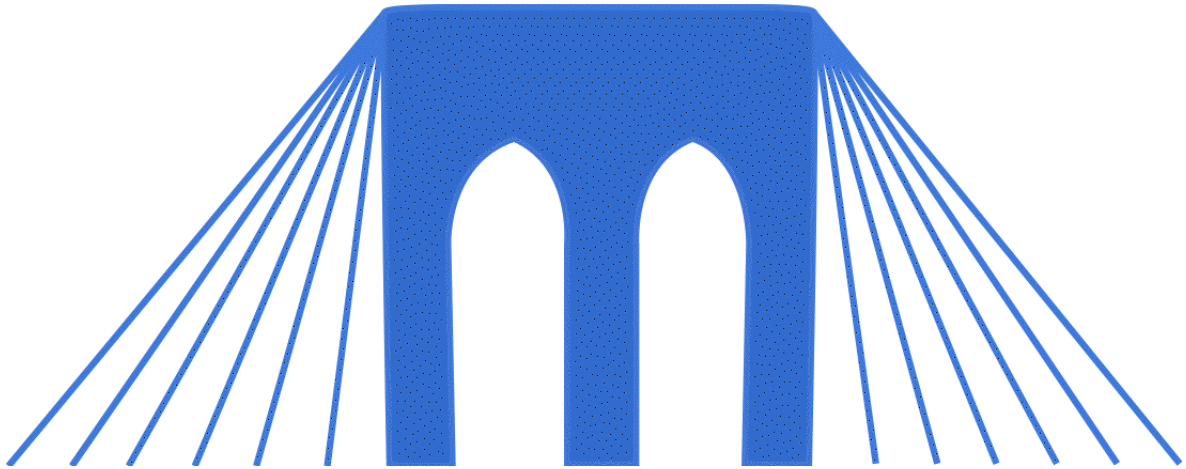
Bridges To Computing

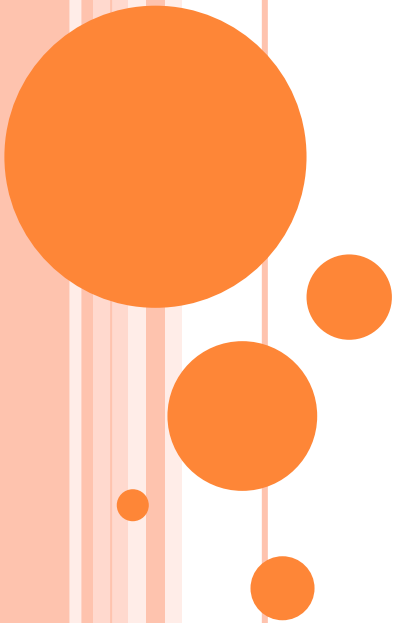


General Information:

- This document was created for use in the "Bridges to Computing" project of Brooklyn College.
- This work is licensed under the Creative Commons Attribution-ShareAlike 3.0 License. You are invited and encouraged to use this presentation to promote computer science education in the U.S. and around the world.
- For more information about the Bridges Program, please visit our website at: http://bridges.brooklyn.cuny.edu/

Disclaimers:

- **IMAGES**: All images in this presentation were created by our Bridges to Computing staff or were found online through open access media sites and are used under the Creative Commons Attribution-Share Alike 3.0 License. If you believe an image in this presentation is in fact copyrighted material, never intended for creative commons use, please contact us at http://bridges.brooklyn.cuny.edu/ so that we can remove it from this presentation.
- **LINKS**: This document may include links to sites and documents outside of the "Bridges to Computing" domain. The Bridges Program cannot be held responsible for the content of 3rd party sources and sites.



Graphics & Interactive Programming

Lecture 1

Introduction to Processing

Resources

- Processing web site:
  - http://www.processing.org/
- Download Processing
  - http://processing.org/download/
- Reference:
  - http://www.processing.org/reference/index.html

Content

1. Graphics Programming
2. Interactive Programming
3. Hexadecimal Color
4. Processing (Introduction)
    1. Basics
    2. Functions – Custom
    3. Functions – Built in, drawing
    4. Variables
    5. Selection
    6. Repetition
5. Don’t Panic

Graphics Programming:

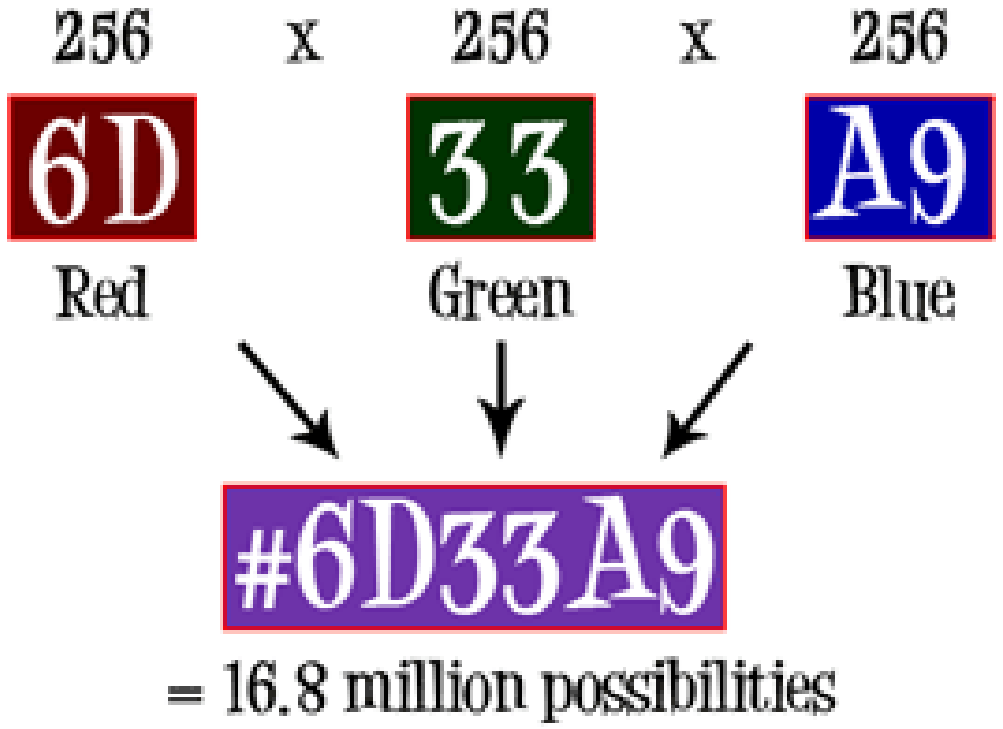
- Utilizing and/or manipulating images within a computer program.
  - Photoshop
  - Game Programming
  - Digital Movie Creation
- Windows and GUI are graphics programming.
- Many programming languages have "graphics libraries" (Direct3D, OpenGL) but the robustness of these libraries can make them difficult to work with.
- Processing has its own special library of functions that we can use to create 2D and 3D images.

Interactive Programming:

- A movie or television show contains visual images, but you can't change or modify those images.
- A GAME or INTERACTIVE STORY by contrast does allow the user to change what is shown.
- Websites are another area where we want INTERACTIVITY as part of our final design
  - links
  - search boxes
  - image maps
- Interactive Programming is programming that uses special functions to users to modify the behavior of a program.
  - We will officially look at event handlers in the next lecture.
  - But look for the special keyboard handler functions in this lecture.

Note on Hexadecimal Color

- Hexadecimal color is a common color mode used in computer graphics.
- It's a very efficient method, using only six digits to identify a single color out of the 16.8 million available on modern computer monitors.



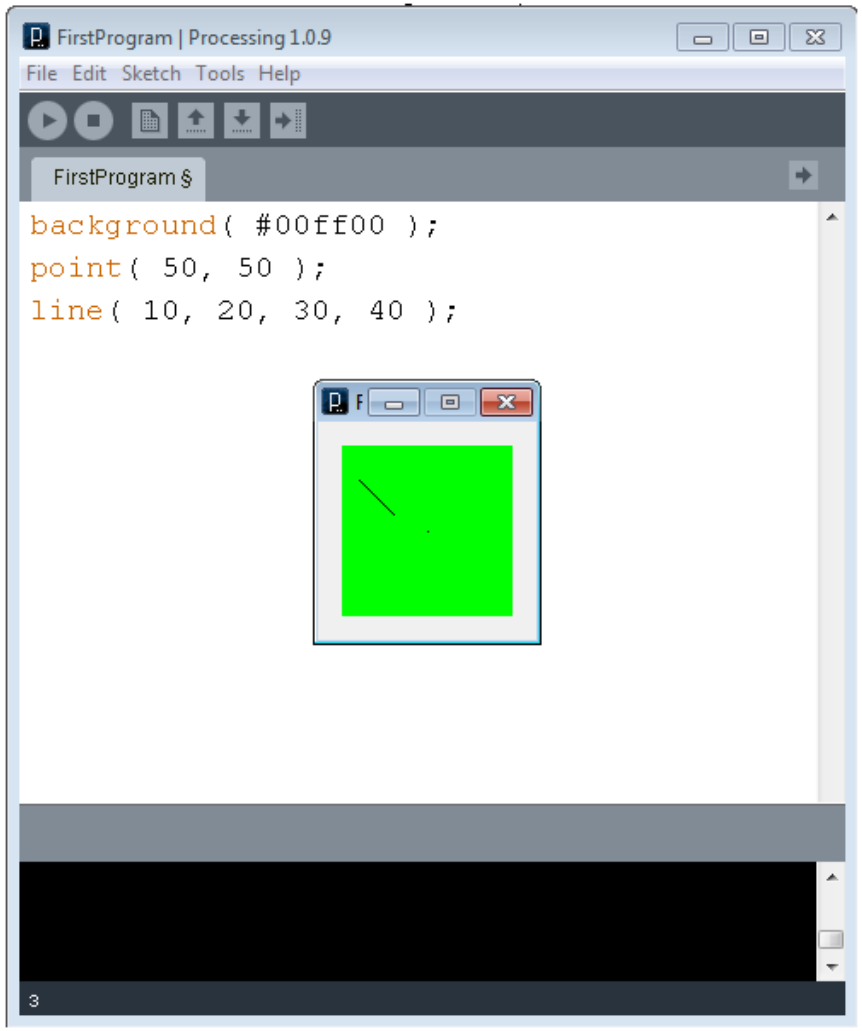
Processing

- Why we are going to use it:
  - It's FREE (www.processing.org)
  - Simple (Imperative, Procedural, OO) scripting language.
  - Has excellent built in graphics capabilities.
  - Programs can also be saved as applets and run inside a browser (client side scripting).
  - Powerful library suite.

- Essentials:
  - Originally written for artists (create narratives, games).
  - Programs in Processing are called sketches.
  - Text-based programming language.
  - Write and run using an integrated development environment (IDE) that is part of Processing.

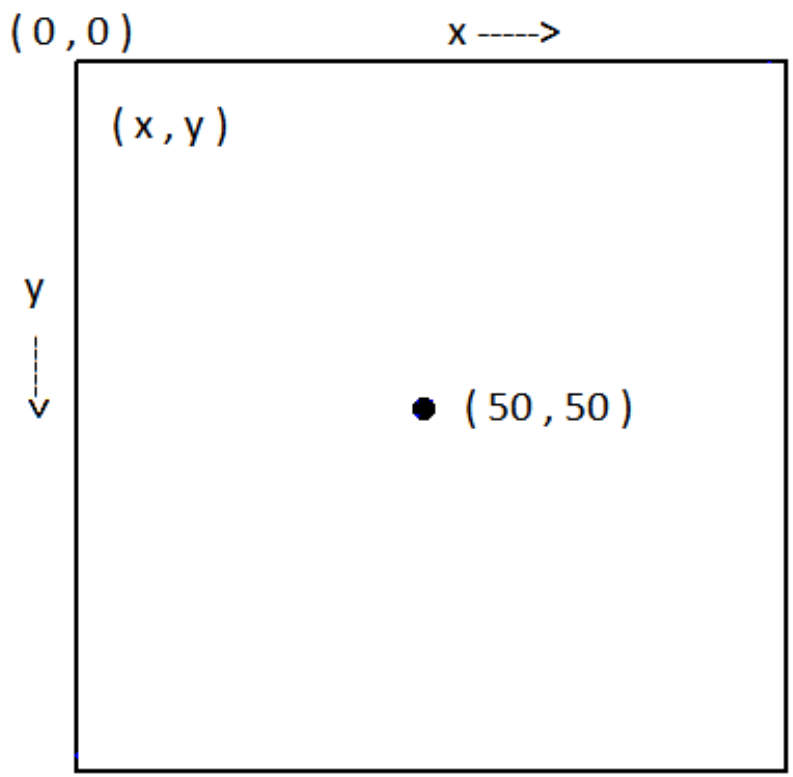
Processing – 1st Program

- Open processing and type in the following commands just as you see them in the window.
- Then hit the "run" button in the top bar.
- The run button is the first of the 6 buttons in the command bar.



Coordinate System

- All graphics are drawn using the following coordinate system:
  - Think of it like a piece of graph paper
  - A point fills in one cell on the graph paper
  - A line fills in multiple cells, from one end-point of the line to the other



Processing Programming Basics

- Programs:
  - Programs are written by typing in text.
  - Programs are processed top-down, left-right (sequence).
  - Special terms and functions exist in the language (semantics).
- Syntax:
  - Each line contains a statement
  - Statements end with a semi-colon (;)
  - Comments are contained within /** and */
  - Another way to make comments: //
  - Blocks or related code (part of a function) are specified using curly braces { }
- Special Functions:
  - Two special functions are part of the syntax and semantics of the language and may only appear ONCE in a program:
  - setup()
  - draw()

Processing Functions

- Functions:
  - Are blocks of code that have been given a name..
  - Are called when needed, by using their name.
  - Are an aspect of the procedural paradigm.
  - Provide a way to re-use code.

- Some functions we will create ourselves:

  Example:

```
void my_function() {
    line( 10, 20, 30, 40 );
    point(120, 50)
}
```

  - void keyword indicates function returns nothing
  - my_function() = the name of the function
  - curly brackets ({ and }) delineate beginning and end of the function

Draw Functions (built-in, Semantics)

- Other Functions are built into the Processing language.
- Processing has built in functions for creating shapes:
  - point(x, y)
    - draws one point (looks like a dot...)
  - line(x1, y1, x2, y2)
    - connects two points
  - triangle(x1, y1, x2, y2, x3, y3)
    - connects three points
  - quad(x1, y1, x2, y2, x3, y3, x4, y4)
    - connects four points
  - rect(x, y, width, height)
    - origin + extent; square if width=height
  - ellipse(x, y, width, height)
    - origin: center of ellipse's; circle if width=height
  - arc(x, y, width, height, start, stop)
    - origin: center of arc's bounding box (see ellipse)
    - start and stop: can be whole (int) or real (float) numbers (float); expressed in degrees or radians, depending on current angle mode; 0 is due east; measured clockwise.

VARIABLES

- A variable is a name and value pair.
- Variables provide a way to save information so that you can refer to it (use it, change it) later.
- You may be familiar with variables from studying algebra.
  - Example: 20 = x / 5
  - What is x? A name for a number whose value is 100
- We can create our own variables in Processing.
  - Example: int x = 3;
    - int is the data type and means x is a whole number
    - x is the name of the variable
    - 3 is the value of x
- Other variables are built into the processing language (part of the semantics of the language) and are controlled/maintained by the OS.
  - Examples: width, height
    - Always indicate the width and height of the screen.

Keyboard Interaction Functions and Variables (Built – In)

- Functions
  - keyPressed()
    - handles behavior when user presses a key down
  - keyReleased()
    - handles behavior when user releases a key
  - keyTyped()
    - handles behavior when user types a key (press and release)
- Variables
  - key
    - indicates which key was pressed/released/typed
    - equals CODED when special key is pressed/released/typed, like an arrow key, shift, control, alt, etc.
  - keyCode
    - indicates special key: UP, DOWN, LEFT, RIGHT, ALT, CONTROL, SHIFT

2ND PROGRAM

```
// This is a comment

/* So is this */

/**
      So is this
*/


void setup() {                          // Special function, only done once!
       background( #ffffff );           // Changes background color to white.
    }


void draw() {                           // Special function, repeats over and over!
       line( 10, 20, 30, 40 );          // Draws a line.
       point( 50, 50 );                 // Draws a point.
    }


void keyPressed() {                     // Special function, called by OS.
       background( #0000ff );           // Changes background to blue.
    }
```

Selection (Imperative Paradigm)

- We will want the ability to make a choice in our programs.

1. Some things we will only want to happen IF something is true.
2. In other cases we will want to choose from more than one option (if – else)
3. At other times we will want to make multiple test at once:
   - || is or
   - && is and

```
if ( test ) {
        statements
}
```

```
if ( test ) {
        statements
} else {
        statements
}
```

```
if ( ( test1 ) || ( test2 ) ) {
        statements}
```

Repetition (Imperative Paradigm)

- You may want to repeat an action. Both for and while loops are supported:

```
while (expression) {
    statements
}
```

- Example:

```
int i=0;
while(i<80) {
    line(30, i, 80, i);
    i = i + 5;
}
```

3RD PROGRAM

```
void setup() {                          // This special function is called once at the start
        background( #ffffff );
}

void draw() {                           // This special function loops over and over
    line( 10, 20, 30, 40 );
    point( 50, 50 );
}

void keyPressed() {                     // This special function is called by the OS
    if ( key == 'R' || key == 'r' ) {
            background( #ff0000 );
    } else {
             int i=0;
            while(i<80) {
                        line(30, i, 80, i);
                         i = i + 5;
            }
    }
}   //  Notice how we indent the content of loops and functions to help us read the code.
```

Don't Panic

- We will have several labs on using processing.
- For those of you with little or no programming experience your final project will not be that hard.
- Anyone can learn to do simple (and fun) things with processing fairly quickly.
- NOTE: Processing is FREE software that you can download and run on your machine at home:
  - http://processing.org/download/