Day 7

Media

Agenda

Data Folder

Image, Font

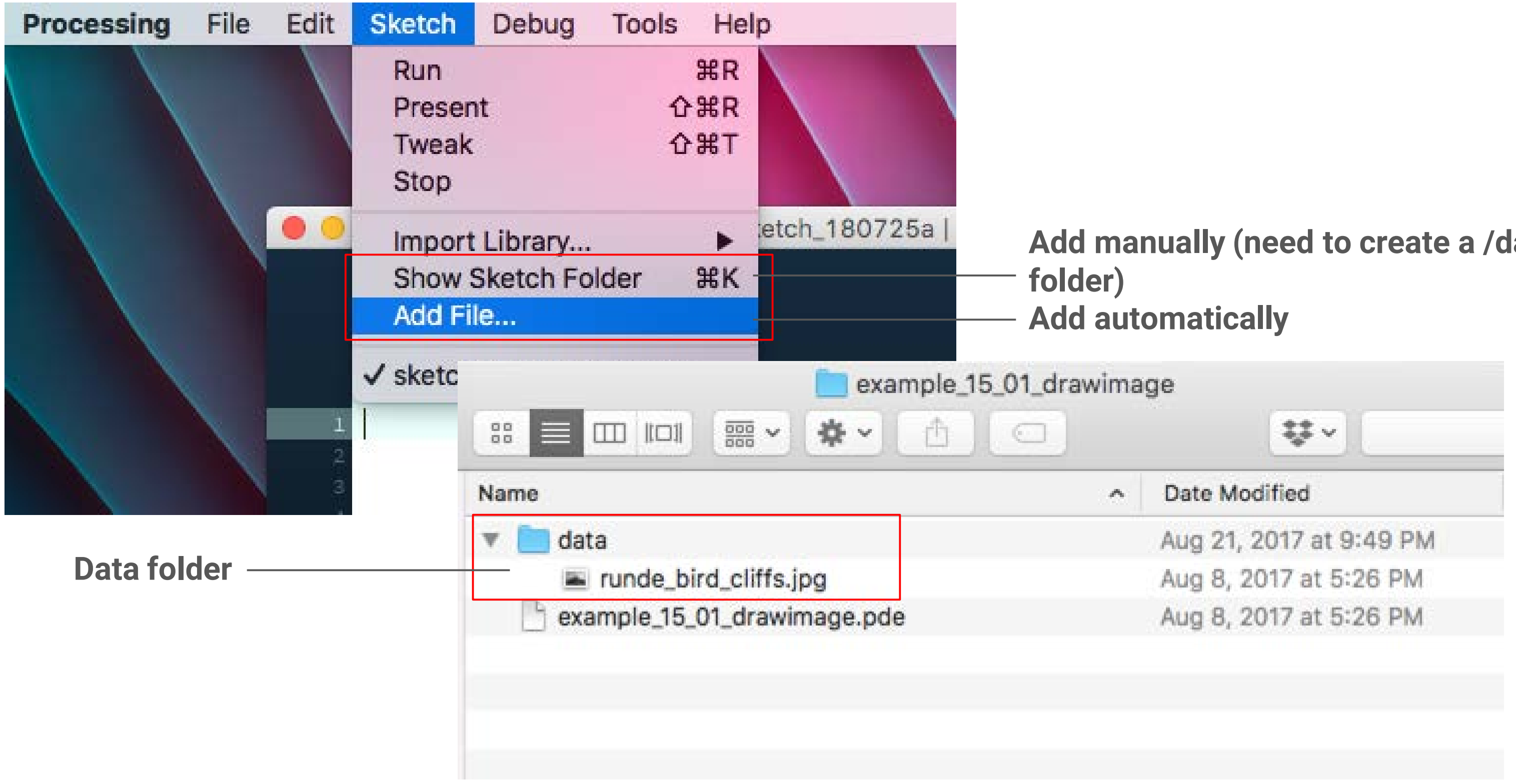
Array of Images

Transform: `translate()`, `rotate()`, `scale()`

`pushMatrix()`, `popMatrix()`

Webcam

Data Folder



Name	Date Modified
data	Aug 21, 2017 at 9:49 PM
runde_bird_cliffs.jpg	Aug 8, 2017 at 5:26 PM
example_15_01_drawimage.pde	Aug 8, 2017 at 5:26 PM

Add manually (need to create a /data folder)

Add automatically

Data folder

Image

```
PImage myImage; //Declare

void setup(){

  // Load
  myImage = loadImage ("image.jpg");

}

void draw(){

  // Draw
  image(myImage, x, y, width,
height);

}
```

Image

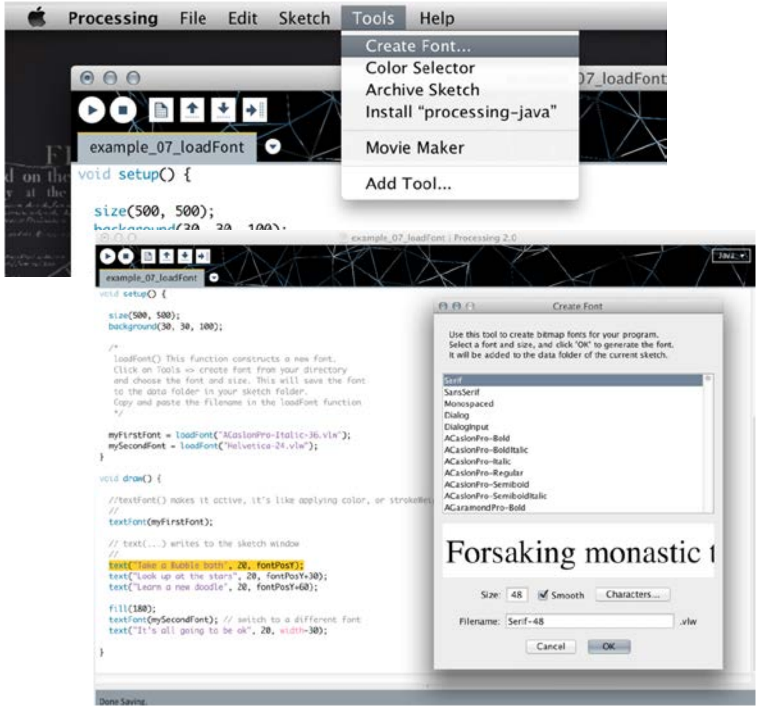
PImage is a Processing-defined *class*.

myImage is a new instance of PImage *object*.

An image is an *object*. It has a list of variables and functions like width, height and loadImage() as defined by the system.

```
PImage myImage; //Declare

void setup(){

  // Load
  myImage = loadImage (“image.jpg”);

}

void draw(){

  // Draw
  image(myImage, x, y, width,
height);

}
```

Font



```
void setup() {

  size(500, 500);
  background(30, 30, 100);

  /*
   loadFont() This function constructs a new font.
   Click on Tools => create font from your directory
   and choose the font and size. This will save the font
   to the data folder in your sketch folder.
   Copy and paste the filename in the loadFont function
   */

  myFirstFont = loadFont("ACaslonPro-Italic-36.vlw");
  mySecondFont = loadFont("Helvetica-24.vlw");
}

void draw() {

  //textFont() makes it active, it's like applying color, or strokeWei
  //
  textFont(myFirstFont);

  // text(...) writes to the sketch window
  //
  text("Take a Bubble bath", 20, fontPosY);
  text("Look up at the stars", 20, fontPosY+30);
  text("Learn a new doodle", 20, fontPosY+60);

  fill(180);
  textFont(mySecondFont); // switch to a different font
  text("It's all going to be ok", 20, width-30);

}
```

Use this tool to create bitmap fonts for your program.
Select a font and size, and click 'OK' to generate the font.
It will be added to the data folder of the current sketch.

- Serif
- SansSerif
- Monospaced
- Dialog
- DialogInput
- ACaslonPro-Bold
- ACaslonPro-BoldItalic
- ACaslonPro-Italic
- ACaslonPro-Regular
- ACaslonPro-Semibold
- ACaslonPro-SemiboldItalic
- AGaramondPro-Bold

```
PFont font; //Declare

void setup(){

  //Load
  font = loadFont (“fontname-size.vlw”);

}

void draw(){

  //Switch font to use
  textFont(font, size);
  //Place text
  text(“Hello World”, xPos, yPos);

}
```

Font

Processing displays fonts using the .vlw font format.

loadFont() - construct a new font
textFont() - activate the font and specify the size

```
PFont font; //Declare

void setup(){

  //Load
  font = loadFont (“fontname-size.vlw”);

}

void draw(){

  //Switch font to use
  textFont(font, size);
  //Place text
  text(“Hello World”, xPos, yPos);

}
```

Live Code
Simple Animation

Let's create a simple animation with the array of images (copy the images from drive **/day07_gifanimation/data**).

Try using LEFT and RIGHT key to control the character. What happened?

Transform

translate()

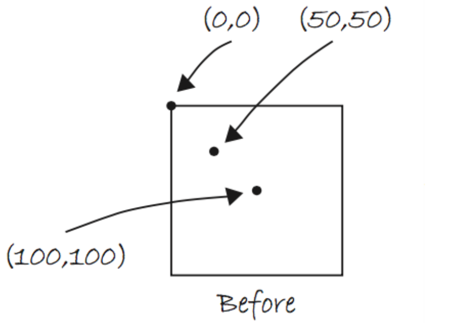
Processing window works like a piece of graph paper.

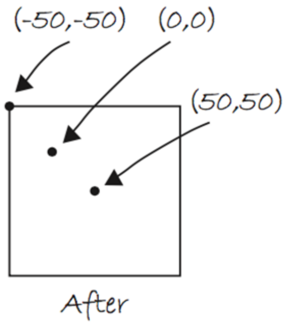
translate() does not change the position of your drawing. It changes the “graph paper” - the origin point and the coordinate system.

Think about drawing one shape vs. drawing multiple shapes in a loop. What is the advantage?

```
void setup(){

   size(200, 200);

}

void draw(){

   rect(0, 0, 50, 50);

   //move the origin point(0, 0)
   translate(50, 50);

   rect(0, 0, 50, 50);

}
```

translate()





```
void setup(){

  size(200,200);

}

void draw(){

  rect(0,0,50,50);

  //move the origin point(0,0)
  translate(50,50);

  rect(0,0,50,50);

}
```

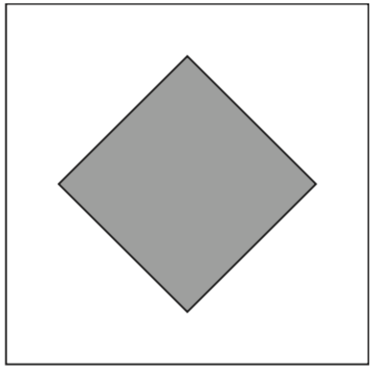
rotate()

rotate() rotates the coordinate system and measures angles in radians. A full circle has 2π radians.

radians() converts degrees to radians.

```
void setup(){

  size(200,200);

}

void draw(){

  //translate to center of window
  translate(width/2,height/2);
  //rotate by 45 degree clockwise
  rotate(radians(45));
  rectMode(CENTER);
  rect(0,0,100,100);

}
```

rotate()



```
void setup(){

  size(200,200);

}

void draw(){

  //translate to center of window
  translate(width/2,height/2);
  //rotate by 45 degree clockwise
  rotate(radians(45));
  rectMode(CENTER);
  rect(0,0,100,100);

}
```

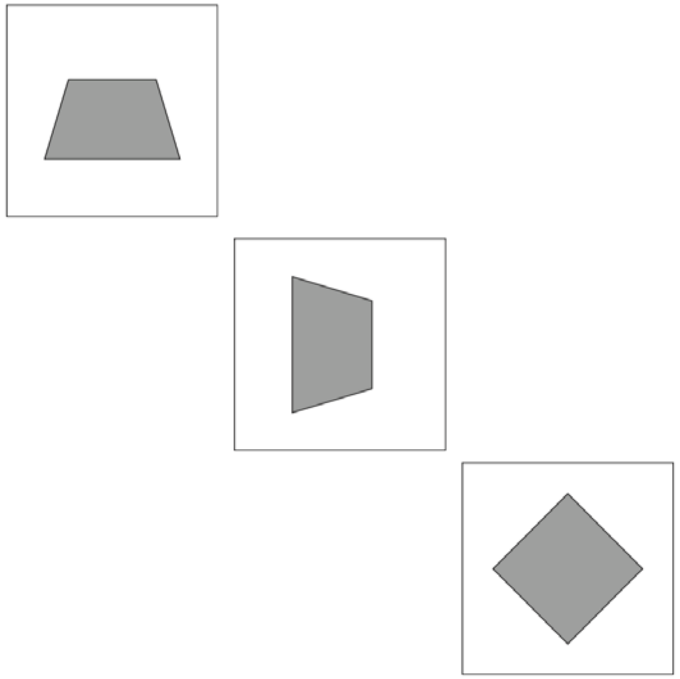
Rotation around different axes

`rotateX()`, `rotateY()`, `rotateZ()` rotates an angle around an axis.

A 3D renderer is required for these functions.

```
size(200, 200, P3D);

rotateX(theta);

rotateY(theta);

rotateZ(theta);
```

Rotation around different axes



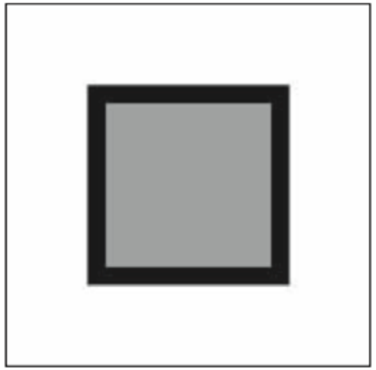
```
size(200, 200, P3D);

rotateX(theta);

rotateY(theta);

rotateZ(theta);
```

scale()

scale() increases the dimensions of an shape relative to the origin by a percentage (1.0 equals to 100%).

```
float r=0.0;

void setup(){

  size(200,200);

}

void draw(){

  translate(width/2,height/2);
  scale(r);
  rectMode(CENTER);
  rect(0,0,10,10);
  r += 0.02;

}
```

scale()



```
float r=0.0;

void setup(){

  size(200,200);

}

void draw(){

  translate(width/2,height/2);
  scale(r);
  rectMode(CENTER);
  rect(0,0,10,10);
  r += 0.02;

}
```

pushMatrix()
popMatrix()

“What is the Matrix?” — Neo

pushMatrix() & popMatrix()

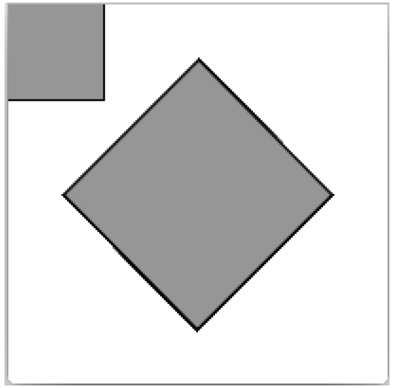
pushMatrix() stores the current status of the coordinate system at the top of a memory area.

popMatrix() pulls that status back out.

This will allow us to move and rotate individual shapes without them affecting others.

```
pushMatrix();
translate(width/2, height/2);
rotate(radians(45));
rectMode(CENTER);
rect(0, 0, 100, 100);
popMatrix();
rect(0, 0, 100, 100);
```

pushMatrix() & popMatrix()

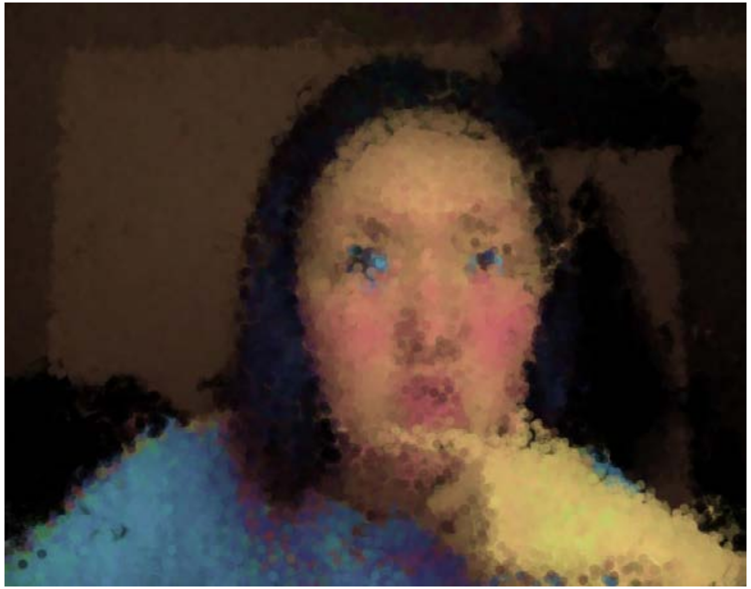


```
pushMatrix();
translate(width/2, height/2);
rotate(radians(45));
rectMode(CENTER);
rect(0, 0, 100, 100);
popMatrix();
rect(0, 0, 100, 100);
```

Live Code

Solar System

Capture



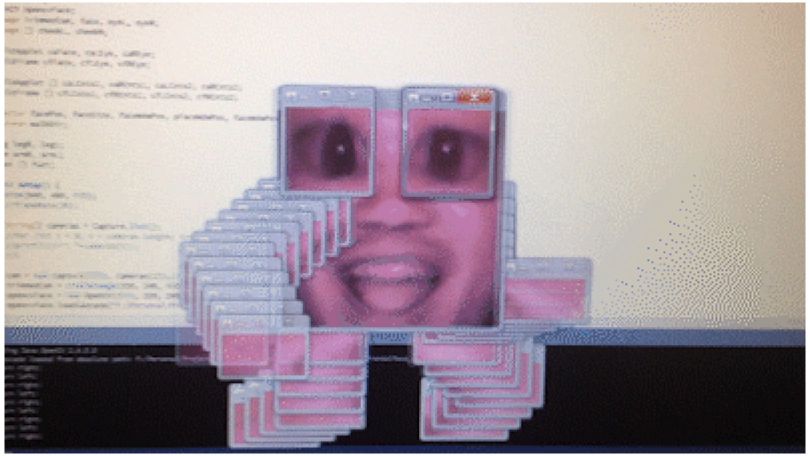
```
import processing.video.*;

Capture cam;

void setup() {
  size(640, 480);

  String[] cameras = Capture.list();
  cam = new Capture(this,
cameras[0]);
  cam.start();
}

void draw() {
  cam.read();
  image(cam, 0, 0);
}
```

Live Code

Manipulate Capture

Homework

Play with the live video captured from your webcam



ravenkwok.com