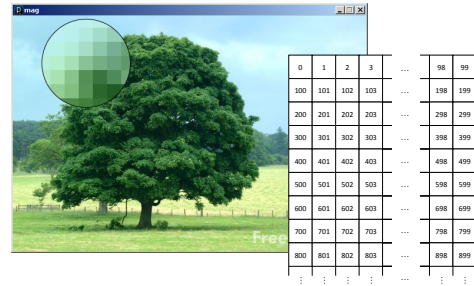


Image Processing

... computing with and about data,
 ... where "data" includes the values and relative locations of the colors that make up an image.

An image is an array of colors

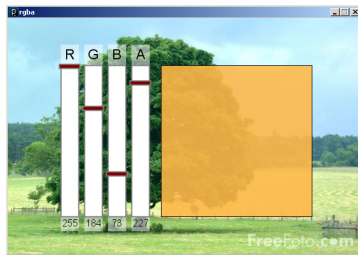


Pixel : Picture Element

mag.pde

Color

- A triple of bytes [0, 255]
 - RGB or HSB
- Transparency (alpha)
 - How to blend a new pixel color with an existing pixel color



rgba.pde

Accessing the pixels of a sketch

- loadPixels()
 - Loads the color data out of the sketch window into a 1D array of colors named pixels[]
 - The pixels[] array can be modified
- updatePixels()
 - Copies the color data from the pixels[] array back to the sketch window

```
// whiteNoise
void setup() {
  size(400, 300);
}

void draw() {
  // Load colors into the pixels array
  loadPixels();

  // Fill pixel array with a random
  // grayscale value
  for (int i=0; i<pixels.length; i++) {
    pixels[i] = color(random(255));
  }

  // Update the sketch with pixel data
  updatePixels();
}
```



Examples

- whiteNoise
- colorNoise
- pixelGradient

Useful color functions

- `red(color)`
 - extract the red component of from color
- `blue(color)`
 - extract the green component from a color
- `green(color)`
 - extract the blue component from a color

tint() / noTint()

- `tint()` modifies the fill value for images

```
tint(gray);
tint(gray, alpha);
tint(red, green, blue);
tint(red, green, blue, alpha);
```

- Turn off applied tint values with `noTint()`

```
void setup() {
  // Load the image three times
  PImage warhol = loadImage("andy-warhol2.jpg");
  size(warhol.width*3, warhol.height);

  // Draw modified images
  tint(255, 0, 0);
  image(warhol, 0, 0);

  tint(0, 255, 0);
  image(warhol, 250, 0);

  tint(0, 0, 255);
  image(warhol, 500, 0);
}
```



Examples

- warholTint
- warholRed
- warhol
- warholArray

Basic Filters

- Color
 - Extracting Red/Green/Blue colors
 - `pixels[i] = color(red(c), 0, 0);`
 - `pixels[i] = color(0, 0, blue(c));`
 - Grayscale
 - `pixels[i] = color(0.3*red(c)+0.59*green(c)+0.11*blue(c));`
 - Negative
 - `pixels[i] = color(255-red(c), 255-green(c), 255-blue(c));`

Sepia- Technique for archiving BW photos

```
float r =
red(c)*0.393+green(c)
*0.769+blue(c)*0.189;
float g =
red(c)*0.349+green(c)
*0.686+blue(c)*0.168;
float b =
red(c)*0.272+green(c)
*0.534+blue(c)*0.131;
pixels[i] = color(r,
g, b);
```



Examples

- blackWhite
- negative
- sepia
- sepiaPalette
- sepiaWithPalette

Notes

- Processing.js
 - Javascript version of Processing
 - how I post sample code for you (it runs in the browser)
- Differences
 - size() can not take variables
 - images used need `/* @pjs preload = "name"; */`
 - comments that are wrapped in `/**`
`*/`
 - some things are just broken (**PImage's updatePixels()**)
 - posted code will work in Java mode for you

2D or 1D?

An image of width 100 pixels

- First pixel at index 0
- Right-most pixel in first row at index 99
- First pixel of second row at index 100

0	1	2	3	...	98	99
100	101	102	103	...	198	199
200	201	202	203	...	298	299
300	301	302	303	...	398	399
400	401	402	403	...	498	499
500	501	502	503	...	598	599
600	601	602	603	...	698	699
700	701	702	703	...	798	799
800	801	802	803	...	898	899
...

The `pixels[]` array is one-dimensional

0	1	2	3	...	98	99	100	101	102	103	...	198	199	200	101	102	103	...
---	---	---	---	-----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Accessing Pixels as a 2D Array

- Pixels can be accessed as a 2D array using the following formula:

```
index = r*width + c
index = y*width + x
```

- Using 0-based indices

```
int idx = r*width + c;
pixels[idx] = color(255);
```

Examples

- whiteLine

What does this program do?

```
void setup() {
  size(400, 400);

  // Load colors into the pixels array
  loadPixels();

  // Access pixels as a 2D array
  for (int y=0; y<height; y++) {
    for (int x=0; x<width; x++) {

      // Compute distance to center
      float d = dist(x, y, width/2, height/2);

      // Set pixel as distance to center
      pixels[y*height+x] = color(d);
    }
  }

  // Update the sketch with pixel data
  updatePixels();
}
```

PImage

```
PImage img = loadImage("myImage.jpg");
image(img, 0, 0);
```

- The PImage object

Fields

width - the width of the image
 height - the height of the image
 pixels[] - the image pixel colors (after a call to loadPixels())

Methods

loadPixels() - loads the pixels for this image to pixels[] array
 updatePixels() - updates the image with the data in pixels[]
 resize() - changes the size of this image

Examples

- blackWhite2

PImage

Methods (Cont'd)

get(...) Reads the color of any pixel or grabs a rectangle of pixels
 set(...) Writes a color to any pixel or writes an image into another
 copy(...) Copies pixels from one part of an image to another
 mask(...) Masks part of the image from displaying
 save(...) Saves the image to a TIFF, TARGA, PNG, or JPEG file
 resize(...) Changes the size of an image to a new width and height
 blend(...) Copies a pixel or rectangle of pixels using different blending modes
 filter(...) Processes the image using one of several algorithms

get(...)

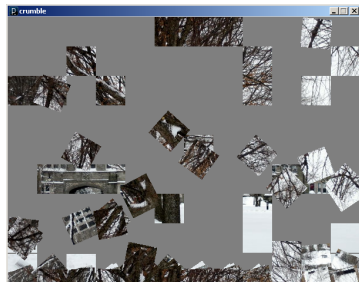
- Get a single pixel (very slow)

```
Color c = img.get(x, y);
```
- Get a rectangular range of pixels

```
PImage img2 = img.get(x, y, w, h);
```

Example

- crumble
- reassemble



```
PImage[] img = new PImage[5];
int alpha = 255;
int i = 0, j = 1;

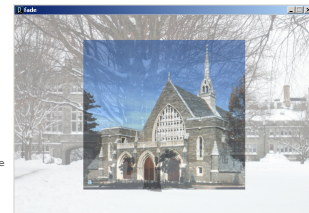
void setup() {
  size(600,400);
  imageMode(CENTER);
  for (int i=0; i<img.length; i++) // Load images
    img[i] = loadImage("bmc"+i+".jpg");
}

void draw() {
  background(255);

  // Fade out current image
  tint(255, alpha);
  image(img[i], 300, 200);

  // Fade in next image
  tint(255, 255-alpha);
  image(img[j], 300, 200);
  alpha--;

  // Swap images when fade complete
  if (alpha < 0) {
    i = (i + 1) % img.length;
    j = (j + 1) % img.length;
    alpha = 255;
  }
}
```

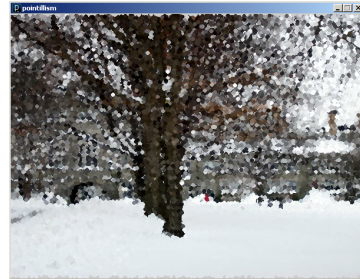


Examples

- fade
- fade2

Example

- pointillism



Simple Image Visualization

- Sample pixel colors every n pixels
- Draw a grid of basic shapes (ellipse, rect, line, triangle, etc) using the sampled color as fill color or stroke color

Example

- imageVis