Miscellaneous Stuff That Might Be Important.

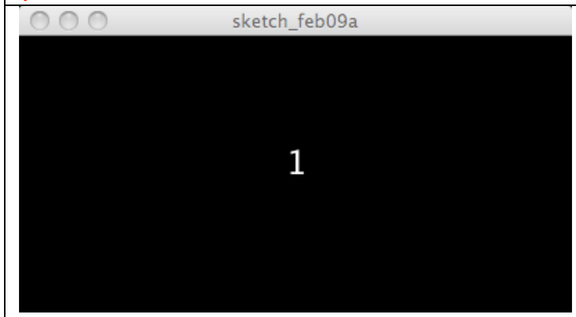
Variable mousePressed VS function mousePressed( )

For much of the work in this class, it is usually "safer" to use the function form of mousePressed( ) instead of the variable form. Here are two sets of code that show why. Both sets seek to count the number of individual key presses made by the user. The second row of the table shows the output from a single key press by the user.

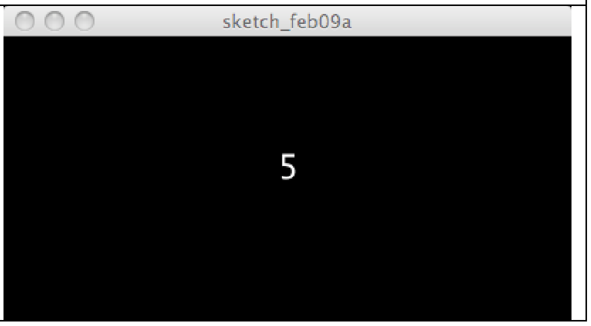
```
int count;
void setup( )
{
  size(400, 200 );
  count = 0;
  textSize( 24 );
  textAlign( CENTER );
}
void draw( )
{
   background( 0 );
   text( count, width/2,
height/2);
}

void keyPressed( )
{
   count++;
}
```

```
int count;
void setup( )
{
  size(400, 200 );
  count = 0;
  textSize( 24 );
  textAlign( CENTER );
}
void draw( )
{
   background( 0 );
   text( count, width/2, height/2);
   if ( keyPressed )
   {
      count++;
   }
}
```



```
1
```



```
5
```

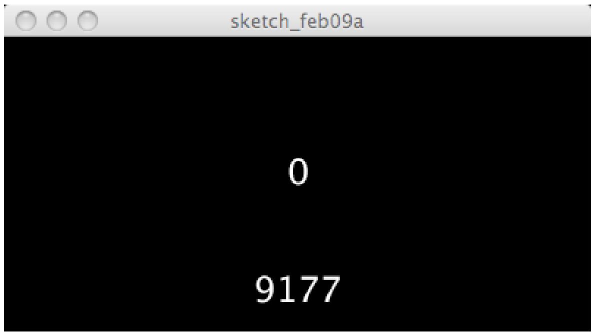
Remember, this is a single key press. The function form counts only one key press. The variable form counts five key presses. The reason the variable counts five is that the variable is checked every frame[1] and the user held the key down for five frames. The function checks the first press of the key and then waits for some small amount of time (the key-repeat time the user sets in the System Preferences) before starting to repeat the count++.

[1] The word frame means a single execution of the draw( ) function. Five frames is five executions of the draw( ) function.

Similar results occur for the mousePressed variable and the mousePressed( ) function. For this reason, we strongly advise you to use the function forms and not the variable forms. This is not to say that you cannot use the variables or that there will never be times you should use the variables but you need to think about the purpose of the key and mouse presses and decide which form is best to use instead of just consistently using one over the other.

Timing Stuff

Processing has a number of time functions. Check the API. One of those functions that is helpful right now is the function, millis( ) which returns as an int value the number of milliseconds since the program began to run. This graphics screen:



```
0

9177
```

was generated by the code used above with a single new line added to draw

```
void draw( )
{
   background( 0 );
   text( count, width/2, height/2);
   text( millis( ), width/2, height*.9);
}
```

The image shows that the screen shot was taken nine thousand one hundred and seventy seven milliseconds after the program began to run and the user has never pressed a key.

This time value can be converted to seconds by dividing the value returned by the funtion by 1000.

```
void draw( )
{
   background( 0 );
   text( count, width/2, height/2);
   text( millis( )1000, width/2, height*.9);
}
```

Detecting Collisions

We can compute the distance between two points using one of several mathematic techniques or we can use a Processing function:

```
   float distance = dist( x1, y1, x2, y2);
```

The function dist( ) returns a float value.

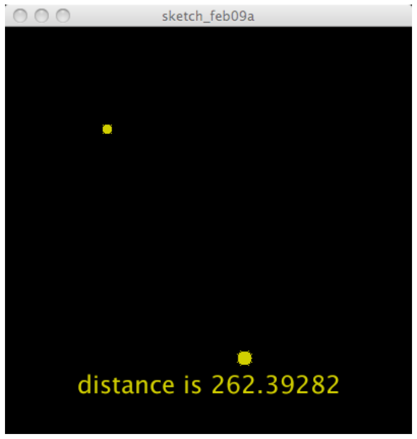
Since we draw figures based on an (x, y) coordinate pair, we can use two (x, y) locations to compute the distance between two figures or a figure and the mouse.

This code

```
float x1, y1, x2, y2, d;
void setup( )
{
  size(400, 400 );

  textSize( 24 );
  textAlign( CENTER );
  x1 = 100;
  y1 = 100;
  x2 = 235;
  y2 = 325;
  fill( 200, 200, 0 );
}
void draw( )
{
   background( 0 );
   ellipse( x1, y1, 10, 10 );
   ellipse( x2, y2, 15, 15 );
   d = dist( x1, y1, x2, y2);
   text("distance is " + d, width/2, height*.9);
}
```

generates this output:



```
distance is 262.39282
```

Formatting the Output

Do we really need five decimal places? NO… Processing has several formatting functions. These take a value and return a string of characters that contain a formatted number.

(First- a word from our sponsor - String…)
A string of characters is referred to as a String and is visually shown as characters between quotation marks (" ") or double quotes.

String is a type of data similar to PImage and PFont. The jargon of programming calls String a class and String variables are called objects. Variables of the types float, int, char, and Boolean can have only one value. Variables (or objects) of a class such as String or PImage actually have many values stored in them. We can declare a String variable just like we declare a float variable:

```
String s;
```

String objects can be assigned values just as we assign values to a float variable:

```
s = "Hello World";
```

We can do both tasks in one line of code:

```
String s = "Hello World";
```

We can also assign a String variable a value that is returned by a function as long as the returned value is a String. The code below uses this feature of String variables.
(Now - back to our regularly scheduled blathering…)

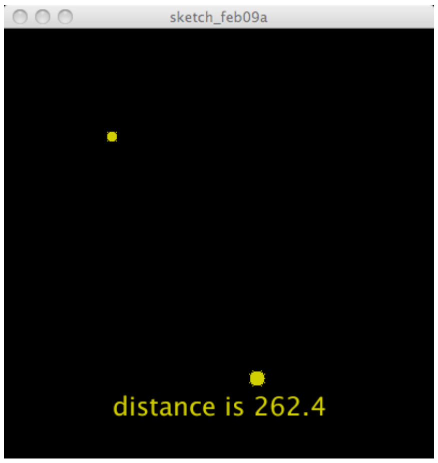
Here is the same code with formatted output:

```
float x1, y1, x2, y2, d;
void setup( )
{
  size(400, 400 );

  textSize( 24 );
  textAlign( CENTER );
  x1 = 100;
  y1 = 100;
  x2 = 235;
  y2 = 325;
  fill( 200, 200, 0 );
}
void draw( )
{
   background( 0 );
   ellipse( x1, y1, 10, 10 );
   ellipse( x2, y2, 15, 15 );
   d = dist( x1, y1, x2, y2);
   String sd = nf(d, 1, 1);
   text("distance is " + sd, width/2, height*.9);
```

}

Here is the output:



distance is 262.4

Check these in the API, tinker with them, and use these as you need them.