

CMPS 5J – Lab 2

Winter 2018

Due: Sunday January 21, 2018 @ 11:59pm

Overview

Attached is the PDF of Exercise 3-7 from Learning Processing edition 2 (with some extra pages for reference). Your lab assignment will be to complete this exercise. Repeated from the text:

Exercise 3-7: Update Exercise 3-4 on page 40 so that the faster the user moves the mouse, the wider the drawn line. Hint: look up `strokeWeight()` in the Processing reference (https://processing.org/reference/strokeWeight_.html).

What to turn in

Submit a .pde file called **lab2.pde** to Canvas. Your program should be drawing a zoog that follows the mouse around such that the faster the user moves the mouse, the wider the stroke.



Exercise 3-4: Complete Zoog so that the rest of its body moves with the mouse.

```
// Draw Zoog's eyes
fill(0);

ellipse(_____,_____, 16, 32);

ellipse(_____,_____, 16, 32);

// Draw Zoog's legs
stroke(0);

line(_____,_____,_____,_____);

line(_____,_____,_____,_____);
```



Exercise 3-5: Recode your design so that shapes respond to the mouse (by varying color and location).

In addition to `mouseX` and `mouseY`, you can also use `pmouseX` and `pmouseY`. These two keywords stand for the *previous* `mouseX` and `mouseY` locations, that is, where the mouse was the last time the sketch cycled through `draw()`. This allows for some interesting interaction possibilities. For example, let's consider what happens if you draw a line from the previous mouse location to the current mouse location, as illustrated in the diagram in Figure 3-6.

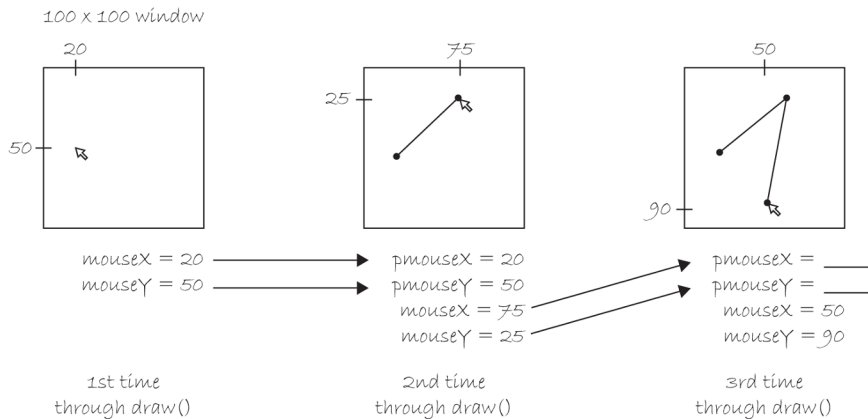


Figure 3-6



Exercise 3-6: Fill in the blank in Figure 3-6.

By connecting the previous mouse location to the current mouse location with a line each time through `draw()`, I am able to render a continuous line that follows the mouse. See Figure 3-7.

Example 3-4. Drawing a continuous line

```
void setup() {
  size(200, 200);
  background(255);
}

void draw() {
  stroke(0);
  line(pmouseX, pmouseY, mouseX, mouseY);
}
```

Draw a line from previous mouse location to current mouse location.

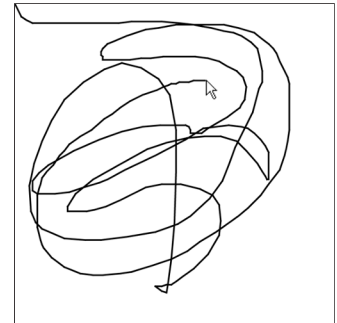


Figure 3-7

Exercise 3-7: Update Exercise 3-4 on page 40 so that the faster the user moves the mouse, the wider the drawn line. Hint: look up `strokeWeight()` in the Processing reference (https://processing.org/reference/strokeWeight_.html).



The formula for calculating the speed of the mouse's horizontal motion is the absolute value of the difference between `mouseX` and `pmouseX`. The absolute value of a number is defined as that number without its sign:

- The absolute value of -2 is 2.
- The absolute value of 2 is 2.

In Processing, you can get the absolute value of the number by placing it inside the `abs()` function, that is `abs(-5)` equals 5. The speed at which the mouse is moving is therefore:

```
float mouseSpeed = abs(mouseX - pmouseX);
```

Fill in the blank below and then try it out in Processing!

```
stroke(0);

_____ (_____);
line(pmouseX, pmouseY, mouseX, mouseY);
```

