Section 12: Images

Introduction: Images are just rectangular grids of pixels, with each pixel corresponding to some color data. Processing includes a special data type `PImage` for storing image data, which can also be converted to and from a special array called `pixels[]`. Here we will cover some basic ways to use images in your programs.

Importing an Image: You first must declare a variable of type `PImage`. To import an image, you need to call the special function `loadImage(String filename)` and then store its return value into your `PImage` variable. `filename` specifies the path to your image file.

- It is easiest if you put your image file into your Processing project folder and then you can just use the image name as the argument.
- Images should be imported *once* at the beginning of your program (i.e. inside `setup()`).

Example:

```
PImage myImg;
void setup() {
   myImg = loadImage("justin.jpg");
}
```

Displaying an Image: Once you have imported an image, you can display it on the drawing canvas:

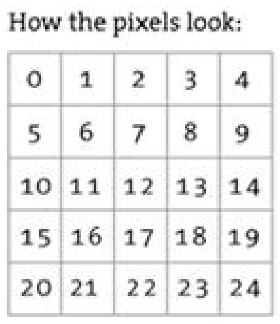
- `image(PImage img, float x, float y);` displays your image at its full size with its upper-left corner at the coordinate (**x**,**y**).
- `image(PImage img, float x, float y, float w, float h);` displays your image resized to width **w** and height **h** with its upper-left corner at the coordinate (**x**,**y**).

Example:

```
void draw() {
   image(myImg,0,0);   // redraws the image every frame
}
```

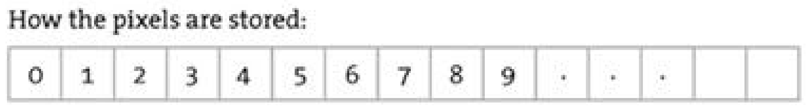
Pixel Data: You can get the color data of the current drawing canvas in a special array `color[] pixels` (note that it is color-coded the same as other system variables) by calling the command `loadPixels()`.

Although the canvas is two-dimensional, `pixels` is a one-dimensional array of length `width`×`height`:



How the pixels look:

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24



How the pixels are stored:

0	1	2	3	4	5	6	7	8	9	.	.	.		

Should you choose to manipulate the color data stored in pixels, you can then display the updated color data on the drawing canvas by calling the command `updatePixels()`.

Example:

```
void setup() {
   size(100,200);
   background(255,0,0);
   loadPixels();  // pixels[] will have length 100*200 and every index
}                 //    currently holds color(255,0,0);
void draw() {
   pixels[0] = color(0,0,0);  // set upper-left pixel to black
   updatePixels();            // update canvas with data from pixels[]
}
```

Extracting Color Data: The three functions `red(color c)`, `green(color c)`, and `blue(color c)` will return the individual color value (between `0-255`, as a `float`) of the color argument.

Exercises:

1) Write out a short Processing program below that loads an image called `pic.png` from the project folder and completely covers a drawing canvas of size 300 × 500.

2) Describe what does the following code does. At what point will this program run into an error?

```
void setup() {
   size(200,100);
}
void draw() {
   loadPixels();
   pixels[frameCount] = color(0,255,0);
   updatePixels();
}
```

3) Go to the course website and start working on the lab titled "Color Filters." [*partners*]