Review

- We can declare an array of any type, even other arrays
- A 2D array is an “array of arrays”
  ```java
  float[][] myFloats = new float[10][20];
  ```
- All elements of a 2D array can be accessed using nested loops
  ```java
  for (int i=0; i<10; i++) {
      for (int j=0; j<20; j++) {
          myFloats[i][j] = random(100);
      }
  }
  ```
- A ragged array is an array of nonuniformly sized arrays

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

Color

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

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

A 100-pixel wide image

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

The pixels[] array is one-dimensional
// shade
void setup() {
  size(100, 100);
  float b = 0.0;
  // Load colors into the pixels array
  loadPixels();
  // Fill pixel array with a grayscale value
  // based on pixel array index
  for (int i=0; i<10000; i++) {
    b = map(i, 0, 10000, 0, 255);
    pixels[i] = color(b);
  }
  // Update the sketch with pixel data
  updatePixels();
}

// whiteNoise
int nPixels;
void setup() {
  size(400, 300);
  nPixels = width*height;
}
void draw() {
  float b;
  // Load colors into pixels array
  loadPixels();
  // Fill pixel array with a random grayscale value
  for (int i=0; i<nPixels; i++) {
    b = random(0, 255);
    pixels[i] = color(b);
  }
  // Update the sketch with pixel data
  updatePixels();
}

Accessing Pixels as a 2D Array

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

  Index = (Col-1) + #Columns * (Row-1)

  Check it ...

- Using 0-based indexes...

  int i = c + width*r;
  pixels[i] = color(b);

Rendering Images in a Sketch

- Image data can be loaded from a file using loadImage() method, and drawn on a sketch with the image() command

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

- The PImage object also permits individual pixel color data to be modified.

  – like the sketch window
Animating with Images

- Animations can be created using
  - Arrays of PImage objects, and
  - Transformations

```java
PImage[] sequence = new PImage[15];
int seqNum = 0;

void setup() {
  size(180, 138);
  // Load images into array
  for (int i=0; i<15; i++) {
    String fileName = "horse" + (i+1) + "\.gif";
    sequence[i] = loadImage(fileName);
  }
  // Set frame rate
  frameRate(16);
}

void draw() {
  // Draw a new image on each draw
  image(sequence[seqNum], 0, 0);
  seqNum = (seqNum + 1) % 15;
}
```

// sequence2

```java
PImage[] sequence = new PImage[15];
int seqNum = 0;
float offsetX = 0.0;

void setup() {
  size(800, 138);
  // Load images into array
  for (int i=0; i<15; i++) {
    String fileName = "horse" + (i+1) + "\.gif";
    sequence[i] = loadImage(fileName);
  }
  // Set frame rate
  frameRate(16);
}

void draw() {
  // translate and draw a new image on each draw
  background(230);
  translate(xOffset, 0);
  image(sequence[seqNum], 0, 0);
  seqNum = (seqNum + 1) % 15;
  offsetX = (offsetX + 20.0) % width;
}
```

PImage

**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 color data out of the PImage object into a 1D array of colors named `pixels[]`.
- `updatePixels()`: Copies the color data from the `pixels[]` array back to the PImage object.
- `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
```java
void setup() {
  size(750, 327);
  // Load the image three times
  PImage warhol_r = loadImage("andy-warhol2.jpg");
  PImage warhol_g = loadImage("andy-warhol2.jpg");
  PImage warhol_b = loadImage("andy-warhol2.jpg");
  // Load pixels
  warhol_r.loadPixels();
  warhol_g.loadPixels();
  warhol_b.loadPixels();
  // Remove color components
  color c;
  for (int i = 0; i < warhol_r.pixels.length; i++) {
    c = warhol_r.pixels[i];
    warhol_r.pixels[i] = color(red(c), 0, 0);
    c = warhol_g.pixels[i];
    warhol_g.pixels[i] = color(0, green(c), 0);
    c = warhol_b.pixels[i];
    warhol_b.pixels[i] = color(0, 0, blue(c));
  }
  // Draw modified images
  image(warhol_r, 0, 0);
  image(warhol_g, 250, 0);
  image(warhol_b, 500, 0);
}
```

get(...)  
- Get a single pixel (very slow)
  ```java
  Color c = img.get(x, y);
  ```
- Get a rectangular range of pixels
  ```java
  PImage img2 = img.get(x, y, w, h);
  ```

tint(...) / noTint()  
- tint() modifies the fill value for images
  ```java
  tint( gray );
  tint( red, green, blue );
  ```
- Turn off applied tint() values with noTint()
Extending Processing with Libraries

- New objects can be added to Processing by "importing" libraries of prewritten code
- An extensive set of libraries are available
  - Video
  - Networking
  - Hardware Interfaces (Serial, Arduino)
  - Graphics (OpenGL)
  - Sound
  - Animation
  - User Interfaces
  - ...

http://processing.org/reference/libraries/

Video Library

- Classes
  - Capture : Grabs images/frames from a camera
  - Movie : Read movie frames and play movies
  - MovieMaker : Create movies from scratch

- Importing ...
  - Add the following line to the top of your program

```java
import processing.video.*;
```
Creating a Video of Your Animation

- **MovieMaker object**
  - A class used to create QuickTime movies

  ```java
  MovieMaker mm = new MovieMaker(this, // Parent sketch width, height, // Sketch size "myMovie.mov", // Video file 30, // Frames per second MovieMaker.ANIMATION, // Codec MovieMaker.HIGH); // Quality
  ```

- **addFrame() method**
  - Adds a snapshot of the sketch to the move as a new frame.

- **finish() method**
  - Stops recording and closes the video file.

http://processing.org/reference/libraries/video/MovieMaker.html

A generic framework for recording animated sketches as a movie

```java
// Capture Video library
import processing.video.*;
import MovieMaker object
MovieMaker mm;
void setup();
void draw();
void keyPressed();
```

- Create new MovieMaker object with target movie file
  ```java
  mm = new MovieMaker(this, width, height, "myMovie.mov", 30,
  MovieMaker.ANIMATION, MovieMaker.HIGH);
  ```

- Print message when done
  ```java
  println("Finished");
  ```

- In all draw steps...
  ```java
  // Add current sketch display to movie as new frame
  mm.addFrame();
  ```

- Stop recording when space bar is pressed
  ```java
  void keyPressed();
  ```

```java
// Capture Video library
import processing.video.*;
import MovieMaker object
MovieMaker mm;
void setup();
void draw();
void keyPressed();
```

```
// Generic framework
// for recording
// animated sketches as
// a movie
```