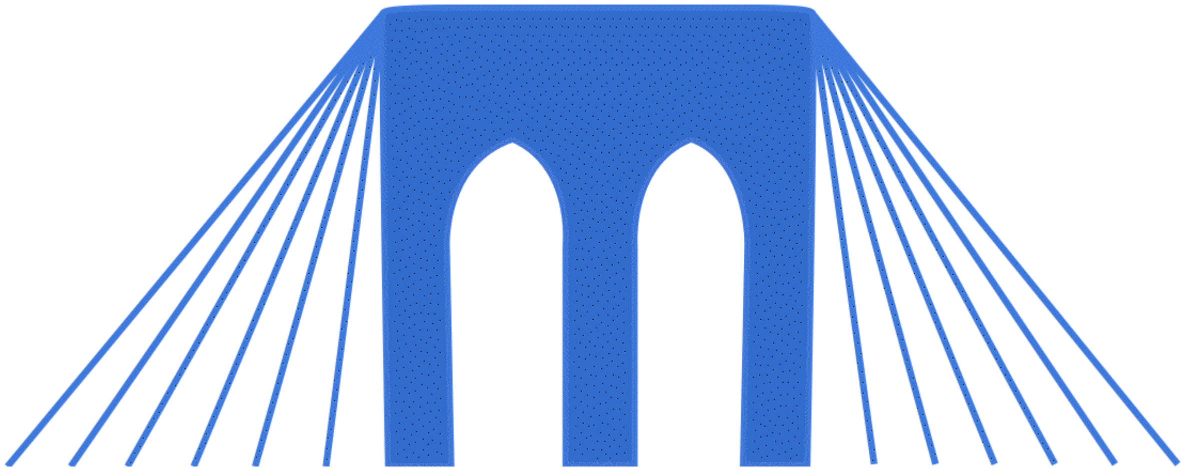
Bridges To Computing



Graphics & Interactive Programming

Lecture 2

More Programming with Processing

Resources

- Processing web site:
  - http://www.processing.org/
- For loops (counter-controlled repetition)
  - http://processing.org/reference/for.html
- While loops (event-controlled repetition)
  - http://processing.org/reference/while.html
- Reference:
  - http://www.processing.org/reference/index.html

Content

1. Variables
2. Bitmap and Vector Graphics
3. Event Handlers
4. Keyboard & Mouse Events
5. Repetition (looping)
    1. Counter Controlled (for)
    2. Event-Controlled (while)
6. Standard Processing Program

Previously in Lecture 1

- Processing is a <u>scripting language</u> that allows you to write simple graphics programs using an IDE.
- Processing is an <u>Imperative</u>, <u>Procedural</u>, <u>Object-Oriented</u> programming language, that uses <u>syntax</u> rules and a <u>semantic</u> keyword set similar to C++.
- In the last lab you started using the Imperative and Procedural paradigm aspects of Processing. How does Processing implement:
    1) Sequence
    2) Selection
    3) Repetition (will cover in more detail)
    4) Functions

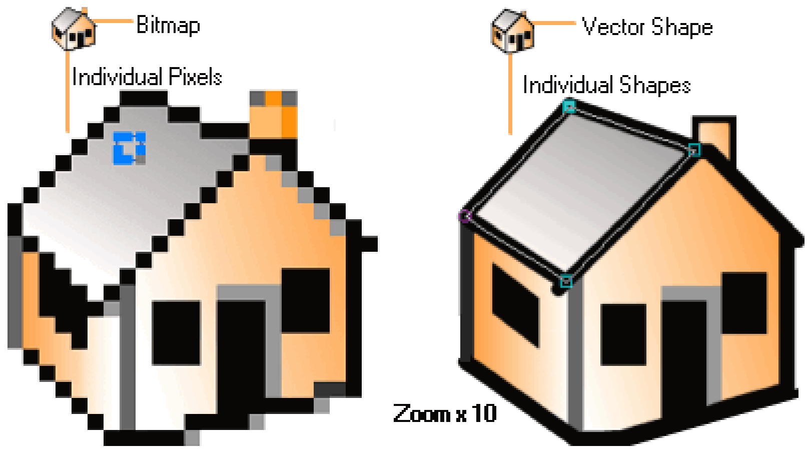
Variables

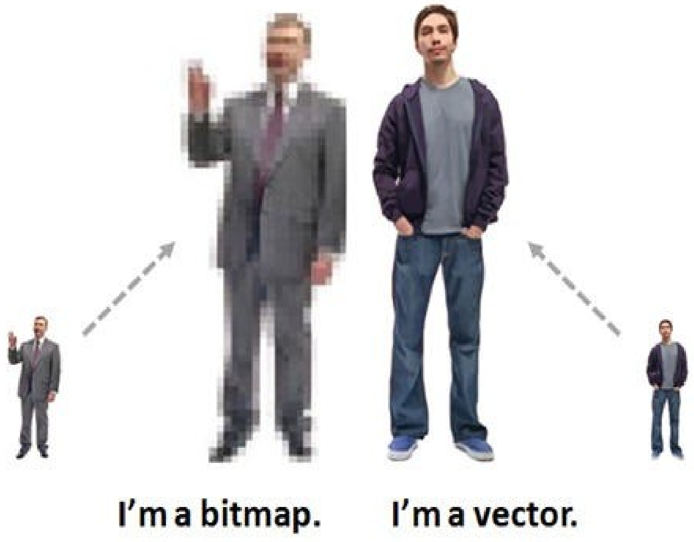
- A variable is a <u>name and value pair</u>.
- Variables provide a way to save information so that you can refer to it (use it, change it) later.
- We will use variables for control (position, color, etc.)
- Valid data types for variables are:
  - int — for storing integers (whole numbers)
  - float — for storing floating point (real) numbers
  - boolean — for storing true or false values
  - char — for storing single characters
  - String — for storing multiple (strings of) characters
- Examples:
  - int x1 = 10;
  - int y1 = 10;
  - int x2 = 20;
  - int y2 = 20;
  - line(x1, y1, x2, y2);

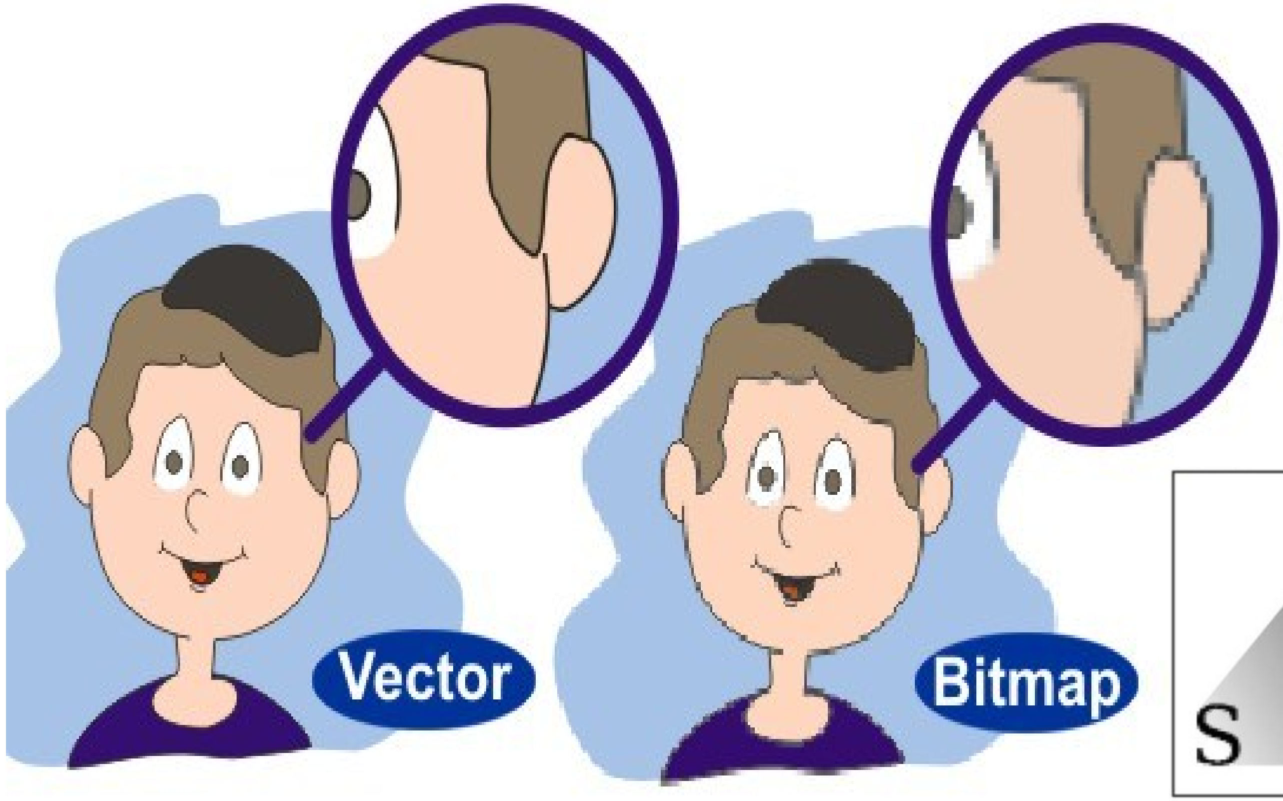
Bitmap & Vector Graphics

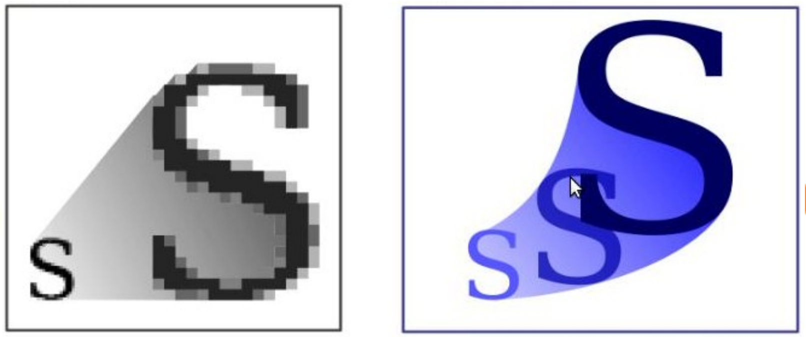
- There are two basic ways to create/store visual images:
  - **Bitmap Graphics:** Images that are composed of individual pixels.
    - Advantages: Easy to create/capture from real life.
    - Disadvantages: Large, don't scale well.
  - **Vector Graphics:** Images we create using a sequence of mathematical function calls.
    - Advantages: Smaller, scale well.
    - Disadvantages: Can be difficult to create.

Bitmap and Vector Images (2)









Event Handlers

- In Processing “event handlers” are special, built-in functions:
  - These functions are called by the OS.
  - But you change the content of the funtions.
- Events are changes in the “state” of the program and/or user initiated actions like keyboard and mouse input.
- Users, can add event handler functions to their programs, modify them, and use them to change program variables, state, program flow, etc..
- You have already used the keyPressed() event handler which responds to keyboard actions.

Keyboard Variables & Event Handler(s)

```
/** ********************
Use event-listeners like keyPressed() to allow
users of your program to cause things to happen.
******************** */

void keyPressed() {
        if ( key == 'A' || key == 'a' ) {
                background( #FF0000 );
        }
}

// Function is called when any key is pressed.
// If key is 'A' or 'a' background is changed to red.
```

Mouse Variables & Event Handlers

- Variables
  - mouseX and mouseY
    - Special variables, managed by the computer.
    - Indicate (x, y) location of mouse pointer
  - mouseButton
    - indicates which button was pressed, on a multi-button mouse (on a Mac, use Cntl-click for left mouse button, Alt-click for middle mouse button and Apple-click for right mouse button)
- Event Handlers
  - mouseClicked()
    - Predefined "event handler" function. You change content.
    - Handles behavior when user clicks mouse button (press and release)
  - mouseMoved()
    - handles behavior when user moves mouse (moves it without pressing button)
  - mouseDragged()
    - handles behavior when user drags mouse (moves it with button pressed)

4th Program

```
void setup() {
    size( 200, 200 );
}
void draw() {
    background( #cccccc );
    fill( #990000 );
    rect( mouseX, mouseY, 20, 20 );
}

void mouseMoved() {
    fill( #000099 );
    rect( mouseX, mouseY, 20, 20 );
}
void mouseDragged() {
    fill( #009900 );
    rect( mouseX, mouseY, 20, 20 );
}
```

5TH PROGRAM

```
void setup() {
    size( 200, 200 );
}
void draw() {
    background( #cccccc );
    rect( mouseX, mouseY, 20, 20 );
}

void mousePressed() {
    if ( mouseButton == LEFT )  {
            fill( #990000 );
    } else if ( mouseButton == CENTER ) {
            fill( #009900 );
    } else if ( mouseButton == RIGHT )    {   // Ctrl-click on mac
            fill( #000099 );
    }
}
```

Repetition

- In the Imperative Paradigm ("smart list") we need sequence, selection and repetition.
- There are three basic types of repetition in Processing:
  - draw() function
  - for loops
  - while loops
- Repetition is necessary for animation, when we will want to display things over and over (although slightly differently each time).

Counter Loops (for)

- Counter controlled loops (<u>for</u> loops) repeat things <u>for</u> a fixed number of times.
- Syntax:

```
for ( init; test; update ) {
        statements
}
```

- Example:

```
int x1 = 10;

for ( int i=0; i<5; i++ ) {
    line( x1, 10, x1, 20 );
    x1 = x1 + 10;
}
```

Event-Controlled Loops (While)

- Event-Controlled (while loops) loops repeat things while a condition holds true
- Syntax:

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

- Example:

```
int x2 = 10;
while ( x2 < width ) {
        line( x2, 30, x2, 40 );
        x2 = x2 + 10;
}
```

6TH PROGRAM

```
int x1 = 10;
int x2 = 10;

void setup() {
   stroke(#FF0000);      // Make line red.
   for(int i=0; i<5; i++ )   {
         line( x1, 10, x1, 20 );
         x1 = x1 + 10;
    }
   stroke(#0000FF);     // Make line blue.
   while ( x2 < width )   {
         line( x2, 30, x2, 40 );
         x2 = x2 + 10;
   }
}
void draw() {
}
```

Standard Processing Program

1) Setup any variables or classes you are going to use.
2) Use setup() function to specify things to do once, when the sketch first opens
3) Use draw() function to specify things to do repeatedly
   - Use frameRate() function to specify how often things should be repeated in draw();
   - Default frame-rate is 60 (60 frames per second)
   - NOTE: call to frameRate() should be done inside setup() function. For animations frameRate should be <u>slower</u> than 30 f/sec/
4) Declare and event-listeners that you are going to use.
5) Declare any custom made functions you are going to use.
6) Declare any classes that you are going to use.

Note: I have created a processing template that you can use to start your programs.

Generalized Program Outline

```
/** **********   Variables: ********** */

/** **********   setup():    **********  */
void setup() {

}
/** **********  draw():      **********  */
void draw() {

}
/** ********** Event Handlers:   **** */
void keyPressed() {

}
/** ********** Custom Functions:  *** */
void  draw_simple_image {

}
/** ********** Classes: ************* */
```

The End