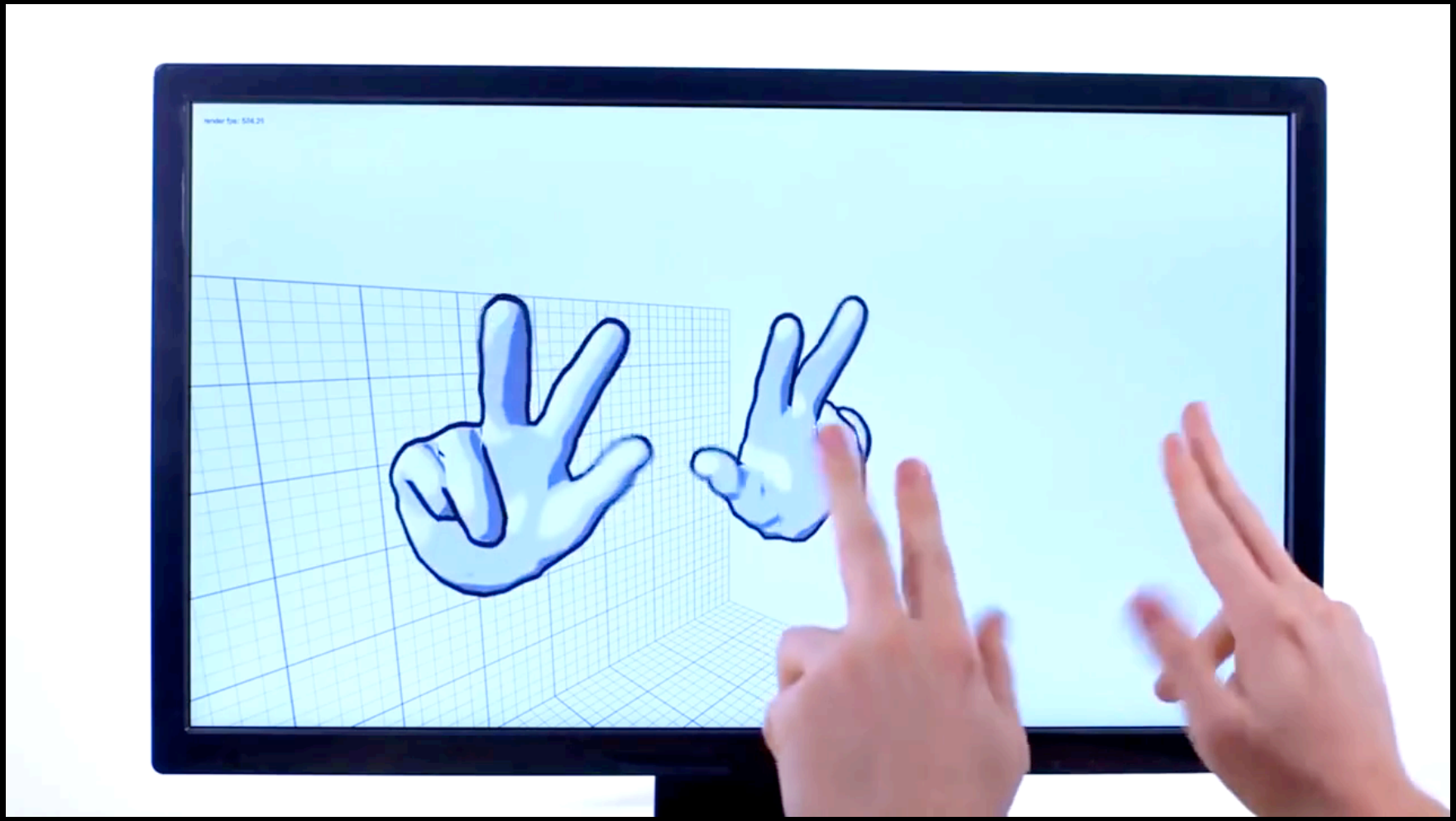
Paper App



https://www.fiftythree.com/think

# History of Drawing

Leap Motion



https://www.youtube.com/watch?v=_d6KuiuteIA

# History of Drawing

Virtual Reality

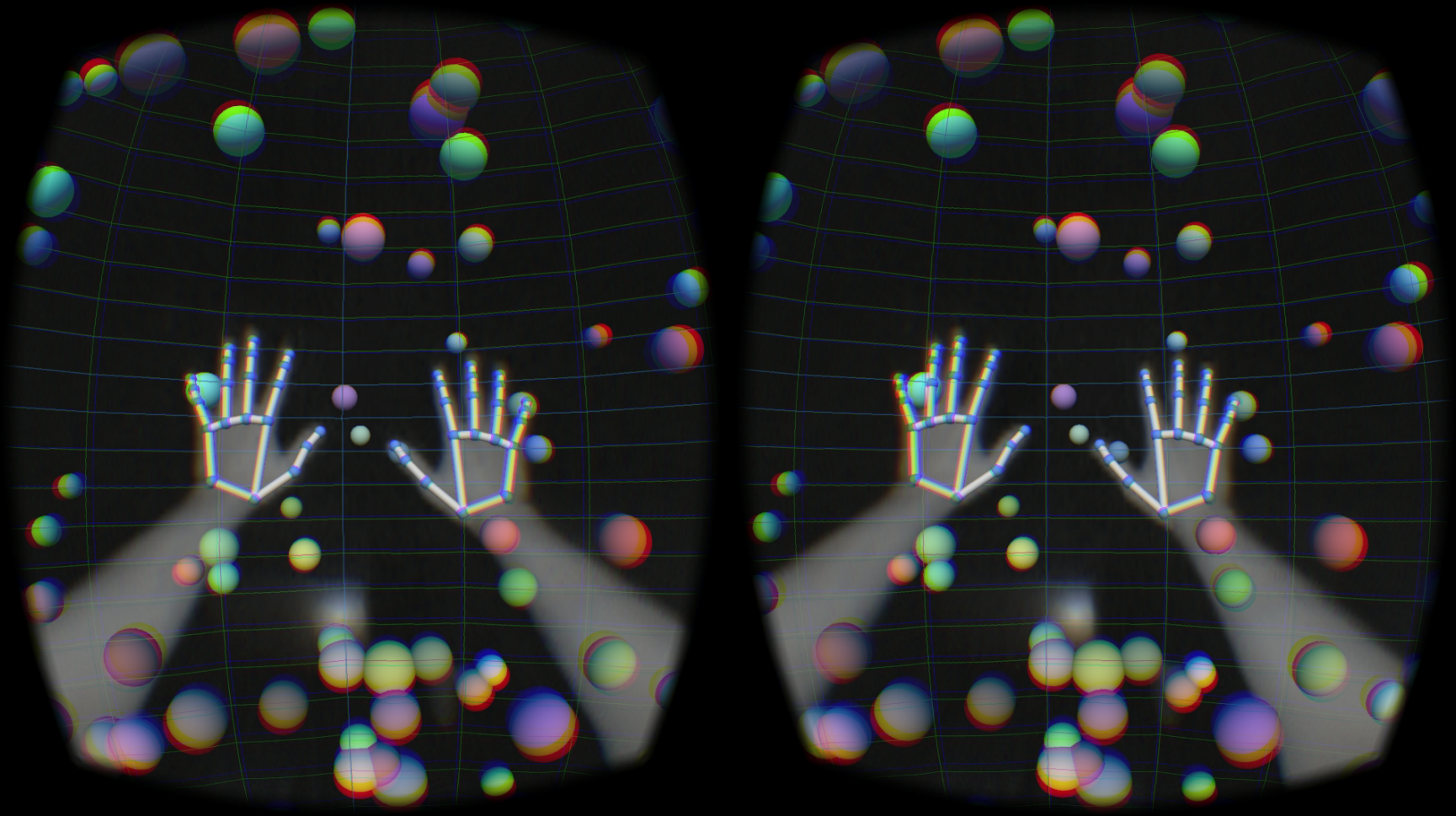
# History of Drawing

Virtual Reality / Tilt Brush

https://www.youtube.com/watch?v=TckqNdrdbgk

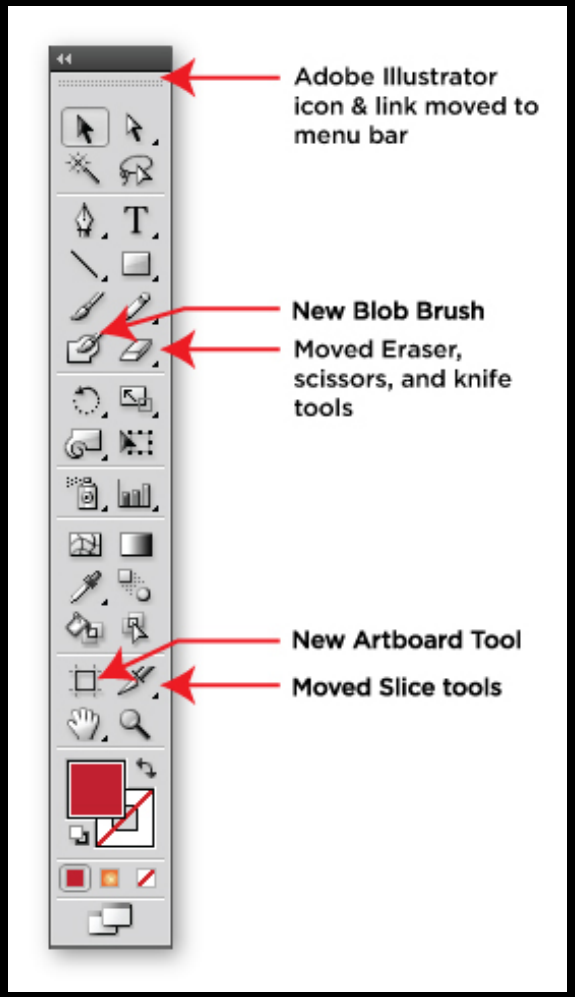
# History of Drawing

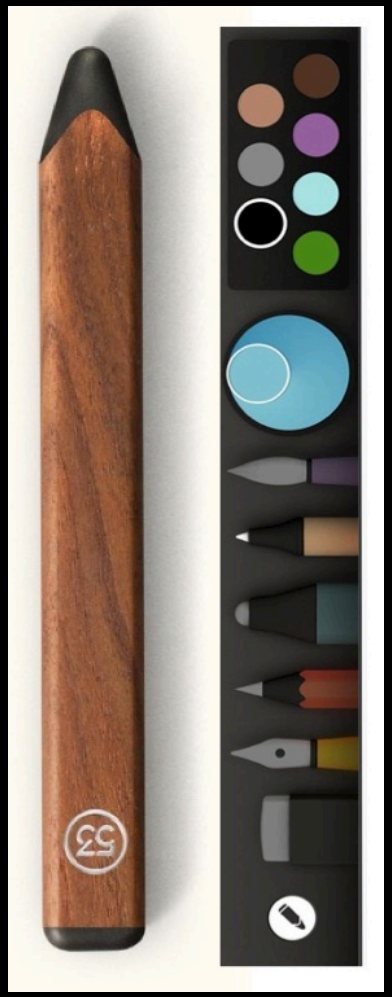
Oculus + Leap



# History of Drawing

Limited Tools



# Drawing and Code

Eno Henze – Red Ambush

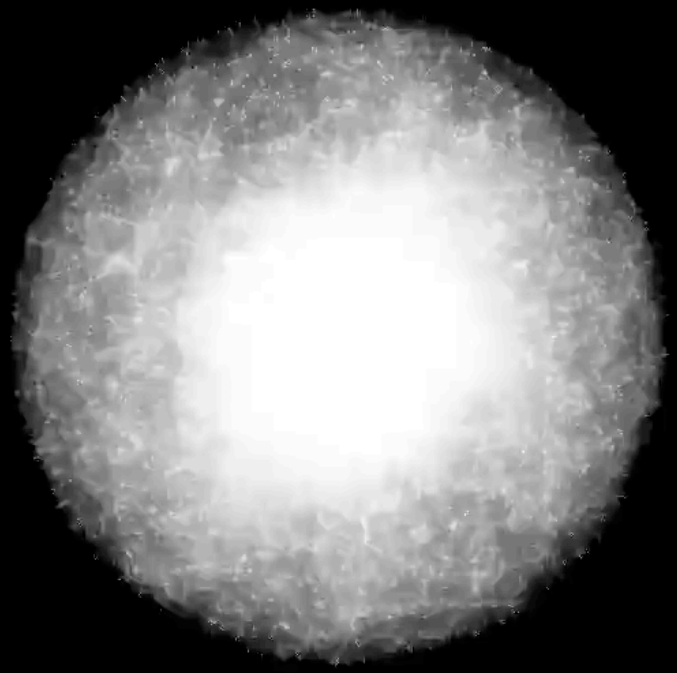
# Drawing and Code

Field – 10,000 Digital Paintings

https://www.field.io/project/digital-paintings/

# Drawing and Code

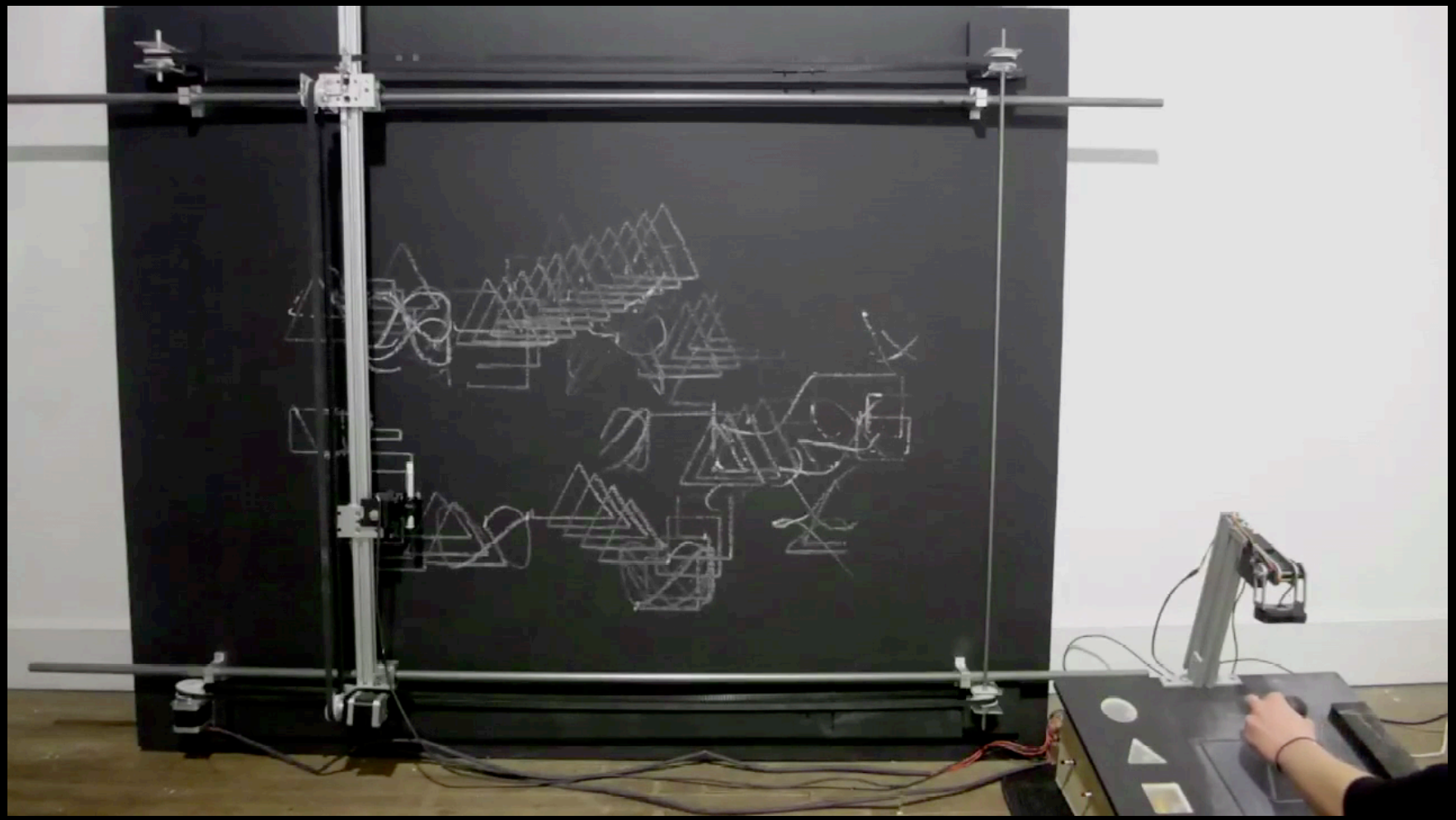
Kyle McDonald – Binned Particle System



https://vimeo.com/7890507

# Drawing and Code

Claire Mitchell – Primitives



http://cargocollective.com/clairemitchell/Primitives

# Mouse Interaction

**Mouse Variables** – mouseX, mouseY, pmouseX, pmouseY

**Mouse Interaction Functions** – mousePressed(), mouseReleased(), mouseDragged()

**Booleans** – True or False statements, mouseIsPressed, mouseButton

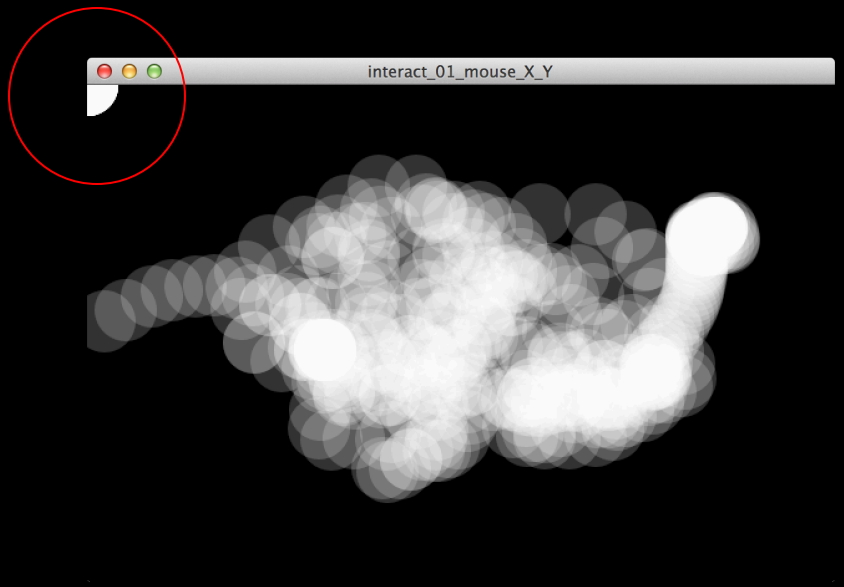
**Mouse Functions** – noCursor(), cursor(), map(), constrain(), dist(), save()

**Math Methods** – Hover State, Easing and Speed Detection

# Mouse Interaction

## mouseX and mouseY

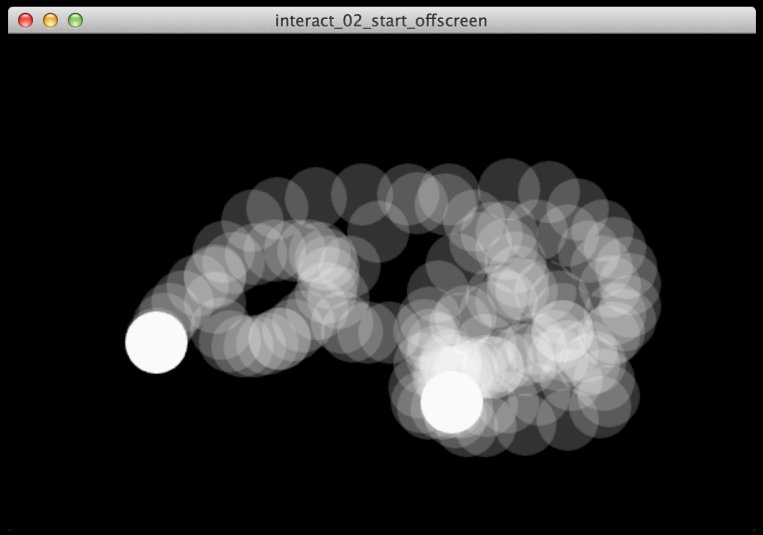
```
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
}
function draw() {
  fill(255, 50);
  ellipse(mouseX, mouseY, 50, 50);
}
```



# Mouse Interaction

## Start Off Screen

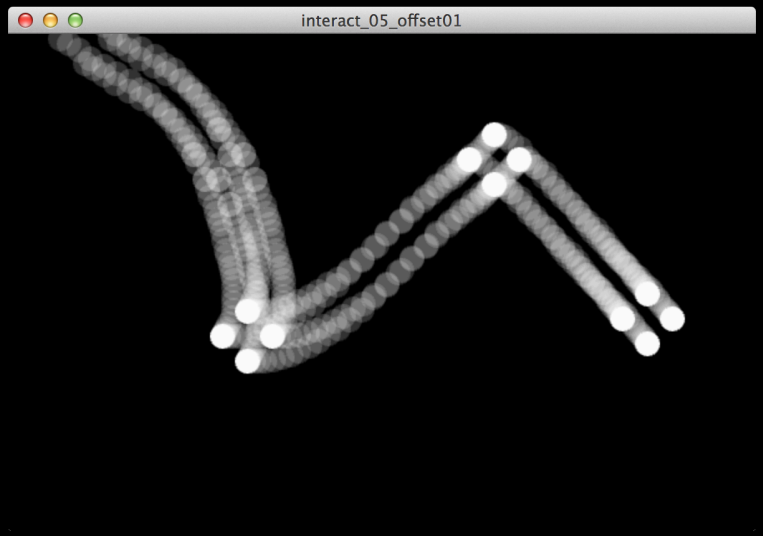
```
var d = 50;
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
  mouseX = -d;
  mouseY = -d;
}
function draw() {
  fill(255, 50);
  ellipse(mouseX, mouseY, d, d);
}
```



# Mouse Interaction

## Offset

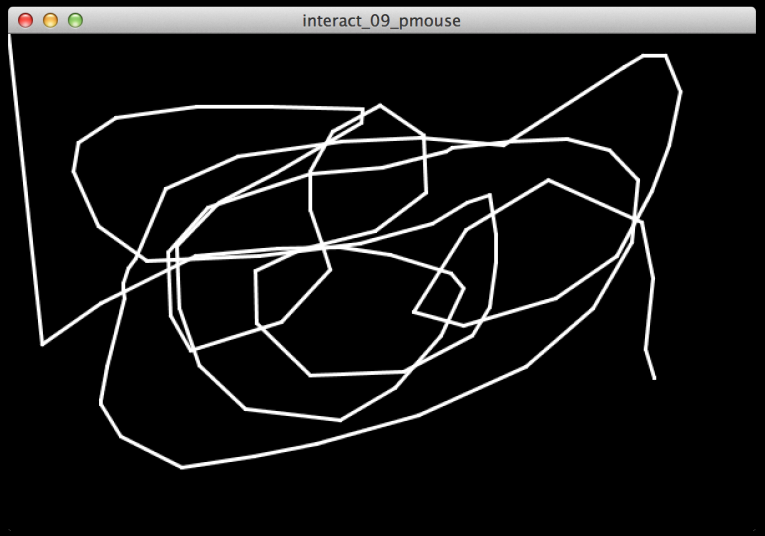
```
var d = 20;
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
  mouseX = -d*2;
  mouseY = -d*2;
}
function draw() {
  fill(255, 50);
  ellipse(mouseX + d, mouseY, d, d);
  ellipse(mouseX - d, mouseY, d, d);
  ellipse(mouseX, mouseY + d, d, d);
  ellipse(mouseX, mouseY - d, d, d);
}
```



# Mouse Interaction

pmouseX and pmouseY

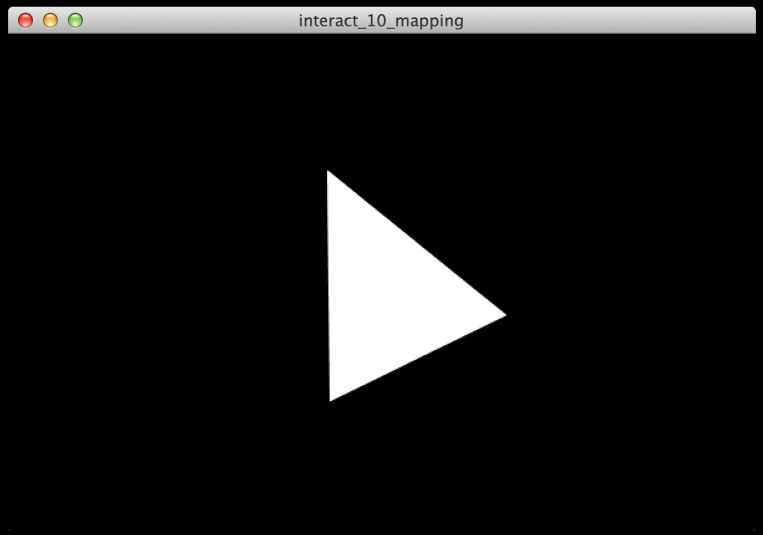
```
function setup() {
  createCanvas(600, 400);
  background(0);
  stroke(255);
  strokeWeight(3);
  frameRate(10);
}

function draw() {
  line(mouseX, mouseY, pmouseX, pmouseY);
}
```



# Mouse Interaction

## Mapping

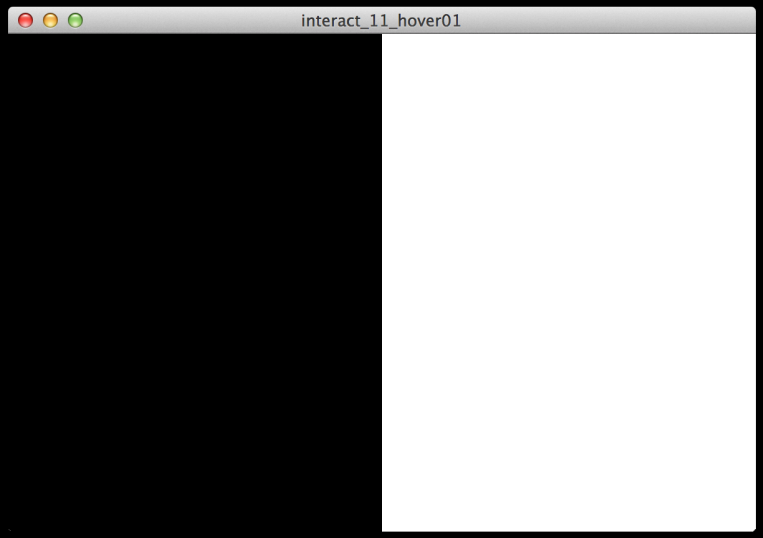
```
var angle;

function setup() {
  createCanvas(600, 400);
  background(0);
  stroke(255);
  strokeWeight(5);
}

function draw() {
  background(0);
  angle = map(mouseX, 0, width, 0, TWO_PI);
  translate(width/2, height/2);
  rotate(angle);
  triangle(0, -95, 75, 65, -75, 65);
}
```



# Mouse Interaction

## Hover

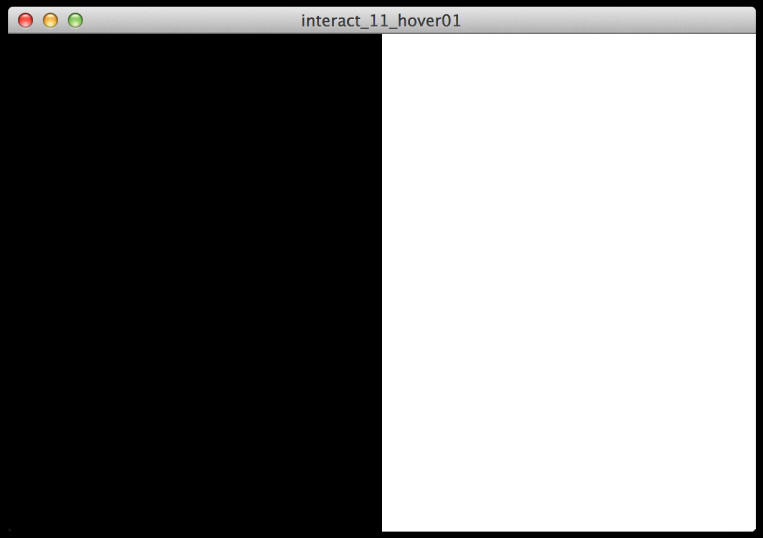
```
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
}

function draw() {
  background(0);
  fill(255);
  if (mouseX < width/2) {
    rect(0, 0, width/2, height);
  } else {
    rect(width/2, 0, width/2, height);
  }
}
```



# Mouse Interaction

## boolean

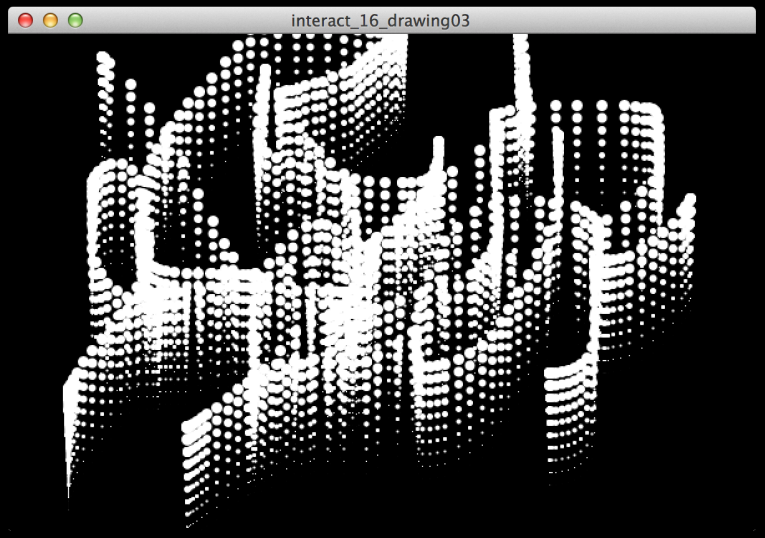
```
var hoverBoolean = true;
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
}
function draw() {
  background(0);
  fill(255);
  if (mouseX < width/2) hoverBoolean = true;
  else hoverBoolean = false;
  if (hoverBoolean == true) {
    rect(0, 0, width/2, height);
  } else if (hoverBoolean == false){
    rect(width/2, 0, width/2, height);
  }
}
```



# Mouse Interaction

## mousePressed

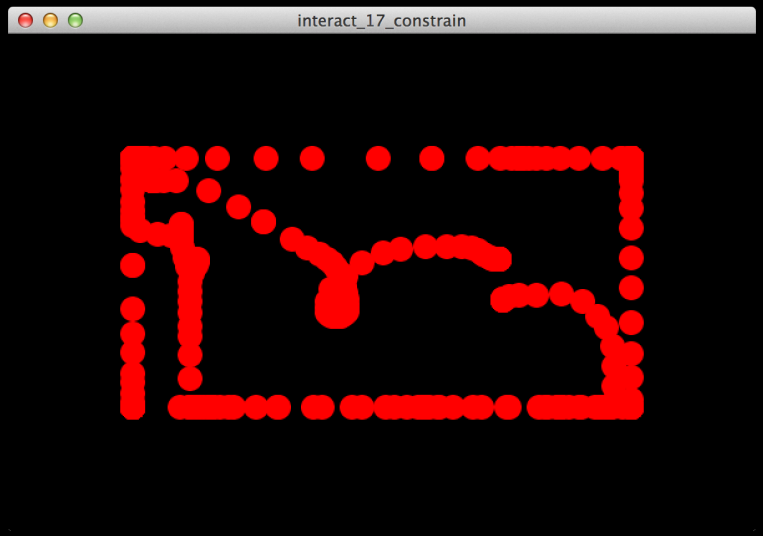
```
function setup() {
  createCanvas(600, 400);
  background(0);
  stroke(255);
}

function draw() {
  if (mouseIsPressed) {
    translate(mouseX, mouseY);
    for(var i = 0; i < 10; i++) {
      strokeWeight(i);
      point(0, i * -10);
    }
  }
}
```



# Mouse Interaction

constrain()

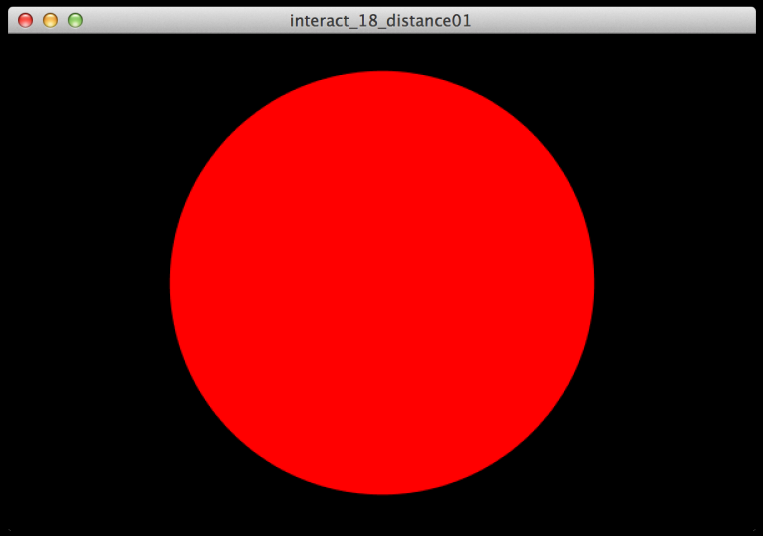
```
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
}

function draw() {
  var x = constrain(mouseX, 100, 500);
  var y = constrain(mouseY, 100, 300);
  if (mouseIsPressed) {
    fill(255, 0, 0);
    ellipse(x, y, 20, 20);
  }
}
```



# Mouse Interaction

dist()

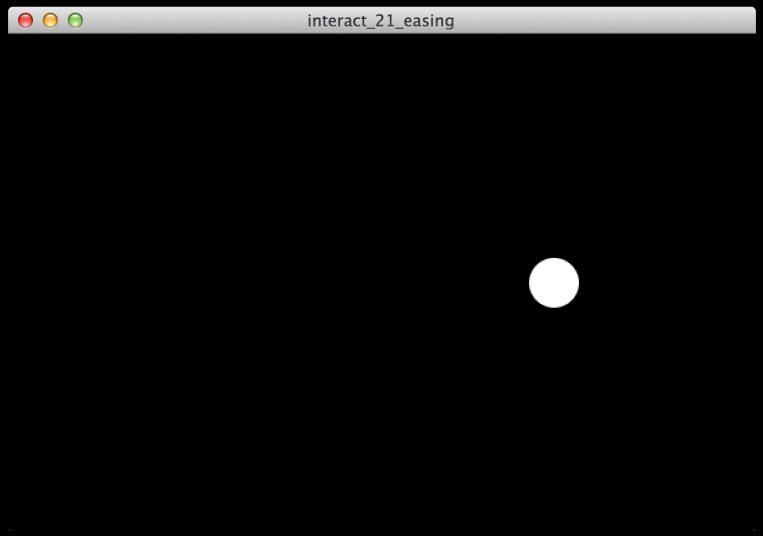
```
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
  fill(255, 0, 0);
}

function draw() {
  background(0);
  var d = dist(width/2, height/2, mouseX, mouseY);
  ellipse(width/2, height/2, d*2, d*2);
}
```



# Mouse Interaction

## Easing

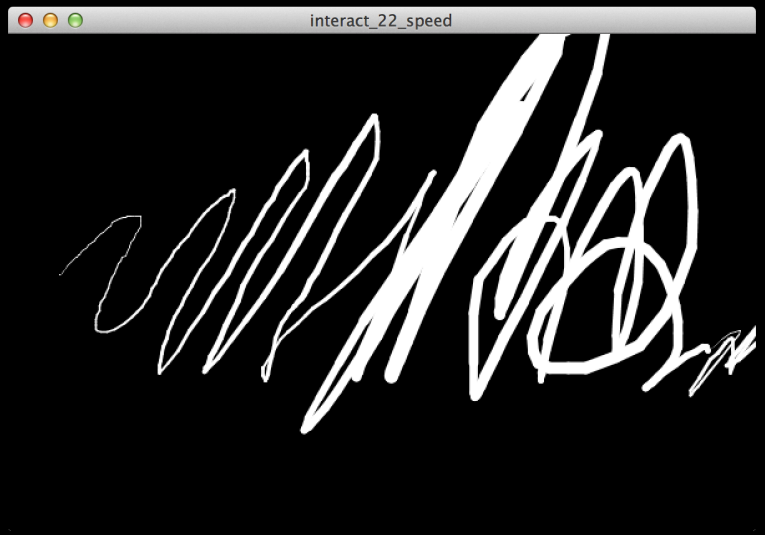
```
var x = 0;
var easing = 0.07;

function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
}

function draw() {
  background(0);
  fill(255);
  var targetX = mouseX;
  x += (targetX - x) * easing;
  ellipse(x, height/2, 40, 40);
}
```



# Mouse Interaction

## Speed

```
var speed = 0;
var easing = 0.03;
var x, y;
function setup() {
  createCanvas(600, 400);
  background(0);
  stroke(255);
}
function draw() {
  var target = dist(mouseX, mouseY, pmouseX, pmouseY);
  speed += (target - speed) * easing;
  if(mouseIsPressed){
    strokeWeight(speed/2);
    line(mouseX, mouseY, pmouseX, pmouseY);
  }
}
```



# Mouse Interaction

## save()

```
var count = 1;
var c;
function setup(){
  c = createCanvas(600, 400);
  background(255);
  noStroke();
}
function draw() {
  if (mouseIsPressed) {
    fill(255, 0, 0, 40);
    ellipse(mouseX, mouseY, 30, 30);
  }
}
function keyPressed() {
 if (keyCode == 80) {
    saveCanvas(c, 'canvasName' + count, 'jpg');
    count++;
  }
}
```

# Mouse Interaction

## save()

```
var count = 1;
var c;
function setup(){
  c = createCanvas(600, 400);
  background(255);
  noStroke();
}
function draw() {
  if (mouseIsPressed) {
    fill(255, 0, 0, 40);
    ellipse(mouseX, mouseY, 30, 30);
  }
}
function keyPressed() {
 if (keyCode == 80) {
    saveCanvas(c, 'canvasName' + count, 'jpg');
    count++;
  }
}
```

# Mouse Interaction

## Button Class for GUI

```
var brush = 0;
var bx = 20;
var by = 20;
var bw = 50;
var bh = 20;
var by2 = by + bh + 10;
var b1, b2;
```

# Mouse Interaction

## Button Class for GUI

```
function Interface(x, y, w, h, c1, c2, b, bSize) {
  this.button = function() {
    if((mouseX > x) && (mouseX < x + w) && (mouseY > y) && (mouseY < y + h) && mouseIsPressed) {
      fill(c1);
      brush = b;
    } else {
      fill(c2);
    }
    rect(x, y, w, h);
  }
  this.brush = function() {
    if(((mouseX < x) || (mouseX > x + w) || (mouseY < y) || (mouseY > y + h)) && brush == b && mouseIsPressed) {
      if (b == 0) fill(255, 0, 0, 50);
      if (b == 1) fill(255, 50);
      ellipse(mouseX, mouseY, bSize, bSize);
    }
  }
}
```

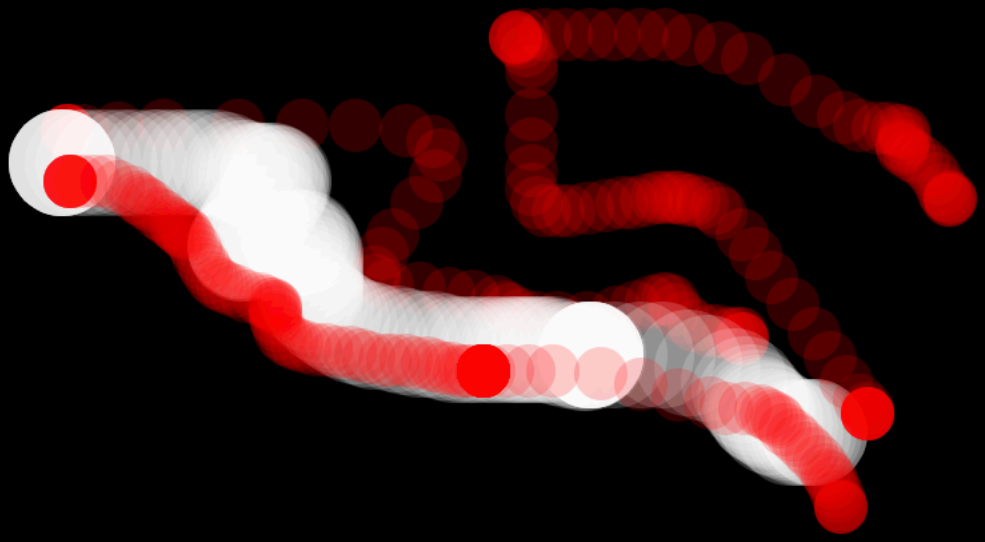
# Mouse Interaction

## Button Class for GUI

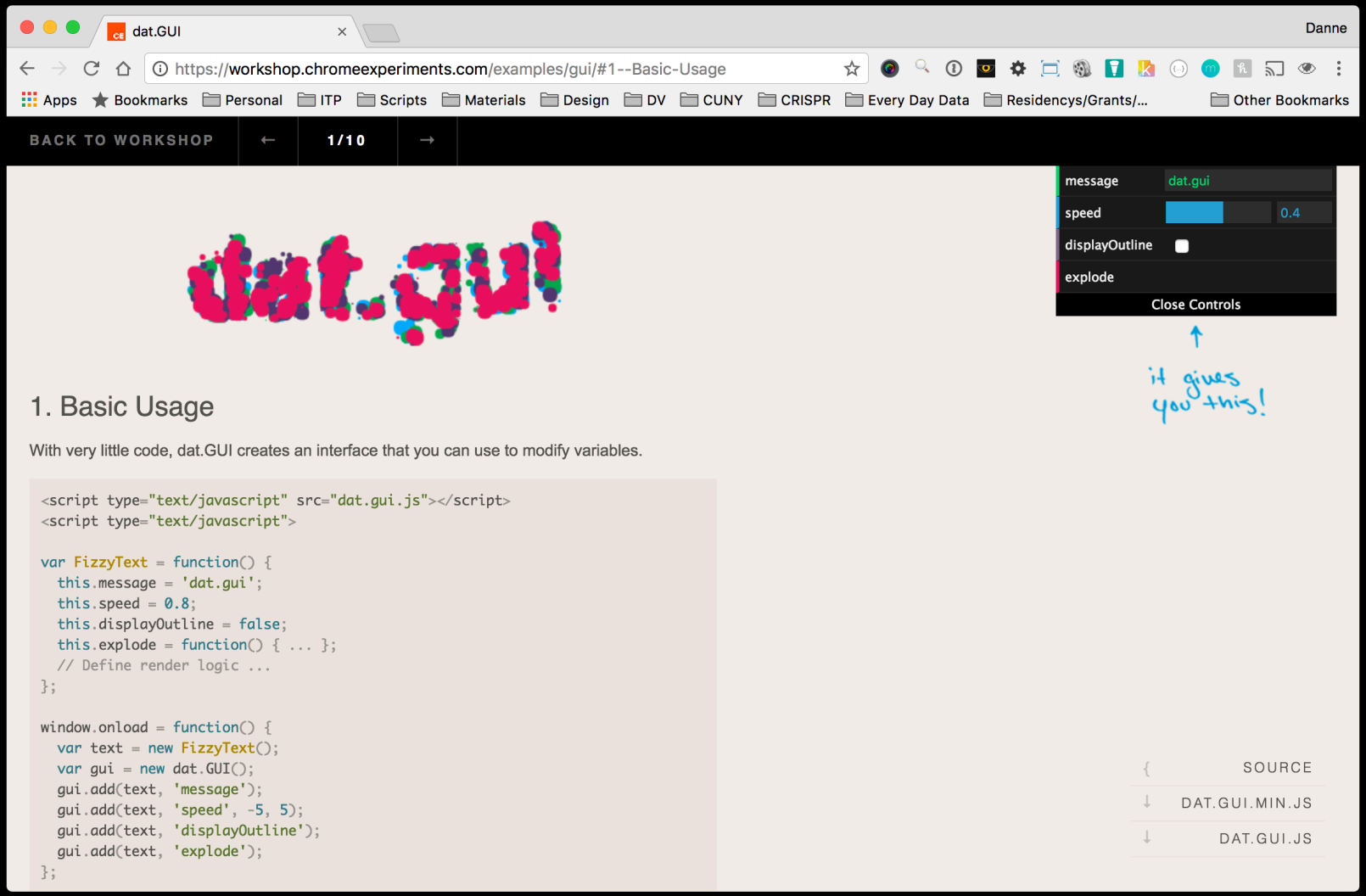
```
function setup() {
  createCanvas(600, 400);
  background(0);
  noStroke();
  // Create a new interface class for each button and brush.
  b1 = new Interface(bx, by, bw, bh, color(100, 0, 0), color(255, 0, 0), 0, 20);
  b2 = new Interface(bx, by2, bw, bh, color(255), color(100), 1, 40);
}

function draw() {
  // Draw the button and brush to the canvas.
  b1.button();
  b1.brush();
  b2.button();
  b2.brush();
}
```

# Mouse Interaction

Button Class for GUI



# dat.GUI Library



# Homework

1. Suggested Reading: pages 205 – 213 and 237 – 243 in Processing, by Casey Reas and Ben Fry
2. Create your own custom design software. Have at least two buttons to switch between two features or use dat.GUI to change brush style. Different brush styles, colors, movements, etc. Draw and export an example sketch to show the class next week.
3. Put your p5js sketch in the Dropbox folder before our next class and be prepared to talk about it.

**Creative Coding**

Professor Danne Woo

dwoo@qc.cuny.edu

creativecode.dannewoo.com