Images, Pixels, ART!!

Return To RGB
- Recall that the screen (and other video displays) use red-green-blue lights, arranged in an array of picture elements, or pixels

Combining Colored Light
- The Amazing Properties of Colored Light!
- Caution: It doesn’t work like pigment

Green + Red = Yellow?
- Colored light seems to violate our grade school rule of green = blue + yellow
  What gives?
- In pigment, the color we see is the reflected color from white light; the other colors are absorbed

White, Gray, Black
- You know that gray is just different degrees of white as the “light is turned down” till we get to black
  \[
  \begin{align*}
  \text{Black} &= [0, 0, 0] \\
  \text{Gray} &= [128, 128, 128] \\
  \text{White} &= [255, 255, 255]
  \end{align*}
  \]
- White-gray-black all have same values for RGB

Colors
- Colors use different combinations of RGB

BLOOD RED is my favorite color
But is it art?

Mondrian, Pollock, Albers are stars ...

IMAGES AND PICTURES

- [http://www.processing.org/learning/pixels/](http://www.processing.org/learning/pixels/)
- This tutorial is really fun.

Representing Images

- gif, jpg, tiff, png, ...
- RGB
- pixel
- compression
- lossy
- lossless

Art In A Click

- Computer art, that is, art generated by computers, not art created by people using computers, leads to some fun Web sites.
- Google “piet mondrian”. He was a cubist and created pictures that look like this: [http://www.google.com/search?client=safari&rls=en&q=Piet+Mondrian&ie=UTF-8&oe=UTF-8](http://www.google.com/search?client=safari&rls=en&q=Piet+Mondrian&ie=UTF-8&oe=UTF-8)
- Could we generate this in processing?

How about something a little easier first?


What would we do in Processing? What does Random do?

```java
void setup() {
    rectMode(RADIUS);
    noStroke();

    // Set random color
    int red = random(255);
    int green = random(255);
    int blue = random(255);
    color = color(red, green, blue);

    // Draw rectangles
    for (int i = 0; i < 30; i++) {
        int x = random(width);
        int y = random(height);
        int width = random(50, 200);
        int height = random(50, 200);
        fill(color); // Draw filled rectangle
        rect(x, y, width, height);
    }
}
```
Random Numbers

- Random numbers should be called random number sequences, because the definition requires that no matter how many numbers you already know in the sequence, it’s not possible to predict the next one. A non-random sequence is 2, 4, 6, 8, 10, ...
- Computers cannot produce random numbers (because computers are completely predictable), but they can produce a sequence of numbers that passes all of the tests for randomness. These are called pseudo-random numbers, but everyone drops the “pseudo” part.
- To generate a random number in Processing we write:
  - random(<smallest possible number>, <largest possible number>).
  - We get back a number – we can’t predict which – between the two limits, including the end points.
- To generate a random number between 0 and 255, write random(0, 255).
- To generate a number between 0 and 1, write random(0, 1).

How about Jackson Pollock?

- Google “jackson pollock”.
- He was an abstract expressionist and created paintings that look like this!
- Circa 1950

Computer generated Jackson Pollock?


POINTILISM

- http://www.processing.org/learning/pixels/

Drawing an image

// Declaring a variable of type PImage
PImage img;

void setup() {
  size(320,240);
  // Make a new instance of a PImage by loading an image file
  img = loadImage("mysunmervacation.jpg");
}

void draw() {
  background(0);
  // Draw the image to the screen at coordinate (0,0)
  image(img,0,0);
}
More basic image

- Add the image to the sketch Sketch/Add File
- image(img, x, y, width, height); // fills the box
- img.width, img.height
- tint(x); // 0-255
- tint(r,g,b);
- loadPixels();
- pixels[i] = someColor;
- updatePixels();

Manipulating individual pixels

void messWithPixels() {
  loadPixels();
  // Loop through every pixel
  for (int i = 0; i < pixels.length; i++) {
    // Set pixel at that location to random color
    pixels[i] = color(red(pixels[i]),
                     green(pixels[i]),
                     blue(pixels[i]) * mouseX/width);
  }
  // When we are finished dealing with pixels
  updatePixels();
}

Transparency, Tints and More!

http://www.processing.org/learning/pixels/