



## Lab Exercise 5: First Encounter with Events

### Goal

The objective is to get more practice controlling action on the screen. The programming in this exercise is easy. The goal is follow the logic of the computation.

### Base Program

Write a Processing program that has an 800 x 100 canvas, and moves a red ball from left to right across the screen. These are the requirements:

- 1) The ball is 80 pixels in diameter, and it is “centered vertically on the canvas.”
- 2) The ball’s horizontal position is an integer variable called `xPos`.
- 3) The ball’s initial position is 100 pixels from the left.
- 4) The ball moves one pixel each time the screen is redrawn.

The result looks like this, but longer:



Write this code and check that the four requirements are all fulfilled.

### Change in Parameters

Declare four more integer variables in your program:

- 1) An integer variable called `incDec` that is initialized to 1; this variable is an abbreviation for “increment / decrement.” *Increment* means to add to something, *decrement* means to subtract from something.
- 2) Three integer variables: `rVal`, initialized to 255, `gVal` initialized to 0, and `bVal`, initialized to 0; these variables will be used for RGB values in `fill()`.

Verify that all four variables are declared and initialized.

Now, change your program so the amount the ball moves is edited from its present 1 to `incDec`.

Also, change the RGB values in your `fill()` command are replaced by the three variables `rVal`, `gVal` and `bVal`.

Check to see that your solution works just as it did in the last part.

### ***keyPressed( )***

Among the functions that Processing “looks” for is `keyPressed( )`. This is a function that you write (like `setup( )` and `draw( )`) that works as follows. Whenever a key is pressed on the keyboard, the function runs. Here is an example of a `keyPressed( )` function:

```
void keyPressed ( ) {  
    incDec = - incDec; //flip incrementing/decrementing  
}
```

Add this code to our Processing program, and try it! What you will see is that each time you press a key, the direction of motion of the ball will reverse. Why is that? Because when the `keyPressed( )` function runs, it causes `incDec` to change sign: When it’s positive (incrementing) and moving right, it changes to negative (decrementing) and left moving on a key press; when it’s negative, it becomes positive. Slick, right?

### ***mousePressed( )***

Analogous to the `keyPressed( )` function is the `mousePressed( )` function – a function that will run when the mouse is pressed. For example,

```
void mousePressed( ) {  
    int temp;  
    temp = bVal; //save bVal value  
    bVal = gVal; //make bVal be gVal  
    gVal = rVal; //make gVal be rVal  
    rVal = temp; //make rVal be original bVal  
}
```

Add this code to our Processing program, and try it! What you will see is that with each click of the mouse the color changes red → green → blue → red → green → blue → ... . Notice what is happening: The “255” value is being moved from its original position in `rVal`, to a new position in `gVal`, and `rVal` is re-assigned what was in `bVal`, namely 0.

After playing with your code some, change the `gVal` variable to have 255 as its initial value, and try it again.

### ***Spreadsheet Explanation***

Open a spreadsheet (Excel) and set it up to have the form shown below. (If you are not familiar with spreadsheets, ask your TA for help.) The heading row should have a column for the mouse click count, and then columns for the variables `temp`, `rVal`, `gVal`, `bVal`. Use the “red” initialization shown. Fill in the numbers for the first 10 clicks as they exist after the last assignment statement (`rVal = temp;`)

Finally, explain in your own words (a short paragraph) as to what is happening with these assignment statements of the `mousePressed()` function. I want to know the concepts involved here.

### Tracing the RGB Values of a Program

Mouse Click	temp	rVal	gVal	bVal
initial	?	255	0	0
#1				
#2				
#3				
#4				
#5				
#6				
#7				
#8				
#9				
#10				
#11				
#12				
#13				
		My Explanation of What's Happening ...		

### Wrap Up

You have seen two functions (`keyPressed()` and `mousePressed()`) which are run whenever a key is pressed or the mouse is pressed, respectively. These occurrences are called *events*, and they interrupt the Processing Engine whenever they happen. The Engine, wanting to respond to the events, calls the functions, which gives us a chance to say what to do when the event happens. For key pressing, we changed the direction of the motion; for mouse pressing, we changed the color, in rotation. BTW, we can know what key was pressed, too, but that will come later.

### Turn In

Save the spreadsheet and turn it in to the class dropbox.