

The Interactive Fringe

Programming in Processing

with Tim Armato, John Keston and Nick Richter
(Thursday 2/1/07 6:00 - 8:00 Room 229)

I. What is "Processing"?

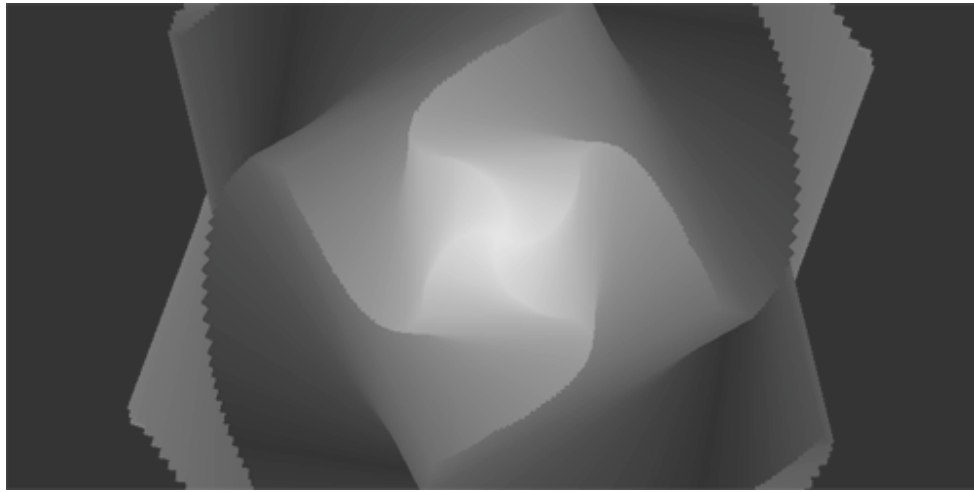
1. Processing is an Open Source programming language and environment aimed at artists working in contexts such as images, animation, sound and interactivity.
2. What does it do?
3. How does it work?
4. Who developed it and why?

II. Examples and Resources

1. <http://www.processing.org> and <http://processing.org/exhibition/>
2. Jeremy Thorp: <http://www.blprnt.com/portfolio/>
3. Jared Tarbell: <http://www.levitated.net/> and <http://www.complexification.net/>
4. More examples: Tim, John, Nick

III. An Overview of the Mechanics of Processing

1. Parts of a program
2. Anatomy of a Processing program
3. Make your own Processing application



Parts of a program

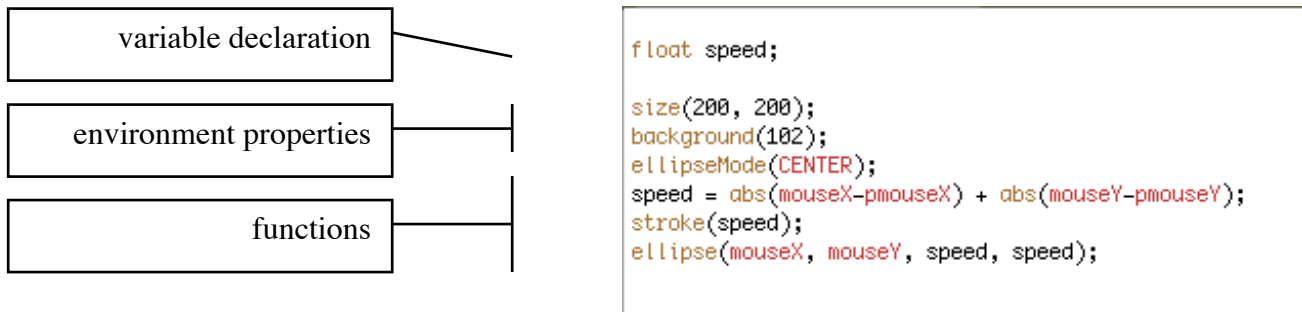
Variables are containers for a changing value. Must define the *data type*:

- *int* = integer, whole number
- *float* = floating point number, decimal
- *char* = a text character or symbol. Essentially, a single letter.
- *boolean* = a binary value equivalent to "true" or "false"
- *color* = a color
- *string* = a sequence of *chars* such as a word or sentence
- *array* = a list of data. Can be any data type.

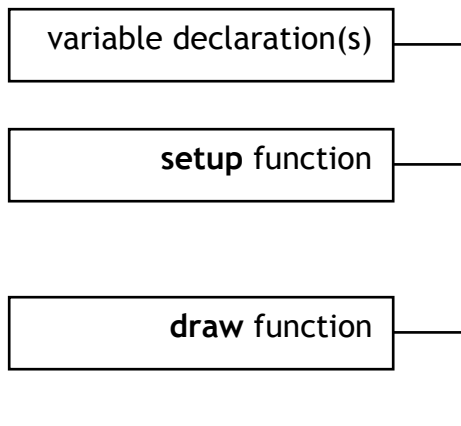
Functions (aka Commands) are the *verbs* of a programming language. Each function perform specific tasks such as drawing a circle. User-defined functions are groups of pre-defined functions that are executed in a specific order to complete a more complex task.

Parameters (aka Arguments) are pieces of information sent to functions. Most functions need to be given some information to complete their tasks, such as the diameter and location of the circle.

Anatomy of a Processing program



More commonly, a Processing program is divided into **setup** and **draw** functions. The statements within **setup** are executed only once, when the program is first run. Then, the statements within **draw** are executed in a perpetual loop until the program is manually stopped. Notice that the variable *speed* (of type *float*) is defined before **setup**. This makes it a *global* value that can be used anywhere in the program. Variables declared within **setup** are not accessible from other functions, such as **draw**.



```
// Adaptation of:  
//  
// Patterns  
// by REAS <http://www.groupc.net>  
// Move the cursor over the image to draw with a software tool  
// which responds to the speed of the mouse.  
// Created 19 January 2003  
  
float speed;  
  
void setup()  
{  
  size(200, 200);  
  background(102);  
  ellipseMode(CENTER);  
}  
  
void draw()  
{  
  speed = abs(mouseX-pmouseX) + abs(mouseY-pmouseY);  
  stroke(speed);  
  ellipse(mouseX, mouseY, speed, speed);  
}
```