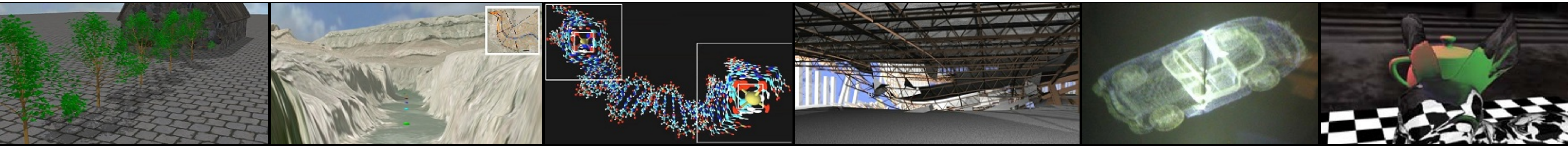
CIS 4930/6930-002
DATA VISUALIZATION

INTRODUCTION TO PROCESSING

Paul Rosen
Assistant Professor
University of South Florida

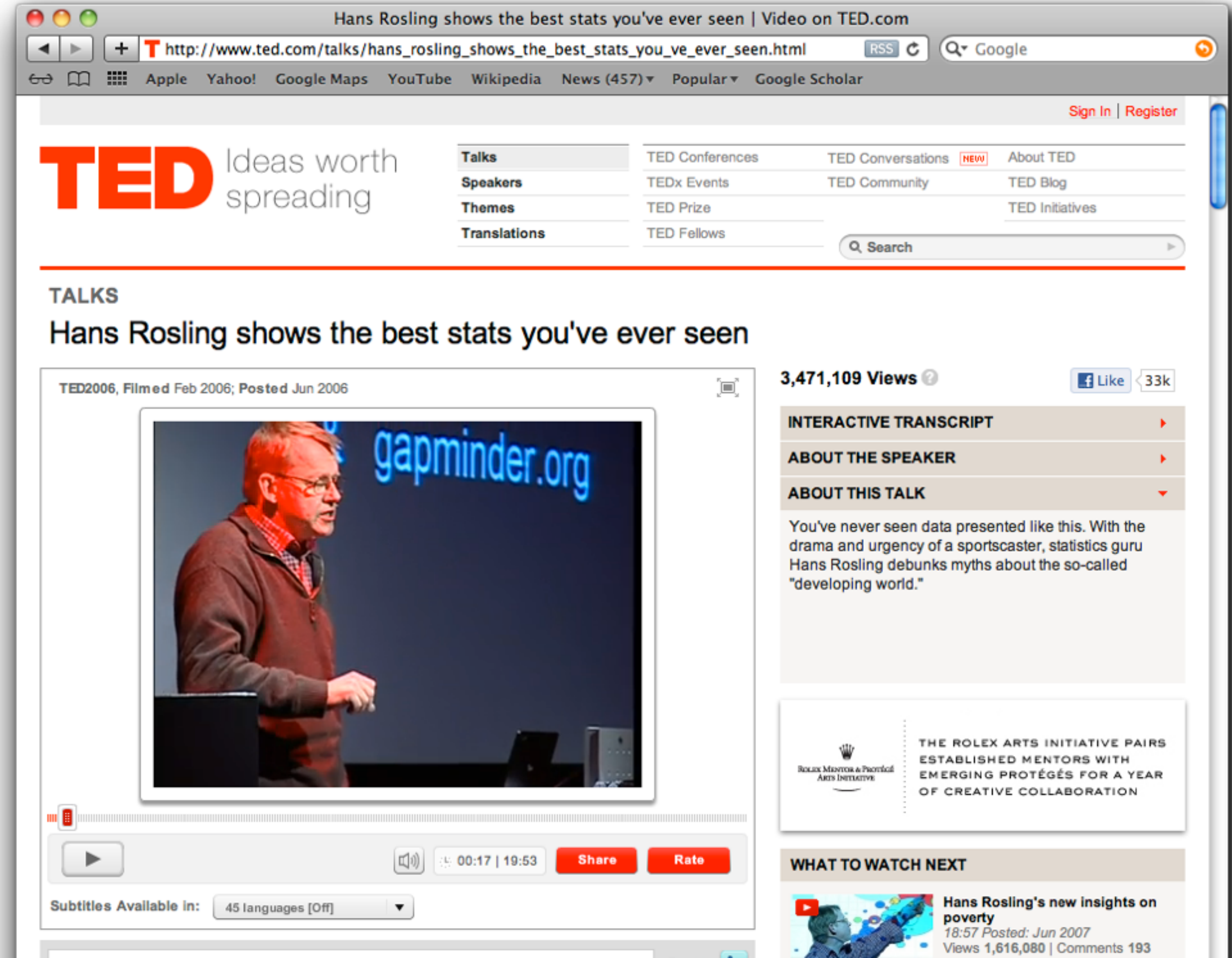
(slide acknowledgments: Hitesh Raju)



REMINDERS

Project #1 Due Next Class (1/17)

Project #2 posted



TALKS

Hans Rosling shows the best stats you've ever seen

TED2006, Filmed Feb 2006; Posted Jun 2006

3,471,109 Views

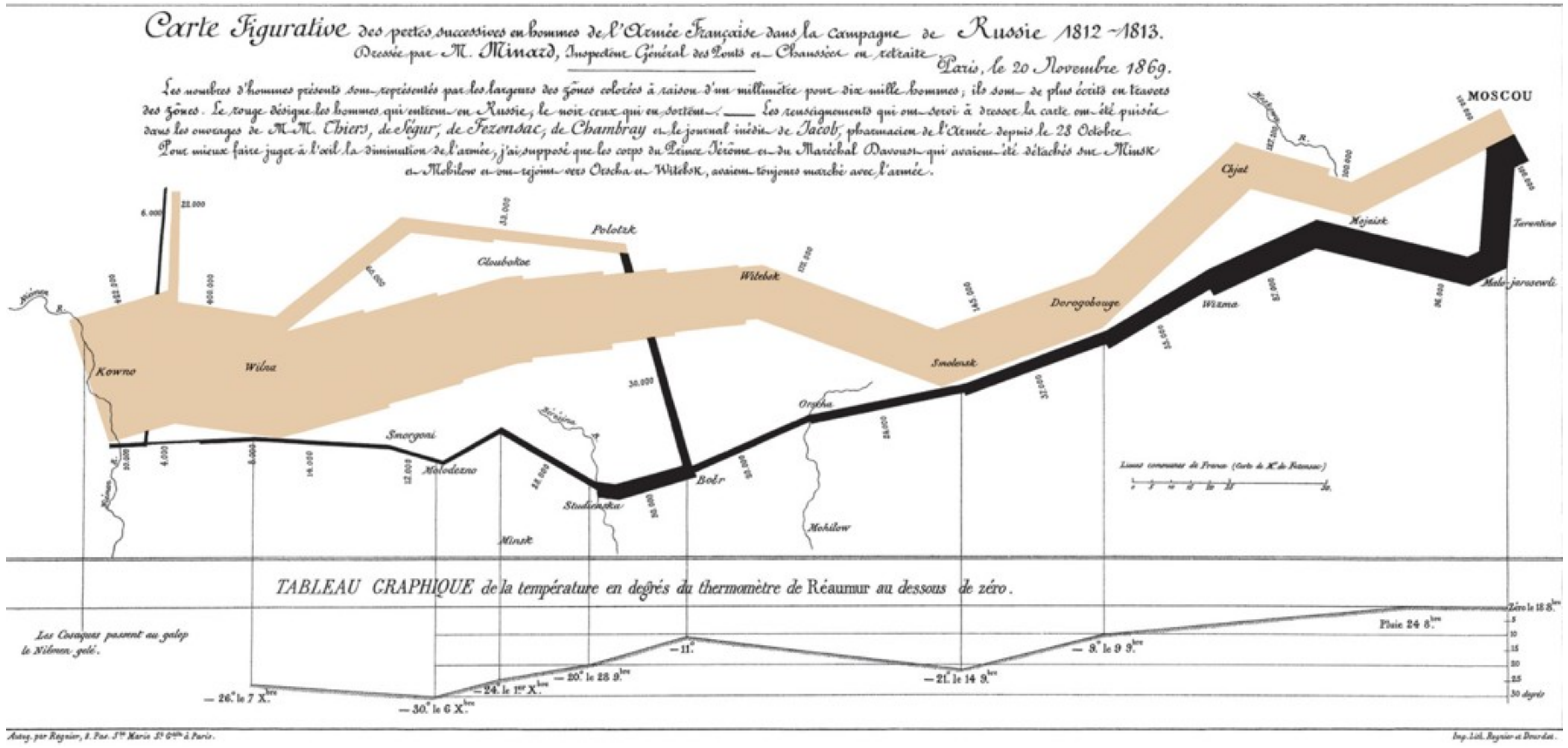
Subtitles Available in: 45 languages [Off]

You've never seen data presented like this. With the drama and urgency of a sportscaster, statistics guru Hans Rosling debunks myths about the so-called "developing world."

Rolex Mentor & Protégé Arts Initiative

THE ROLEX ARTS INITIATIVE PAIRS ESTABLISHED MENTORS WITH EMERGING PROTÉGÉS FOR A YEAR OF CREATIVE COLLABORATION

Hans Rosling's new insights on poverty
18:57 Posted: Jun 2007
Views 1,616,080 | Comments 193


Carte Figurative des pertes successives en hommes de l'Armée Française dans la campagne de Russie 1812–1813.
Dressée par M. Minard, Inspecteur Général des Ponts et Chaussées en retraite.
Paris, le 20 Novembre 1869.

Les nombres d'hommes présents sont représentés par les largeurs des zônes colorées à raison d'un millimètre pour dix mille hommes; ils sont de plus écrits en travers des zônes. Le rouge désigne les hommes qui entrent en Russie, le noir ceux qui en sortent. —— Les renseignements qui ont servi à dresser la carte ont été puisés dans les ouvrages de M.M. Thiers, de Ségur, de Fezensac, de Chambray et le journal inédit de Jacob, pharmacien de l'Armée depuis le 28 Octobre.
Pour mieux faire juger à l'œil la diminution de l'armée, j'ai supposé que les corps du Prince Jérôme et du Maréchal Davoust qui avaient été détachés sur Minsk et Mobilow et ont rejoint vers Orscha et Witebsk, avaient toujours marché avec l'armée.

TABLEAU GRAPHIQUE de la température en degrés du thermomètre de Réaumur au dessous de zéro.

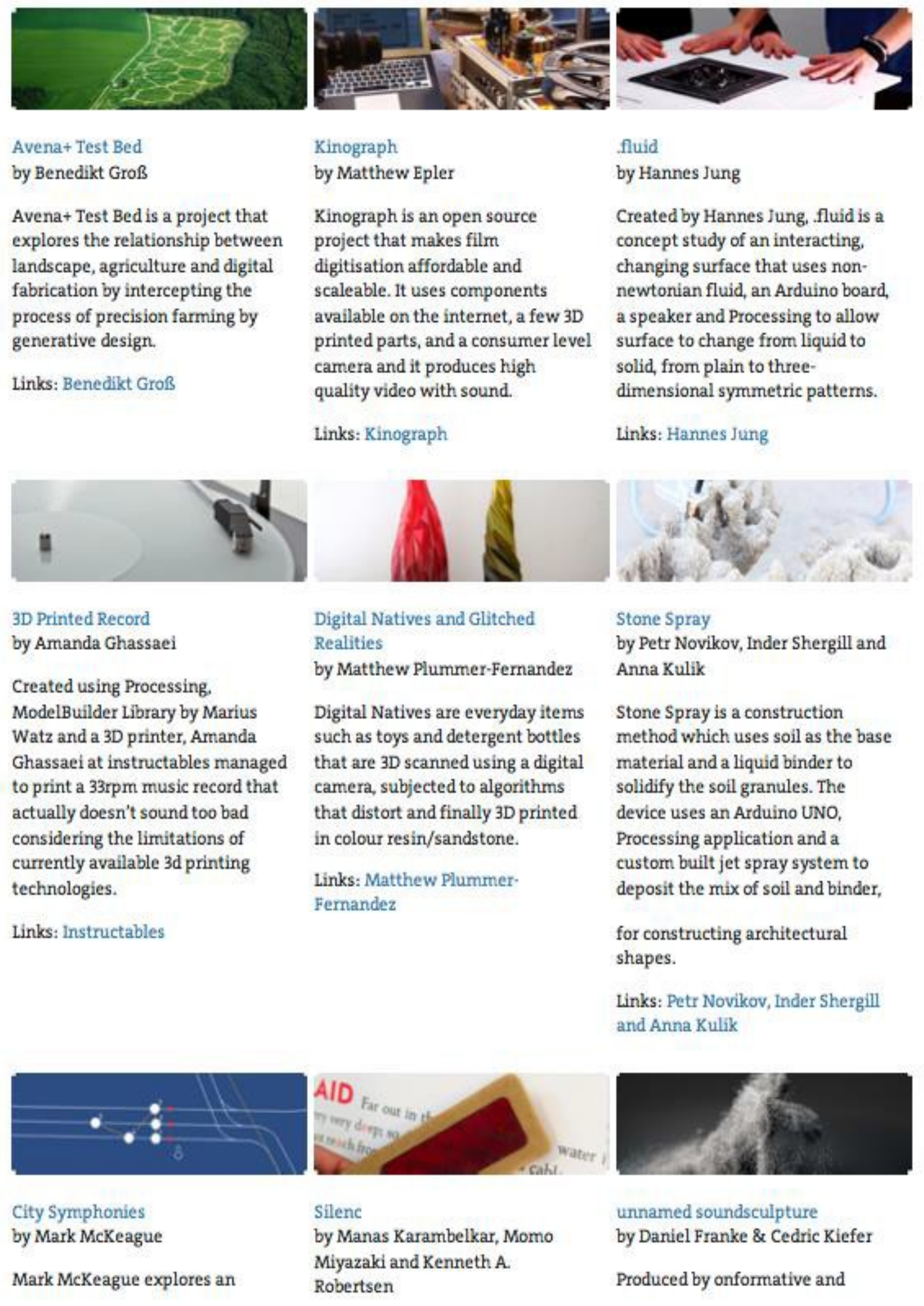
Joseph Minard 1861

WHAT IS IT?

programming environment

visually oriented applications

targets artists, designers, etc.



Avena+ Test Bed
by Benedikt Groß

Avena+ Test Bed is a project that explores the relationship between landscape, agriculture and digital fabrication by intercepting the process of precision farming by generative design.

Links: Benedikt Groß

Kinograph
by Matthew Epler

Kinograph is an open source project that makes film digitisation affordable and scaleable. It uses components available on the internet, a few 3D printed parts, and a consumer level camera and it produces high quality video with sound.

Links: Kinograph

.fluid
by Hannes Jung

Created by Hannes Jung, .fluid is a concept study of an interacting, changing surface that uses non-newtonian fluid, an Arduino board, a speaker and Processing to allow surface to change from liquid to solid, from plain to three-dimensional symmetric patterns.

Links: Hannes Jung

3D Printed Record
by Amanda Ghassaei

Created using Processing, ModelBuilder Library by Marius Watz and a 3D printer, Amanda Ghassaei at instructables managed to print a 33rpm music record that actually doesn't sound too bad considering the limitations of currently available 3d printing technologies.

Links: Instructables

Digital Natives and Glitched Realities
by Matthew Plummer-Fernandez

Digital Natives are everyday items such as toys and detergent bottles that are 3D scanned using a digital camera, subjected to algorithms that distort and finally 3D printed in colour resin/sandstone.

Links: Matthew Plummer-Fernandez

Stone Spray
by Petr Novikov, Inder Shergill and Anna Kulik

Stone Spray is a construction method which uses soil as the base material and a liquid binder to solidify the soil granules. The device uses an Arduino UNO, Processing application and a custom built jet spray system to deposit the mix of soil and binder,

for constructing architectural shapes.

Links: Petr Novikov, Inder Shergill and Anna Kulik

City Symphonies
by Mark McKeague

Mark McKeague explores an

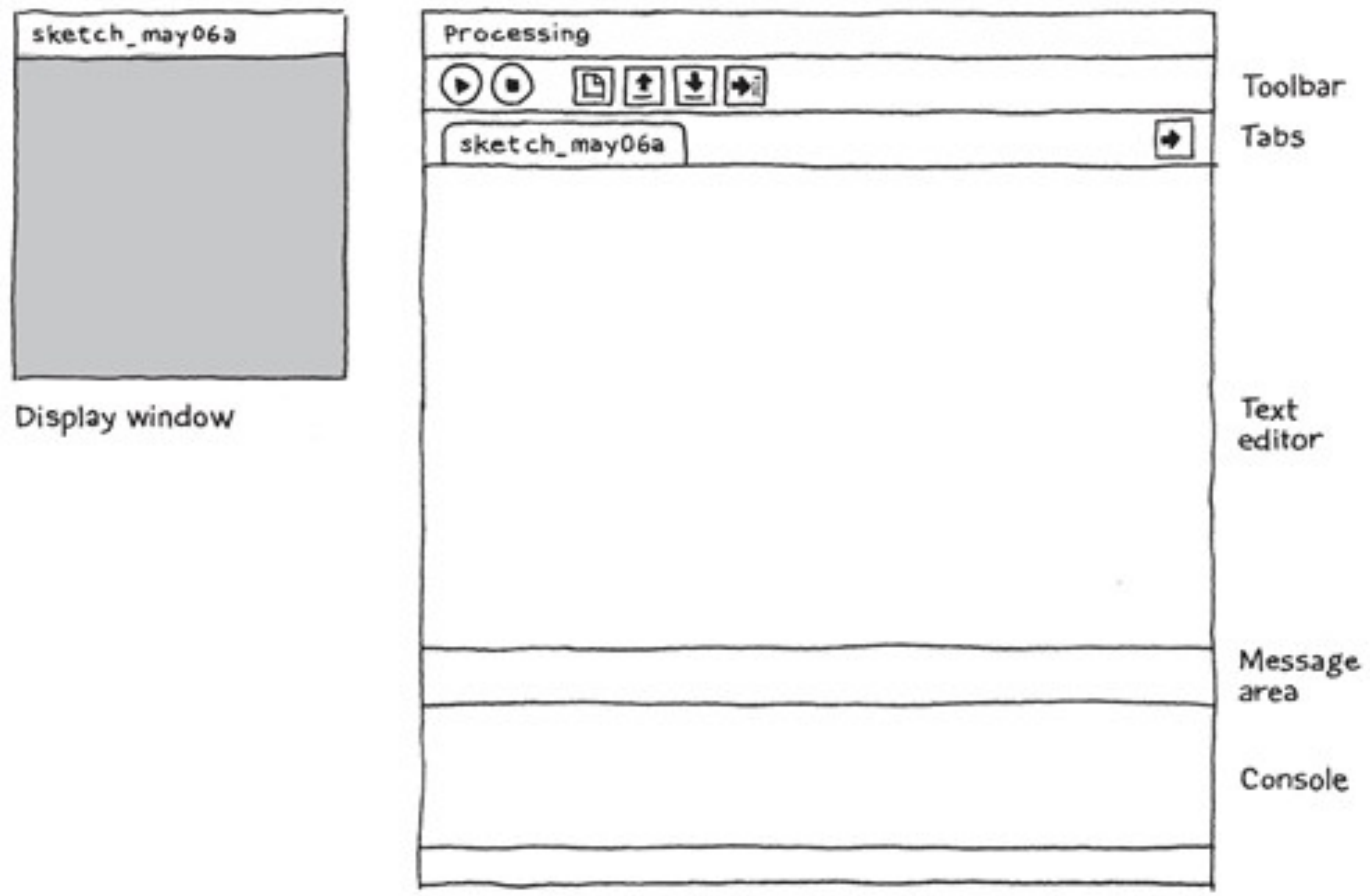
Silenc
by Manas Karambelkar, Momo Miyazaki and Kenneth A. Robertsen

unnamed soundsculpture
by Daniel Franke & Cedric Kiefer

Produced by onformative and

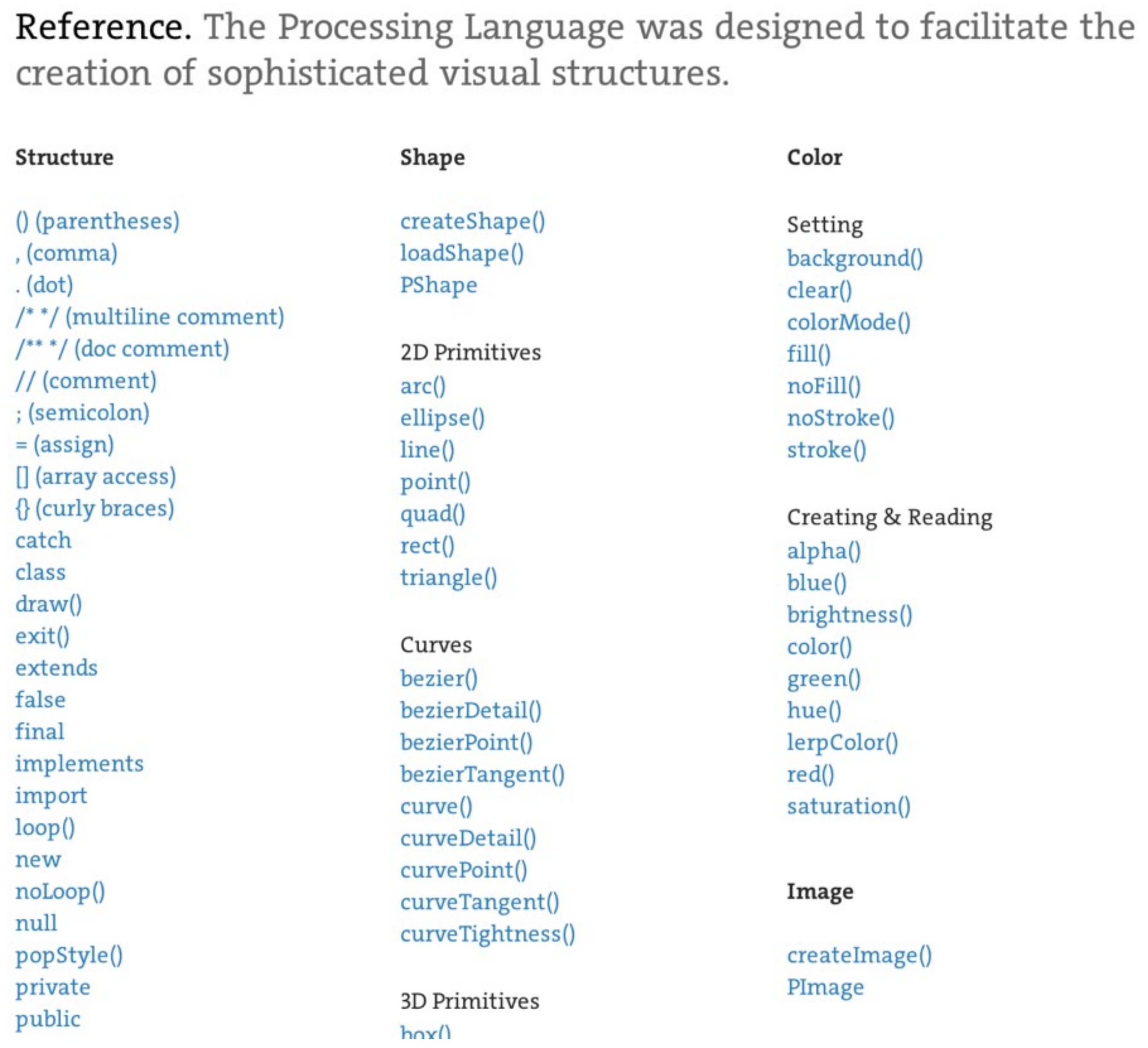
WHAT IS IT?

Processing Development Environment (PDE)



WHAT IS IT?

Processing API



Reference. The Processing Language was designed to facilitate the creation of sophisticated visual structures.

Structure

() (parentheses)
, (comma)
. (dot)
/* */ (multiline comment)
/** */ (doc comment)
// (comment)
; (semicolon)
= (assign)
[] (array access)
{} (curly braces)
catch
class
draw()
exit()
extends
false
final
implements
import
loop()
new
noLoop()
null
popStyle()
private
public

Shape

createShape()
loadShape()
PShape

2D Primitives
arc()
ellipse()
line()
point()
quad()
rect()
triangle()

Curves
bezier()
bezierDetail()
bezierPoint()
bezierTangent()
curve()
curveDetail()
curvePoint()
curveTangent()
curveTightness()

3D Primitives
box()

Color

Setting
background()
clear()
colorMode()
fill()
noFill()
noStroke()
stroke()

Creating & Reading
alpha()
blue()
brightness()
color()
green()
hue()
lerpColor()
red()
saturation()

Image

createImage()
PImage

WHAT IS IT?

open-source, online community

http://forum.processing.org/

https://github.com/processing

WHY PROCESSING?

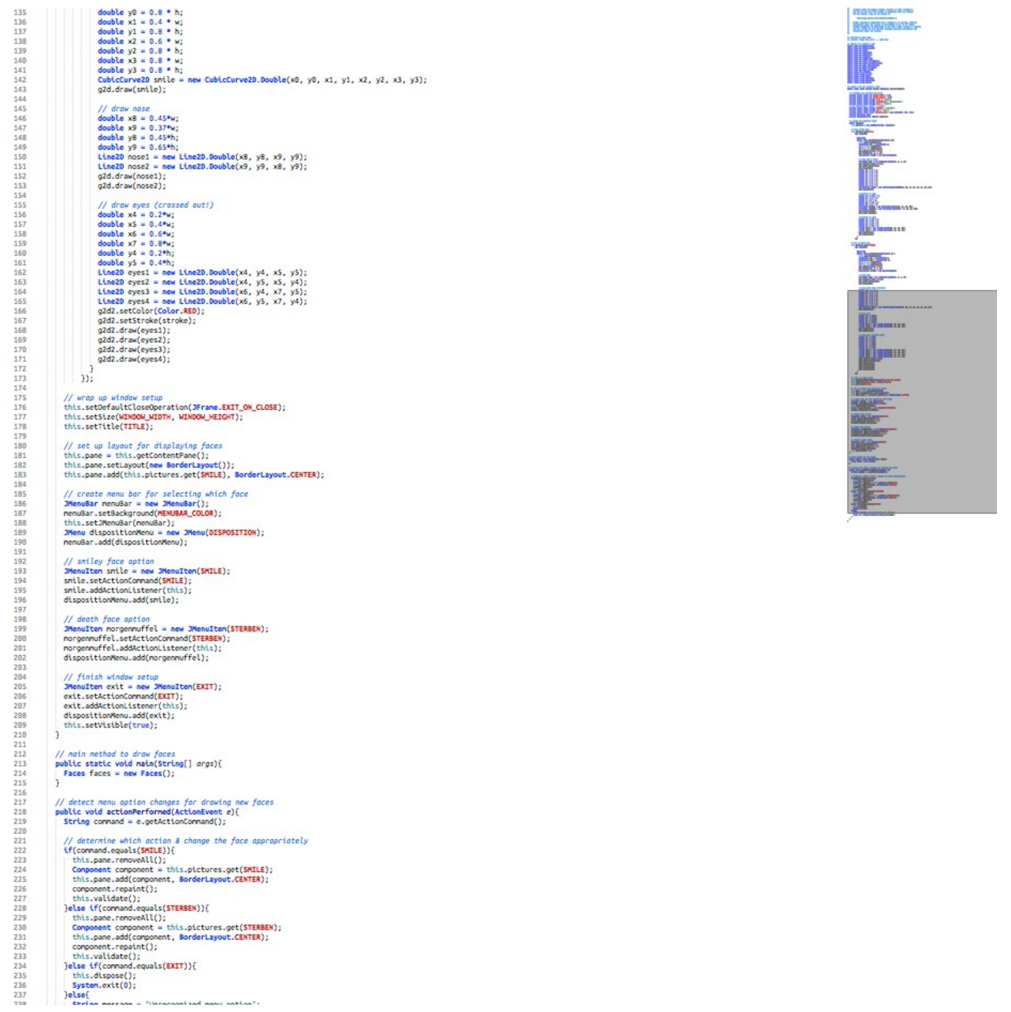
WHY PROCESSING?

difficulty to sketch with other languages

complicated setup

not easy to learn

repetitive code



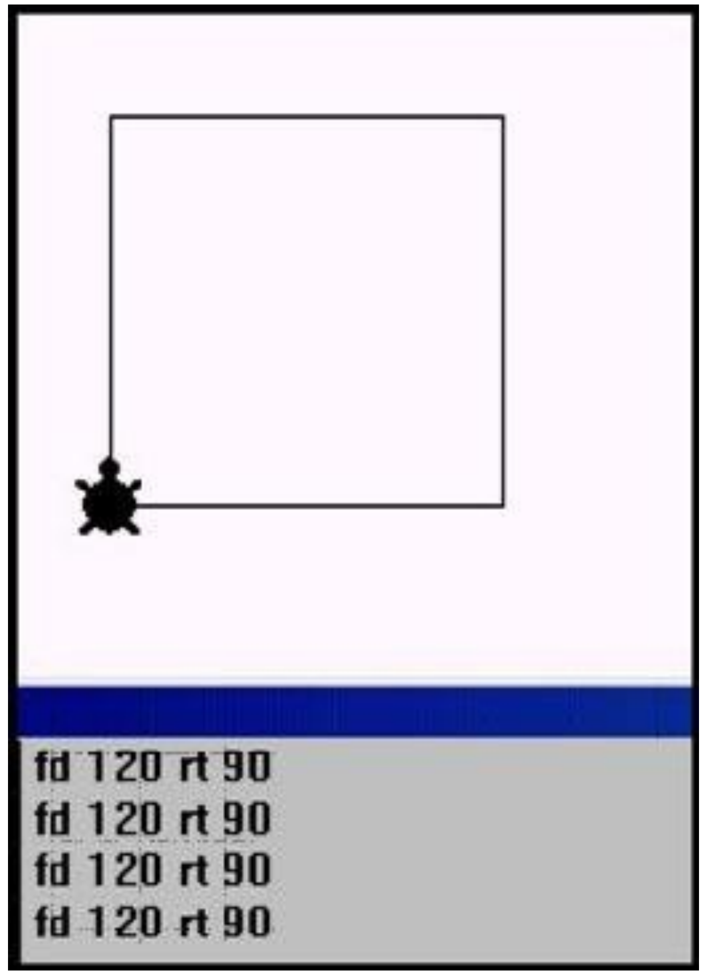
```java
            double y0 = 0.8 * h;
            double x1 = 0.4 * w;
            double y1 = 0.8 * h;
            double x2 = 0.6 * w;
            double y2 = 0.8 * h;
            double x3 = 0.8 * w;
            double y3 = 0.8 * h;
            CubicCurve2D smile = new CubicCurve2D.Double(x0, y0, x1, y1, x2, y2, x3, y3);
            g2d.draw(smile);

            // draw nose
            double x8 = 0.45*w;
            double x9 = 0.37*w;
            double y8 = 0.45*h;
            double y9 = 0.65*h;
            Line2D nose1 = new Line2D.Double(x8, y8, x9, y9);
            Line2D nose2 = new Line2D.Double(x9, y9, x8, y9);
            g2d.draw(nose1);
            g2d.draw(nose2);

            // draw eyes (crossed out!)
            double x4 = 0.2*w;
            double x5 = 0.4*w;
            double x6 = 0.6*w;
            double x7 = 0.8*w;
            double y4 = 0.2*h;
            double y5 = 0.4*h;
            Line2D eyes1 = new Line2D.Double(x4, y4, x5, y5);
            Line2D eyes2 = new Line2D.Double(x4, y5, x5, y4);
            Line2D eyes3 = new Line2D.Double(x6, y4, x7, y5);
            Line2D eyes4 = new Line2D.Double(x6, y5, x7, y4);
            g2d2.setColor(Color.RED);
            g2d2.setStroke(stroke);
            g2d2.draw(eyes1);
            g2d2.draw(eyes2);
            g2d2.draw(eyes3);
            g2d2.draw(eyes4);
          }
        });

    // wrap up window setup
    this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    this.setSize(WINDOW_WIDTH, WINDOW_HEIGHT);
    this.setTitle(TITLE);

    // set up layout for displaying faces
    this.pane = this.getContentPane();
    this.pane.setLayout(new BorderLayout());
    this.pane.add(this.pictures.get(SMILE), BorderLayout.CENTER);

    // create menu bar for selecting which face
    JMenuBar menuBar = new JMenuBar();
    menuBar.setBackground(MENUBAR_COLOR);
    this.setJMenuBar(menuBar);
    JMenu dispositionMenu = new JMenu(DISPOSITION);
    menuBar.add(dispositionMenu);

    // smiley face option
    JMenuItem smile = new JMenuItem(SMILE);
    smile.setActionCommand(SMILE);
    smile.addActionListener(this);
    dispositionMenu.add(smile);

    // death face option
    JMenuItem morgenmuffel = new JMenuItem(STERBEN);
    morgenmuffel.setActionCommand(STERBEN);
    morgenmuffel.addActionListener(this);
    dispositionMenu.add(morgenmuffel);

    // finish window setup
    JMenuItem exit = new JMenuItem(EXIT);
    exit.setActionCommand(EXIT);
    exit.addActionListener(this);
    dispositionMenu.add(exit);
    this.setVisible(true);
  }

  // main method to draw faces
  public static void main(String[] args){
    Faces faces = new Faces();
  }

  // detect menu option changes for drawing new faces
  public void actionPerformed(ActionEvent e){
    String command = e.getActionCommand();

    // determine which action & change the face appropriately
    if(command.equals(SMILE)){
      this.pane.removeAll();
      Component component = this.pictures.get(SMILE);
      this.pane.add(component, BorderLayout.CENTER);
      component.repaint();
      this.validate();
    }else if(command.equals(STERBEN)){
      this.pane.removeAll();
      Component component = this.pictures.get(STERBEN);
      this.pane.add(component, BorderLayout.CENTER);
      component.repaint();
      this.validate();
    }else if(command.equals(EXIT)){
      this.dispose();
      System.exit(0);
    }else{
      String message = "Unrecognized menu option";
```

WHY PROCESSING?

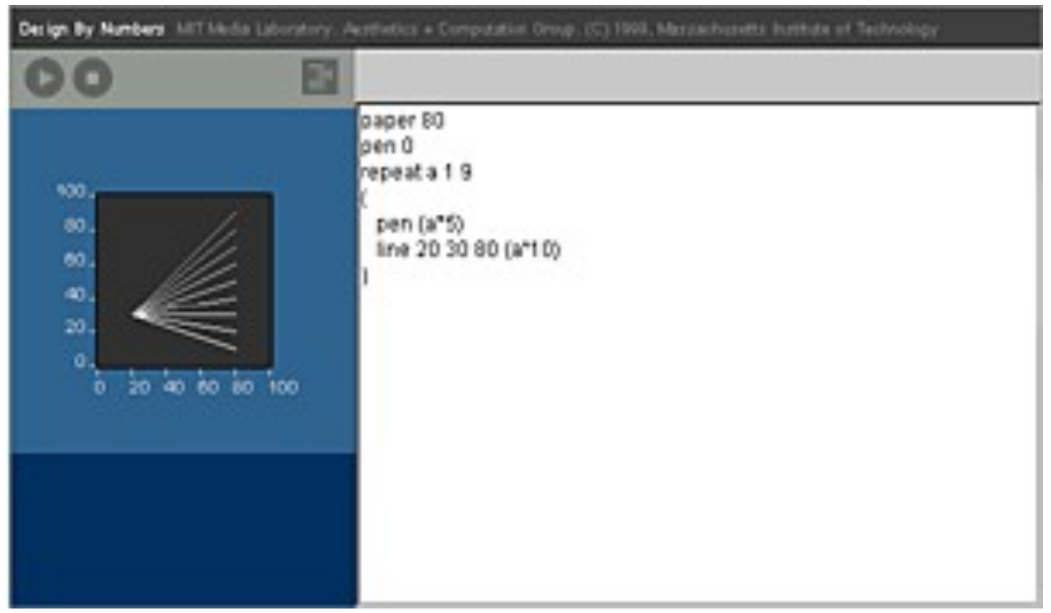
based on:

Logo

Design by Numbers



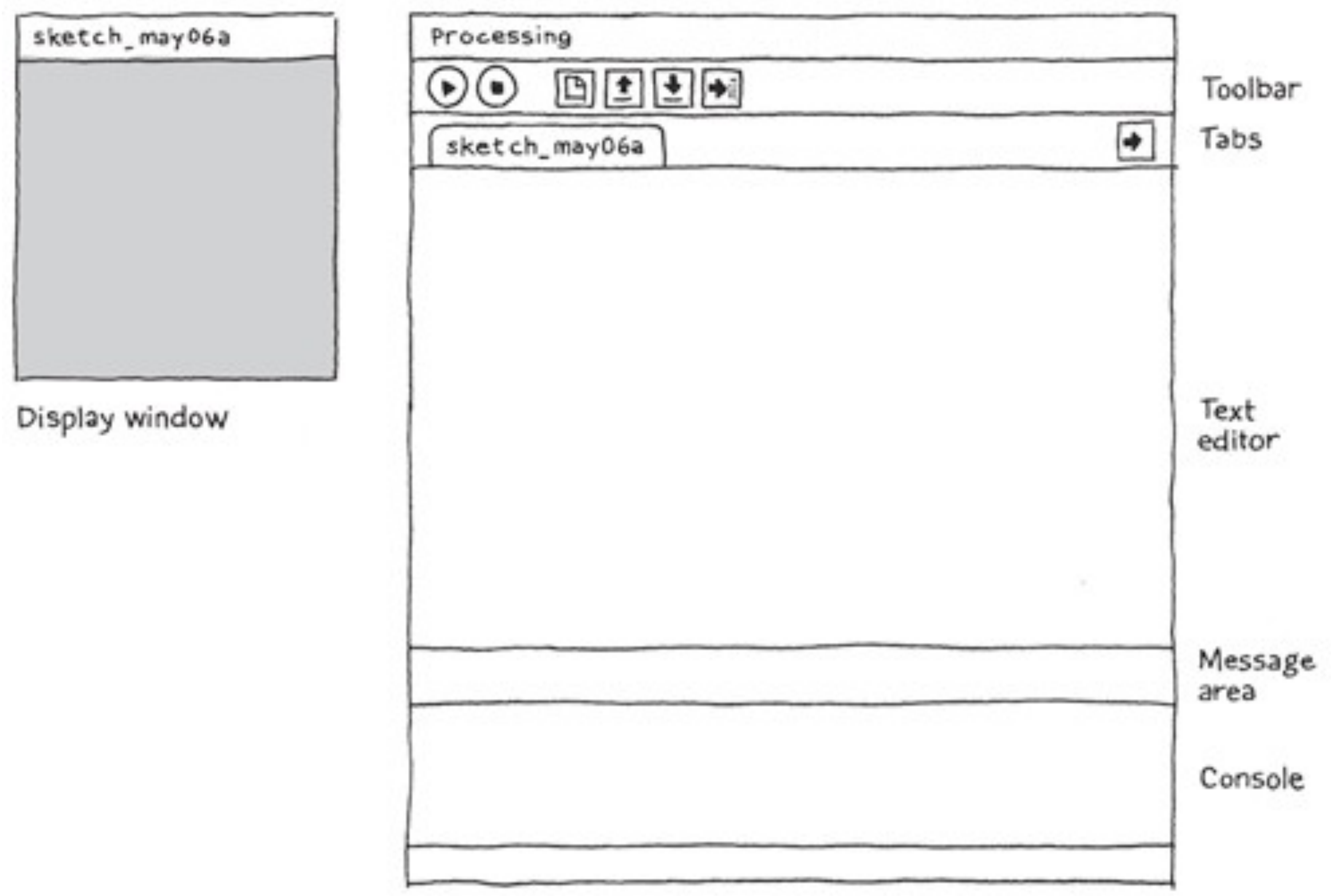
```
fd 120 rt 90
fd 120 rt 90
fd 120 rt 90
fd 120 rt 90
```


Design By Numbers MIT Media Laboratory, Aesthetics + Computation Group, (C) 1999, Massachusetts Institute of Technology
100
80
60
40
20
0
0 20 40 60 80 100

```
paper 80
pen 0
repeat a 1 9
{
  pen (a*5)
  line 20 30 80 (a*10)
}
```

WHY PROCESSING?

program = sketch



Why Processing?

programming syntax

```
void setup() {
  size(480, 120);
}

void draw() {
  if (mousePressed) {
    fill(0);
  } else {
    fill(255);
  }
  ellipse(mouseX, mouseY, 80, 80);
}
```

WHY PROCESSING?

Java-based

- complexity

+ big standard library

+ lots of user-contributed libraries

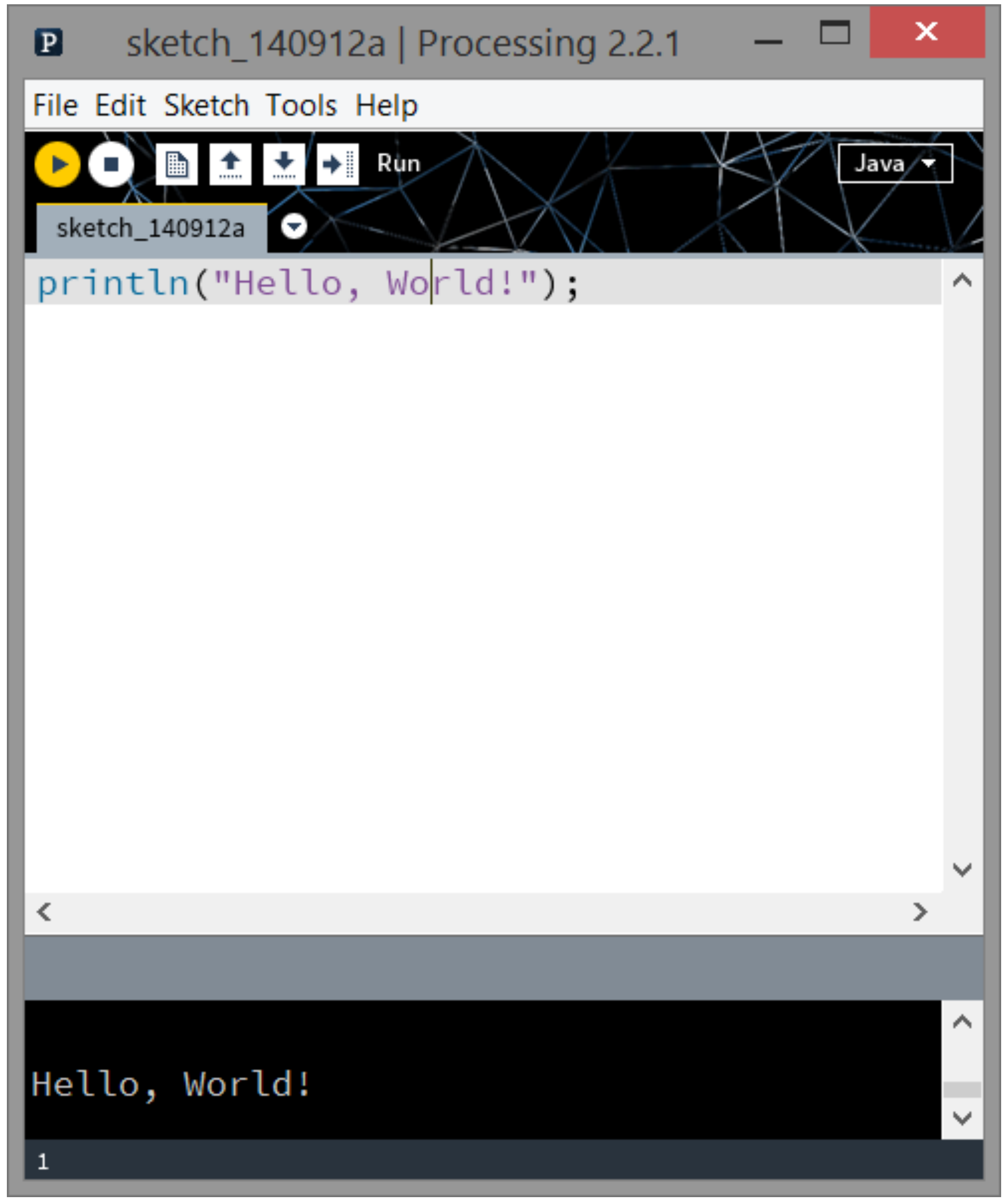
similar syntax & portability

WHY PROCESSING?

```
public class Hello
{
   public static void main (String args[])
   {
      System.out.println("Hello, world!");
   }
}

javac Hello.java
java Hello
```

WHY PROCESSING?

```
println(“Hello, World!”);
```


sketch_140912a | Processing 2.2.1
File Edit Sketch Tools Help
Run
Java
sketch_140912a

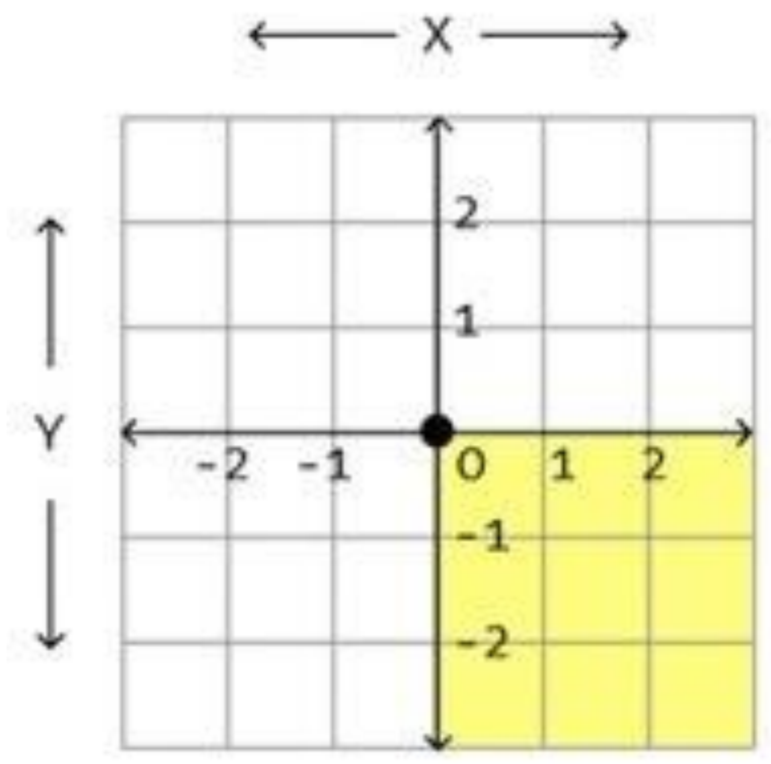
```
println("Hello, World!");
```

```
Hello, World!
```

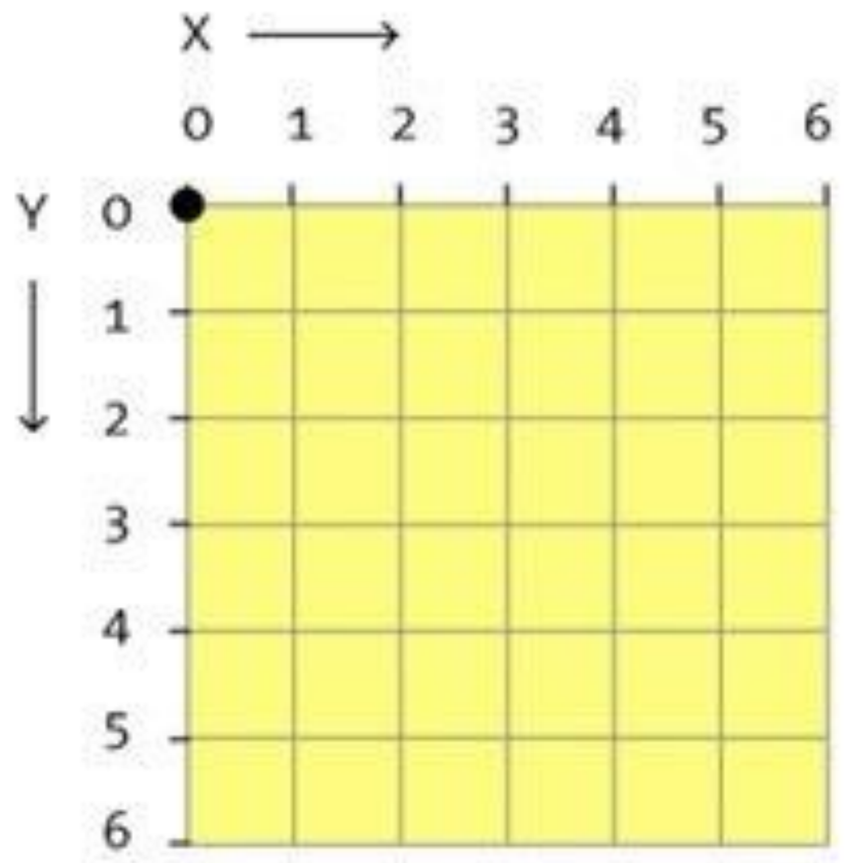
GRAPHICS

MONITORS

grid of pixels



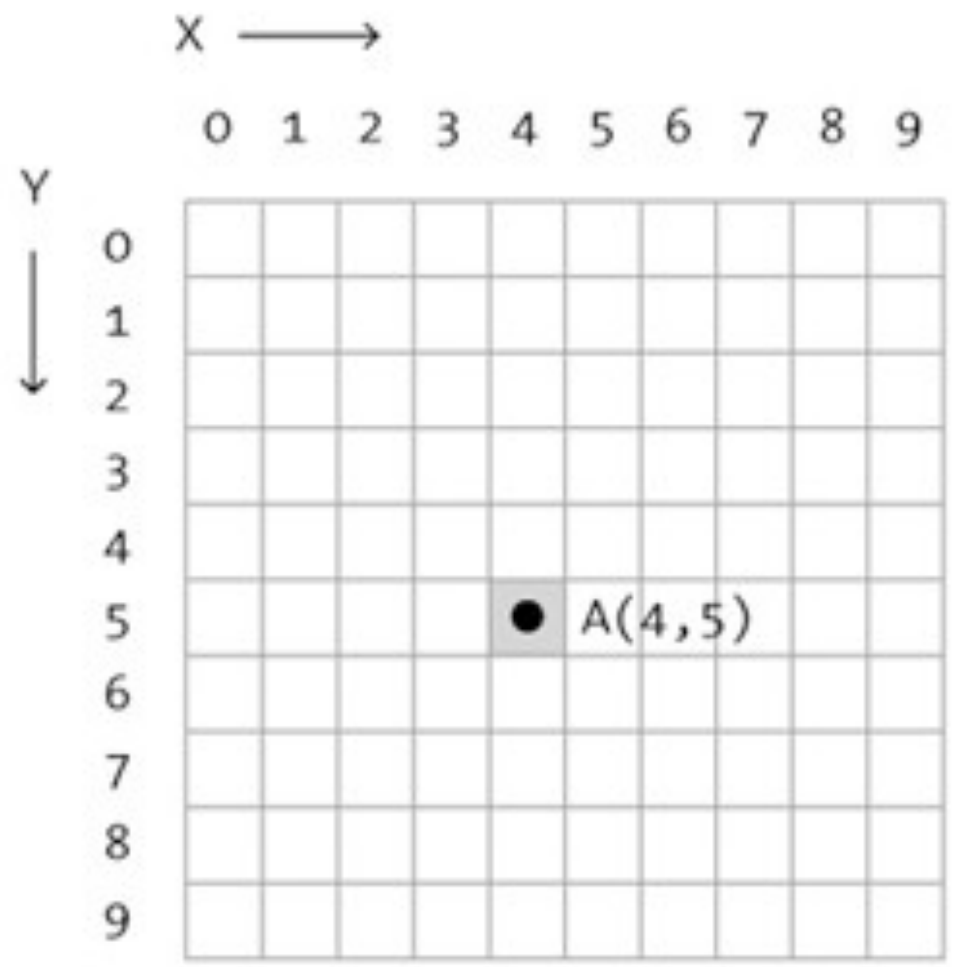
Eighth Grade



Computer

SHAPE

```
point(x, y);
```



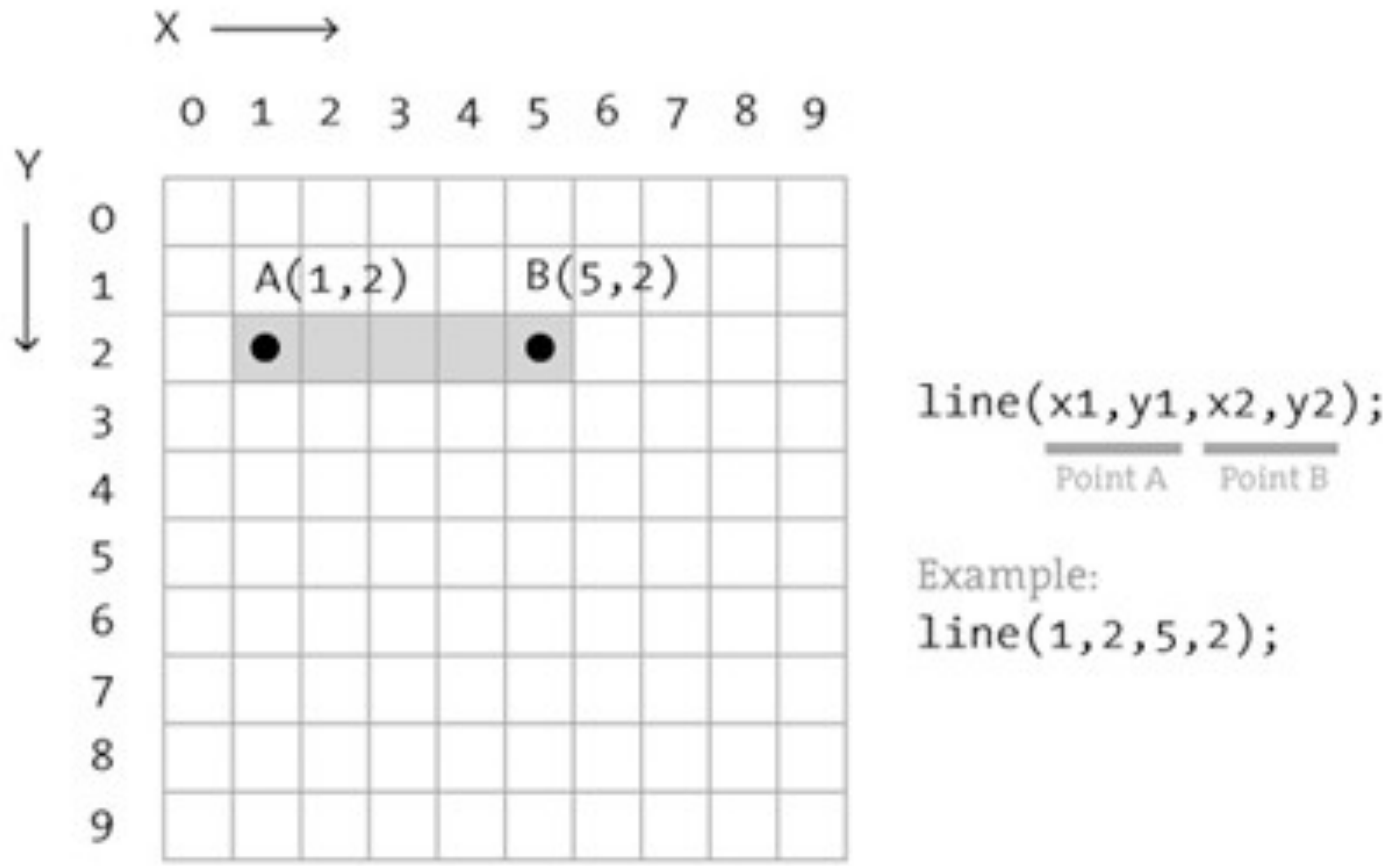
```
point(x,y);
```

Example:

```
A(4,5);
```

SHAPE

```
line(x1, y1, x2, y2);
```



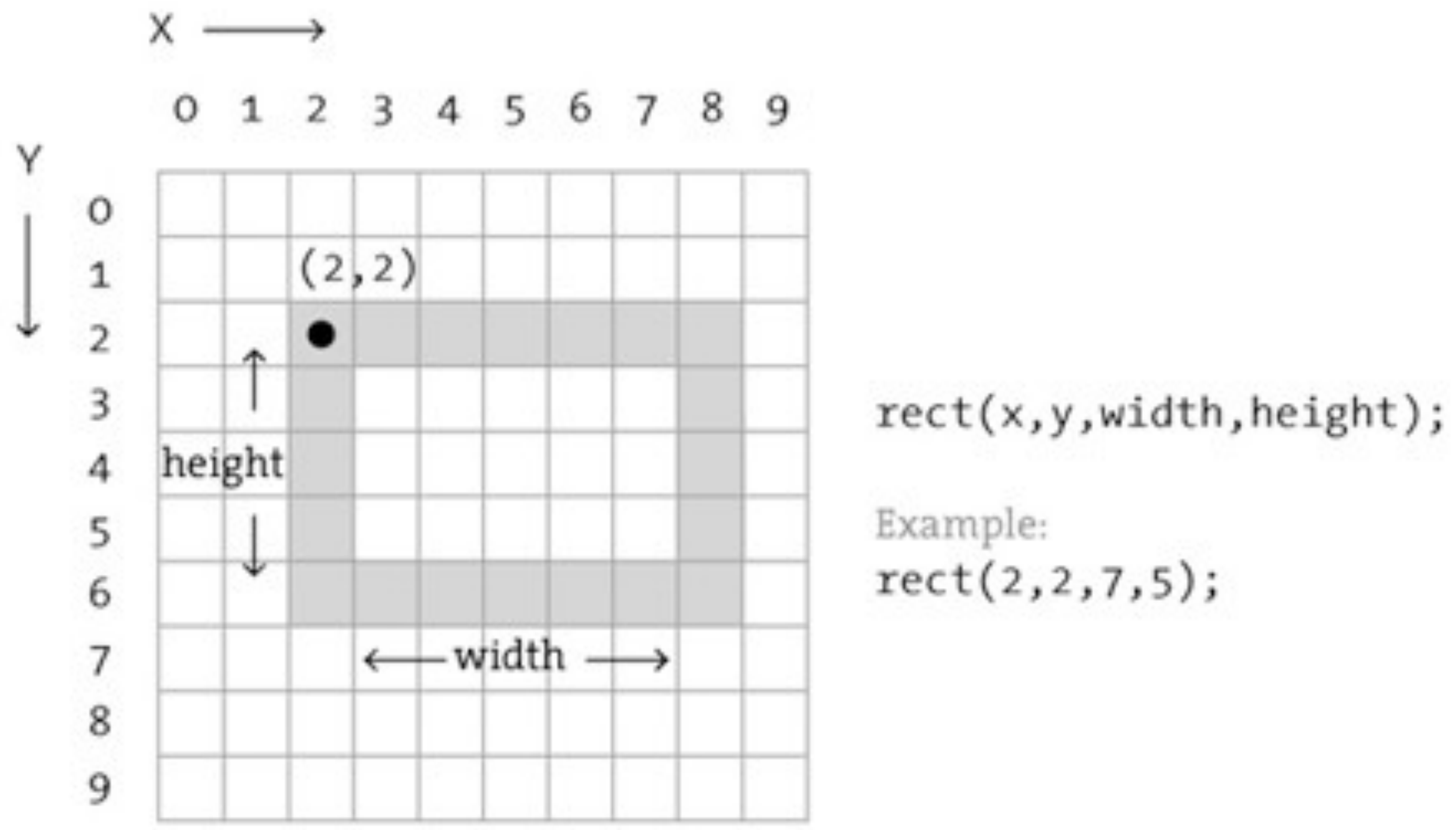
```
line(x1,y1,x2,y2);
```

Example:

```
line(1,2,5,2);
```

SHAPE

```
rect(x, y, width, height);
```



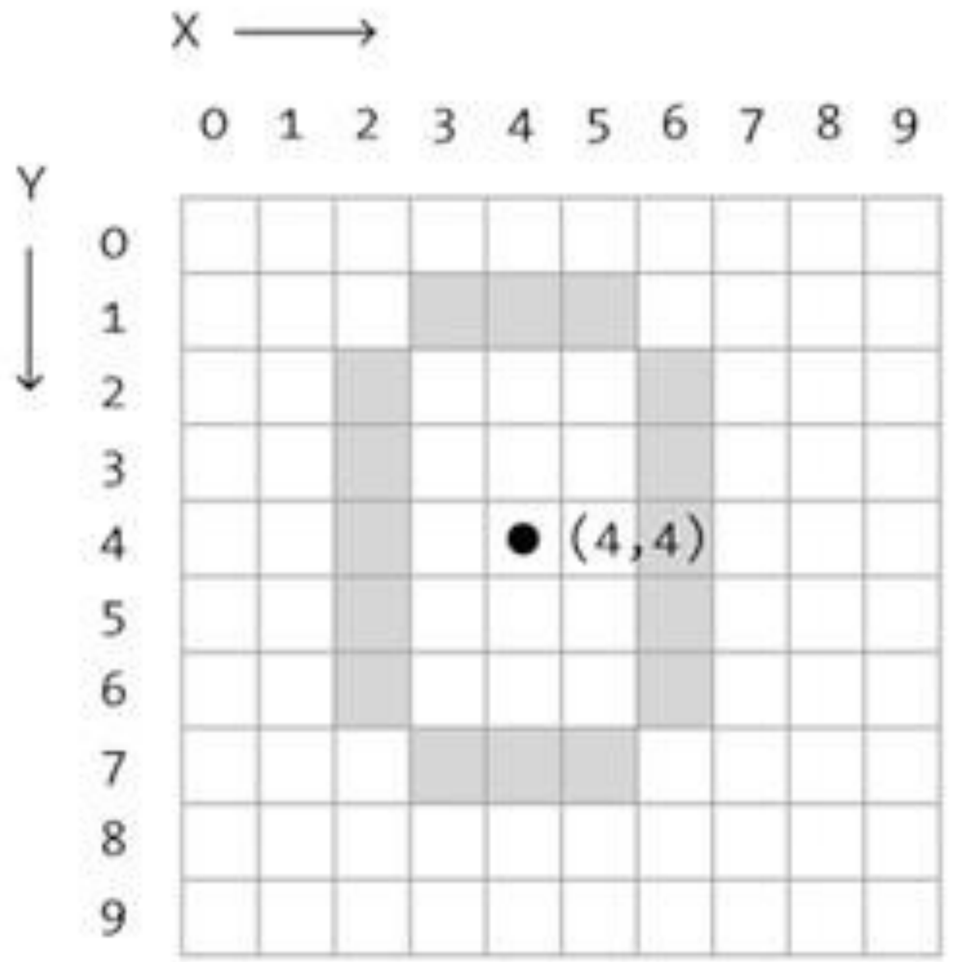
```
rect(x,y,width,height);
```

Example:

```
rect(2,2,7,5);
```

Shape

```
ellipseMode(CENTER);
ellipse(x, y, width, height);
```



```
ellipseMode(CENTER);
ellipse(x,y,width,height);
```

Example:

```
ellipseMode(CENTER);
ellipse(4,4,5,7);
```

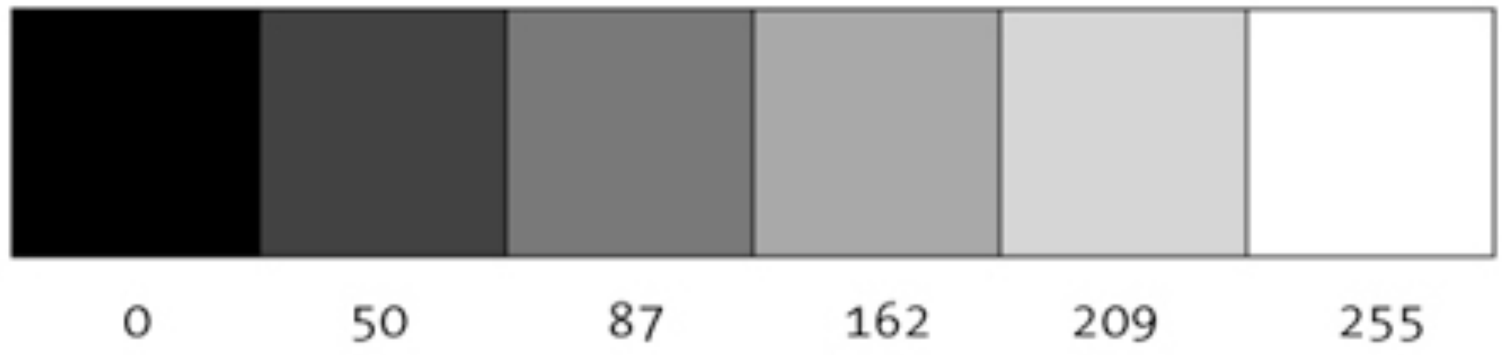
SHAPE

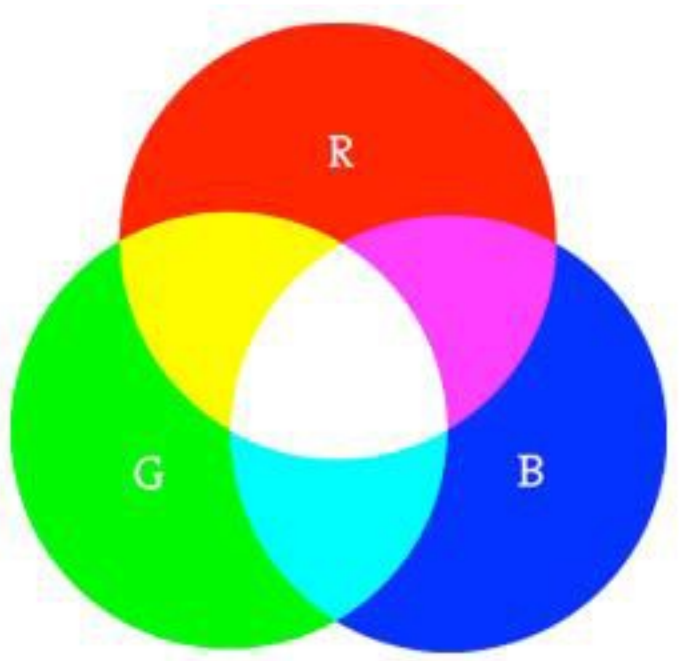
```
triangle(x1, y1, x2, y2, x3, y3);

quad(x1, y1, x2, y2, x3, y3, x4, y4);

arc(x, y, width, height, start, stop);
```

Color

luminance

```
background(255);
```





COLOR

RGB (default)

```
color c1 = color(r, g, b);
color c2 = #RRGGBB;
```

COLOR

RGBA

a = alpha / transparency / opacity

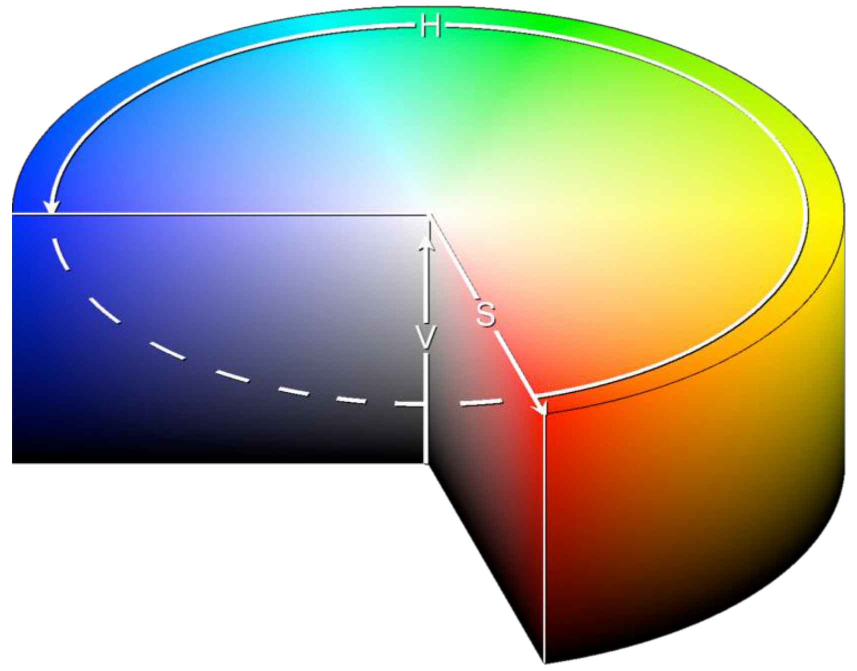
0 = transparent; 255 = opaque (solid)

```
color c1 = color(r, g, b, a);
```

COLOR MODES

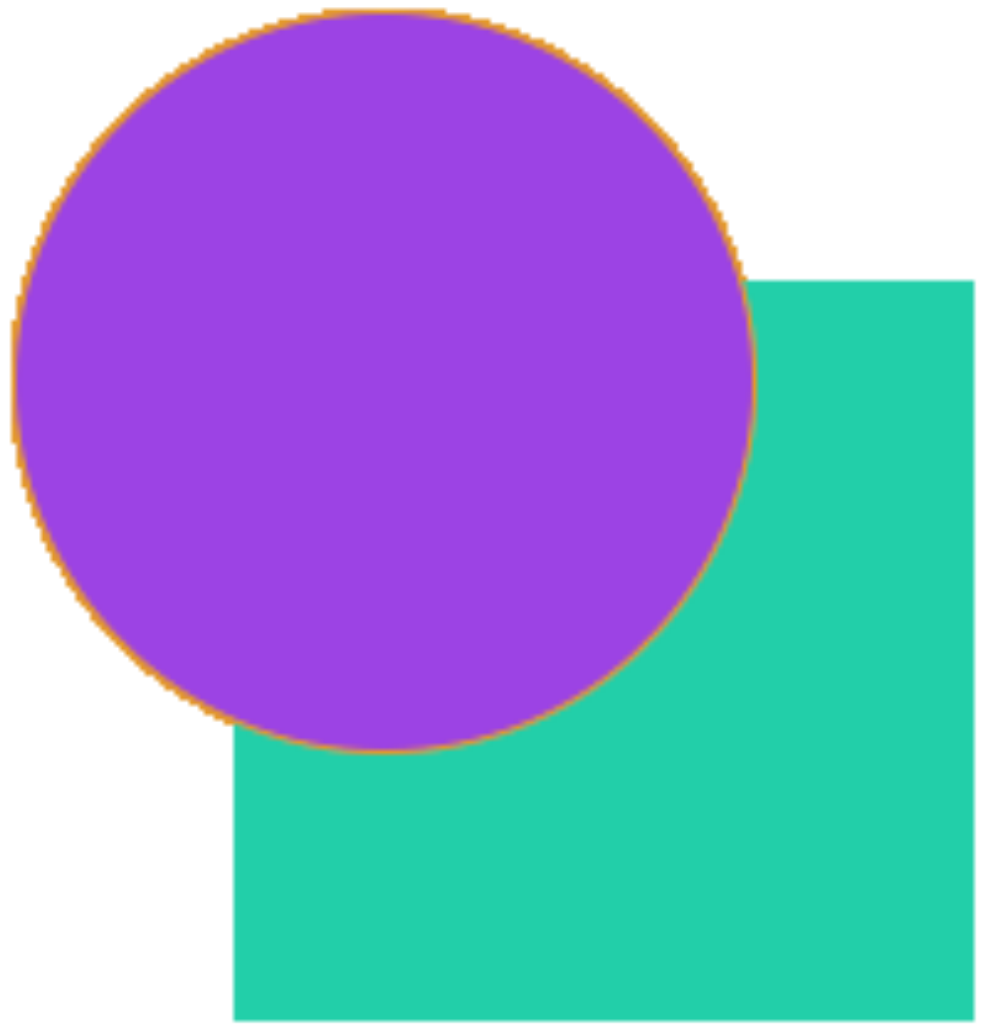
custom range: `colorMode(RGB, 100);`

HSB: `colorMode(HSB);`



PROPERTIES

```
noStroke();
fill(c1);
rect(...);

fill(c2);
stroke(c3);
ellipse(...);
```



PROPERTIES

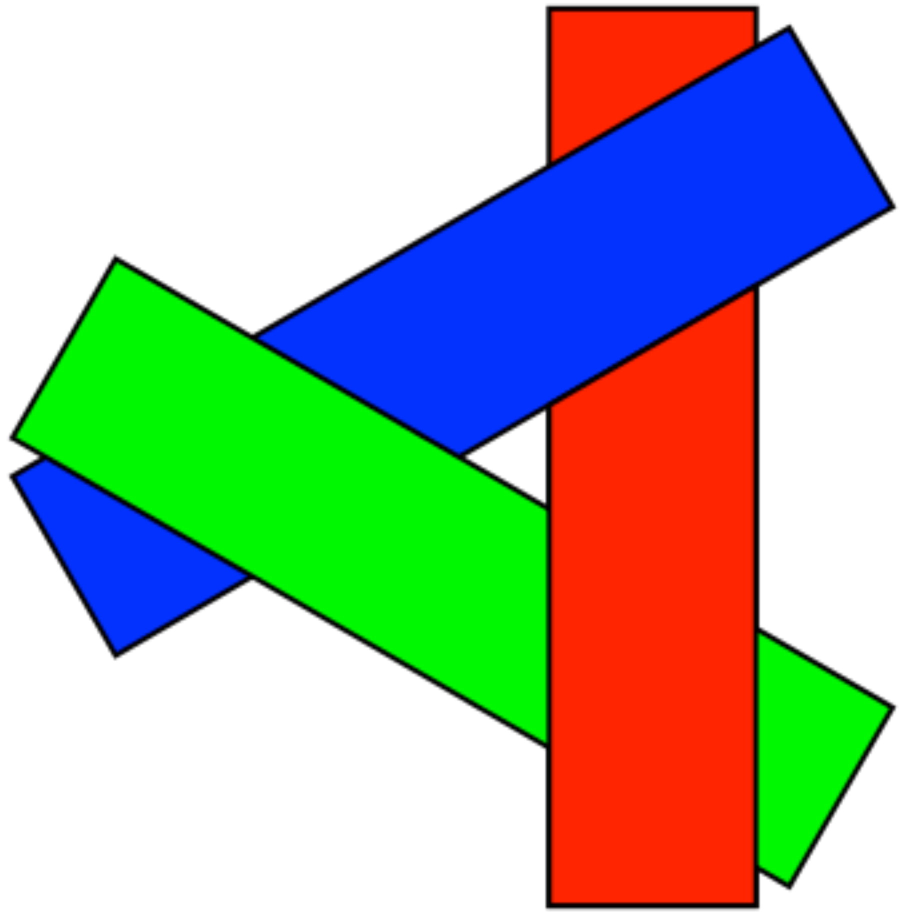
```
noFill();
noStroke();
ellipse(...);



rect(...);
```

ORDER

shapes are painted one at a time

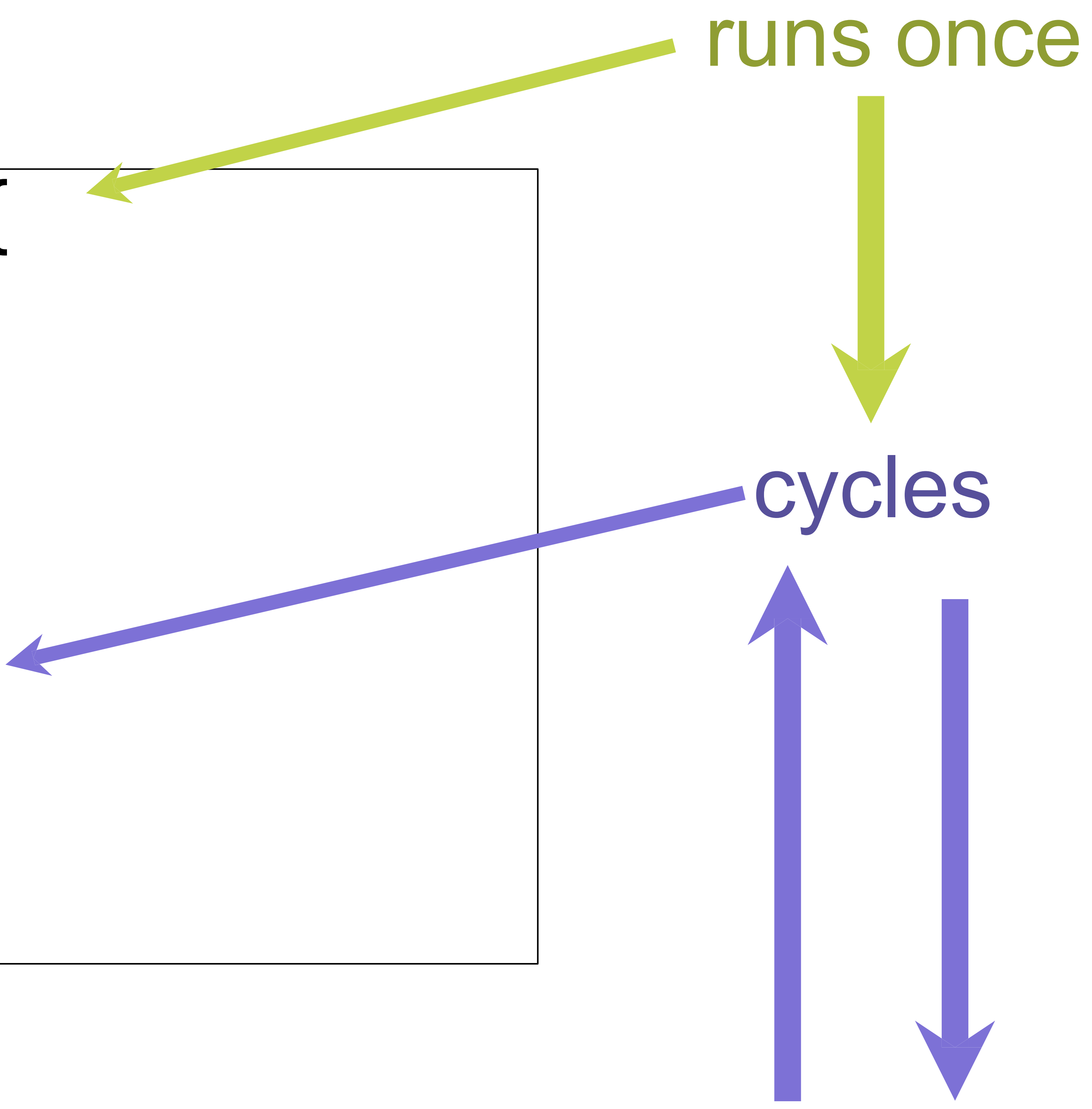
overlap can occur

some shapes are not supported



ANIMATION

```
void setup(){
    ...
}


void draw(){
    ...
}
```



TEXT

```
// in setup()
PFont myFont;
myFont = createFont("Georgia", 32);


// in draw()
textFont(myFont);
textAlign(CENTER, CENTER);
text("Hello, World!", width/2, height/2);
```

PROGRAMMING

INTERACTION: MOUSE

```
void mouseClicked(){

  if(mouseButton == LEFT)
    fill(0);

  else if(mouseButton == RIGHT)
    fill(255);

  else
    fill(126);
}
```

```
void mousePressed()

void mouseReleased()

void mouseClicked()

void mouseDragged()

void mouseMoved()

void mouseWheel()
```

```
mouseX
mouseY
pmouseX
pmouseY
```

Interaction: Keyboard

```
void keyTyped(){

  if(key == 'b')
    fill(0);

  else if(key == 'w')
    fill(255);

  else
    fill(126);
}
```

```
void keyPressed()

void keyReleased()

void keyTyped()
```

```
keyPressed
key
keyCode
```

Structure

comments, variables, arrays, loops

`ArrayList` (also `FloatList`, `IntList`, `StringList`)

`HashMap` (dict: also `FloatDict`, `IntDict`, `StringDict`)

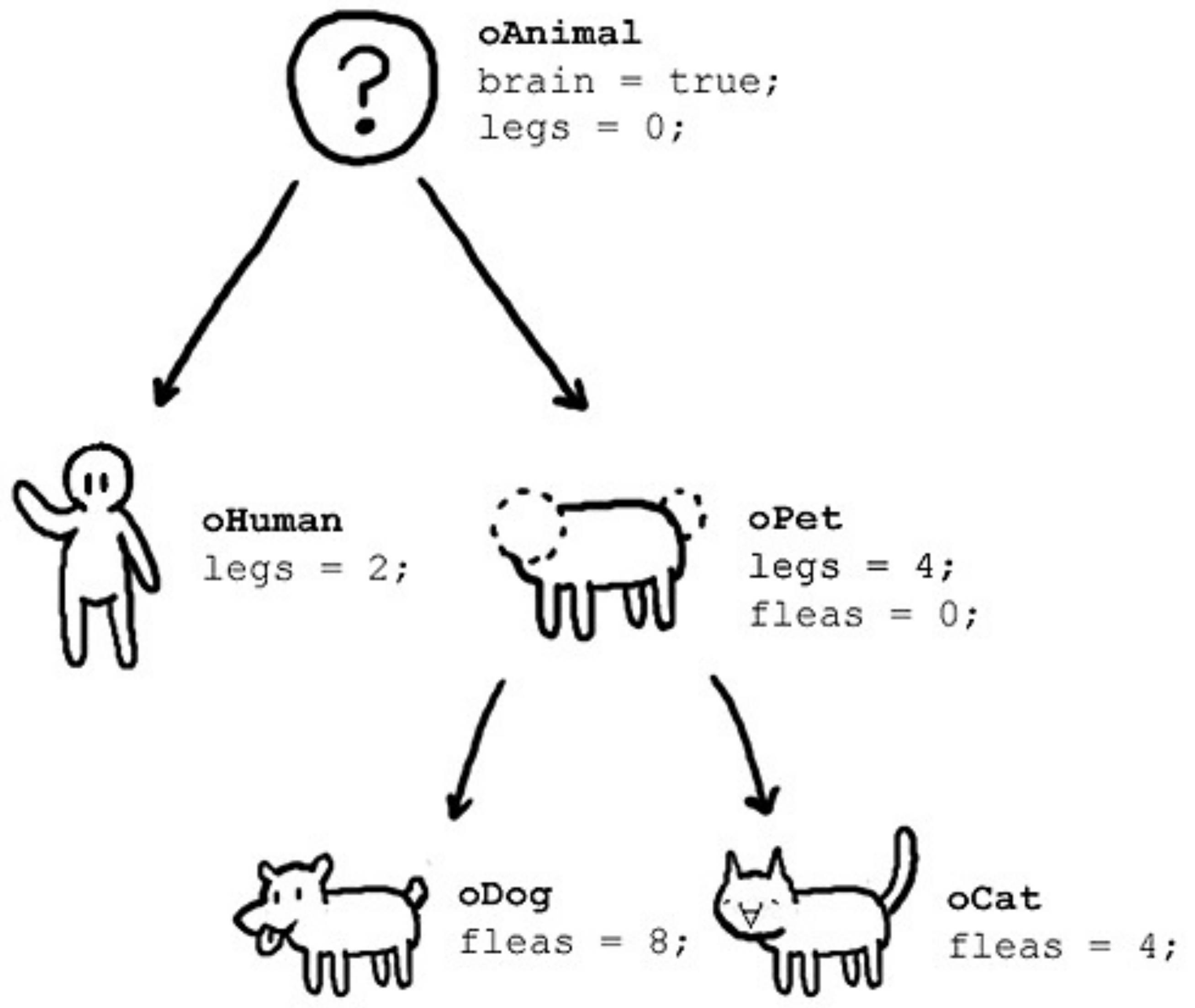
`Table`, `XML`, `JSON`

(and anything else Java!)

OBJECT-ORIENTED

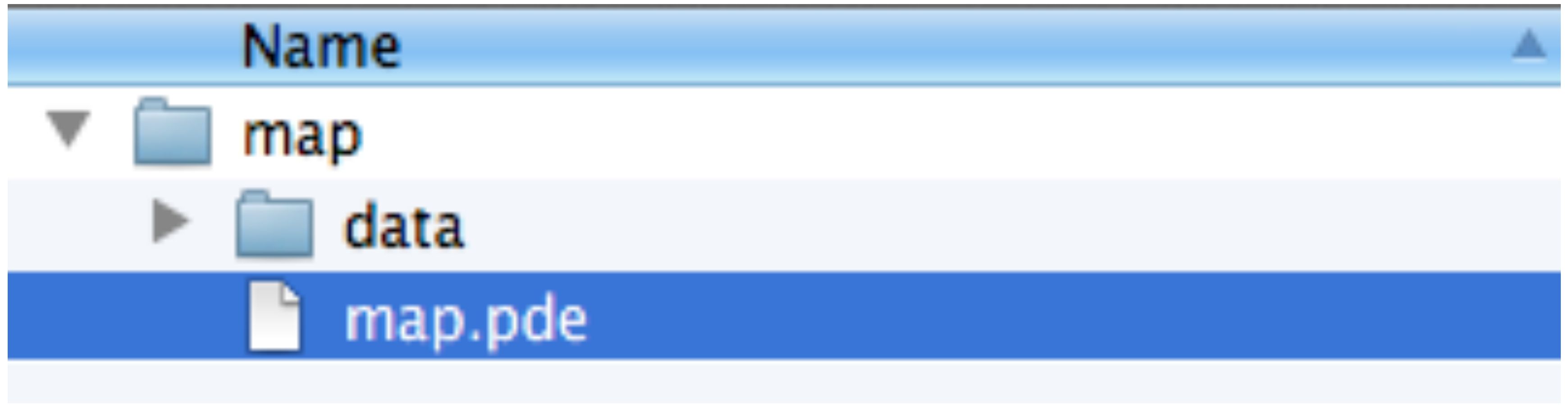
with classes

```
class oAnimal{
  boolean brain;
  int legs;

  oAnimal(){
    brain = true;
    legs = 0;
  }
}
```



FOLDER STRUCTURE

folder [NAME] & [NAME].pde must match



optional data folder (for images, input)

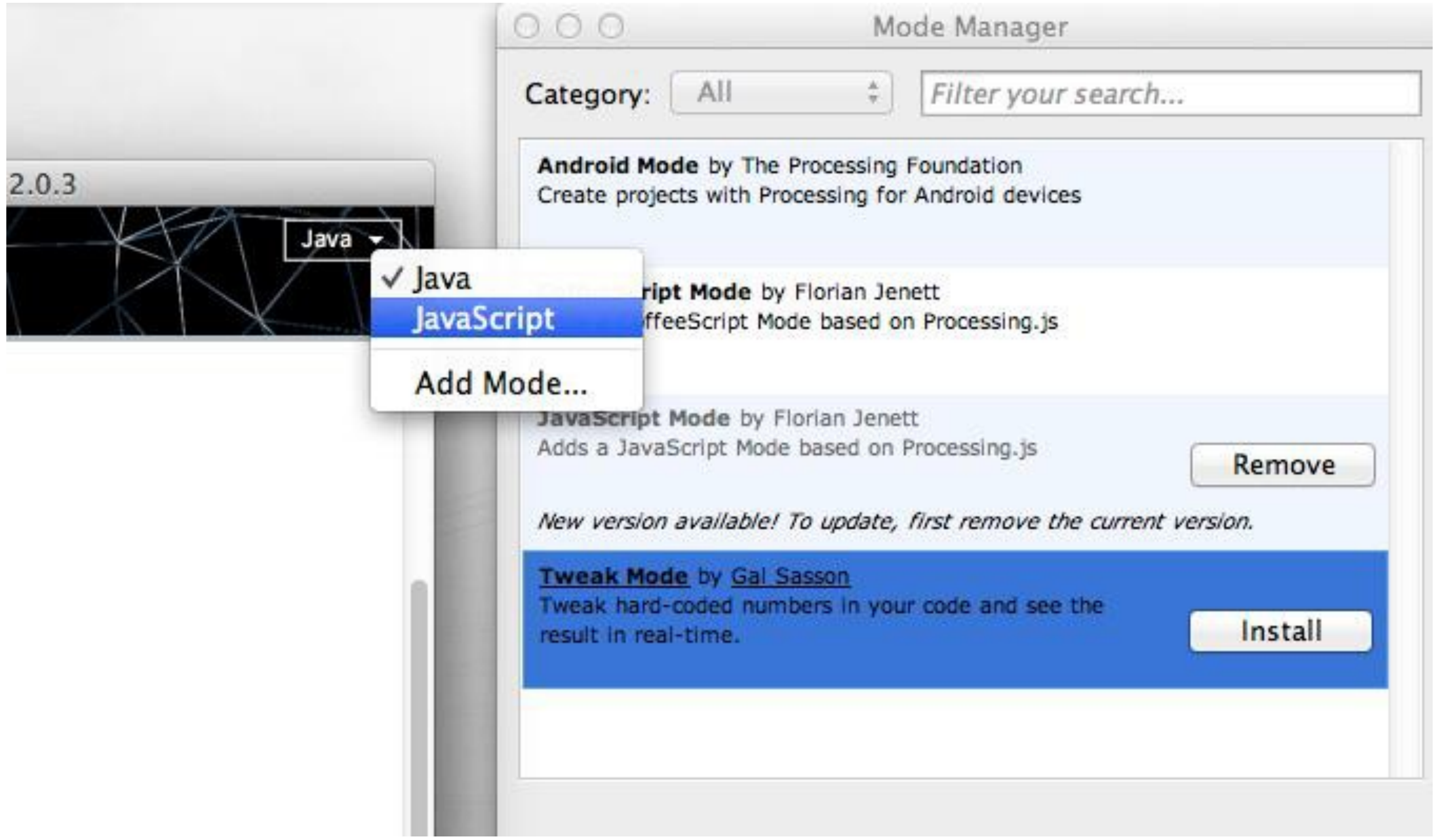
MODES

Java (default)

JavaScript

Android

etc.



Android Mode by The Processing Foundation
Create projects with Processing for Android devices

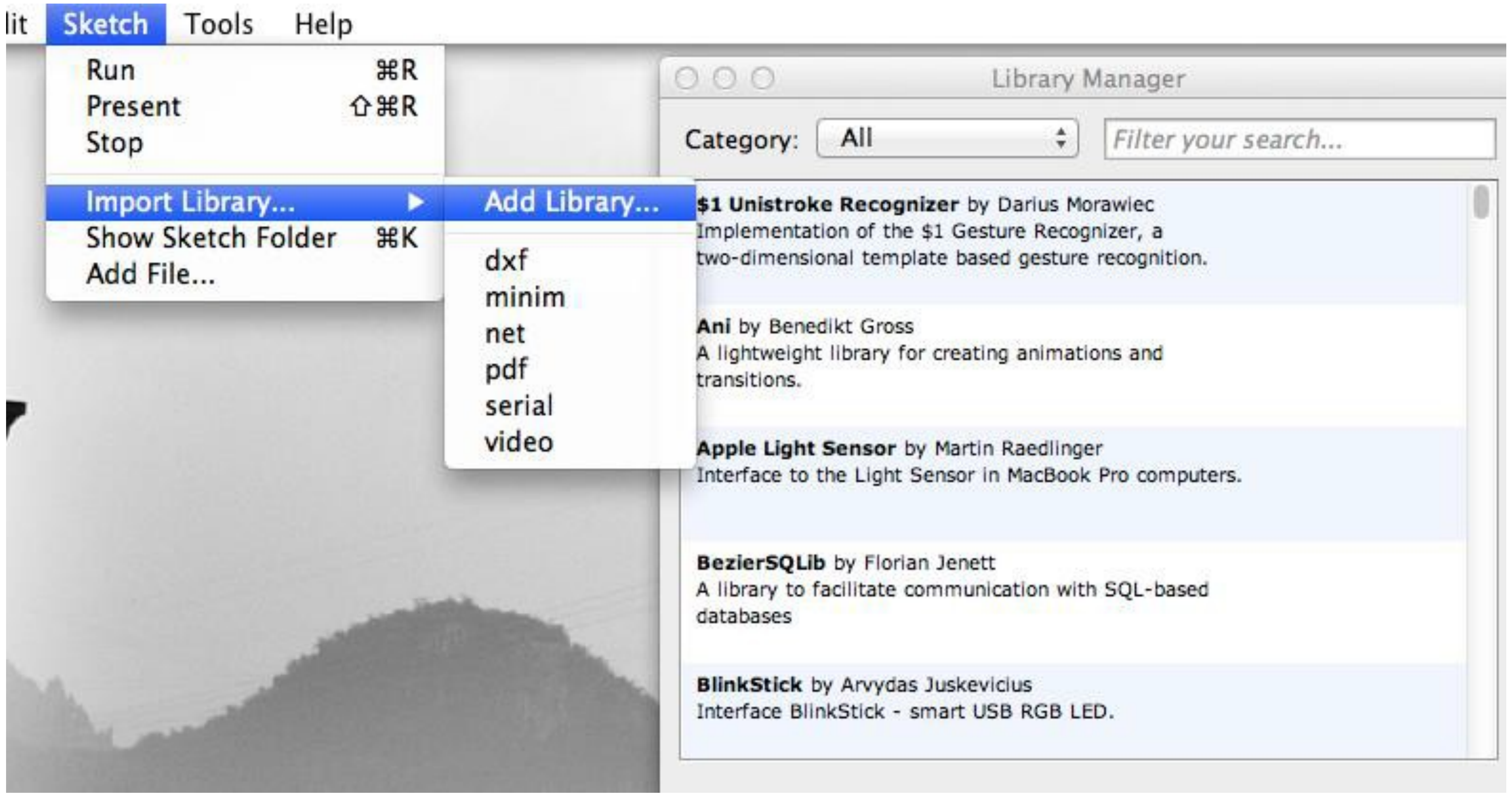
ript Mode by Florian Jenett
ffeeScript Mode based on Processing.js

JavaScript Mode by Florian Jenett
Adds a JavaScript Mode based on Processing.js

New version available! To update, first remove the current version.

Tweak Mode by Gal Sasson
Tweak hard-coded numbers in your code and see the result in real-time.

LIBRARIES



$1 Unistroke Recognizer by Darius Morawiec
Implementation of the $1 Gesture Recognizer, a two-dimensional template based gesture recognition.

Ani by Benedikt Gross
A lightweight library for creating animations and transitions.

Apple Light Sensor by Martin Raedlinger
Interface to the Light Sensor in MacBook Pro computers.

BezierSQLib by Florian Jenett
A library to facilitate communication with SQL-based databases

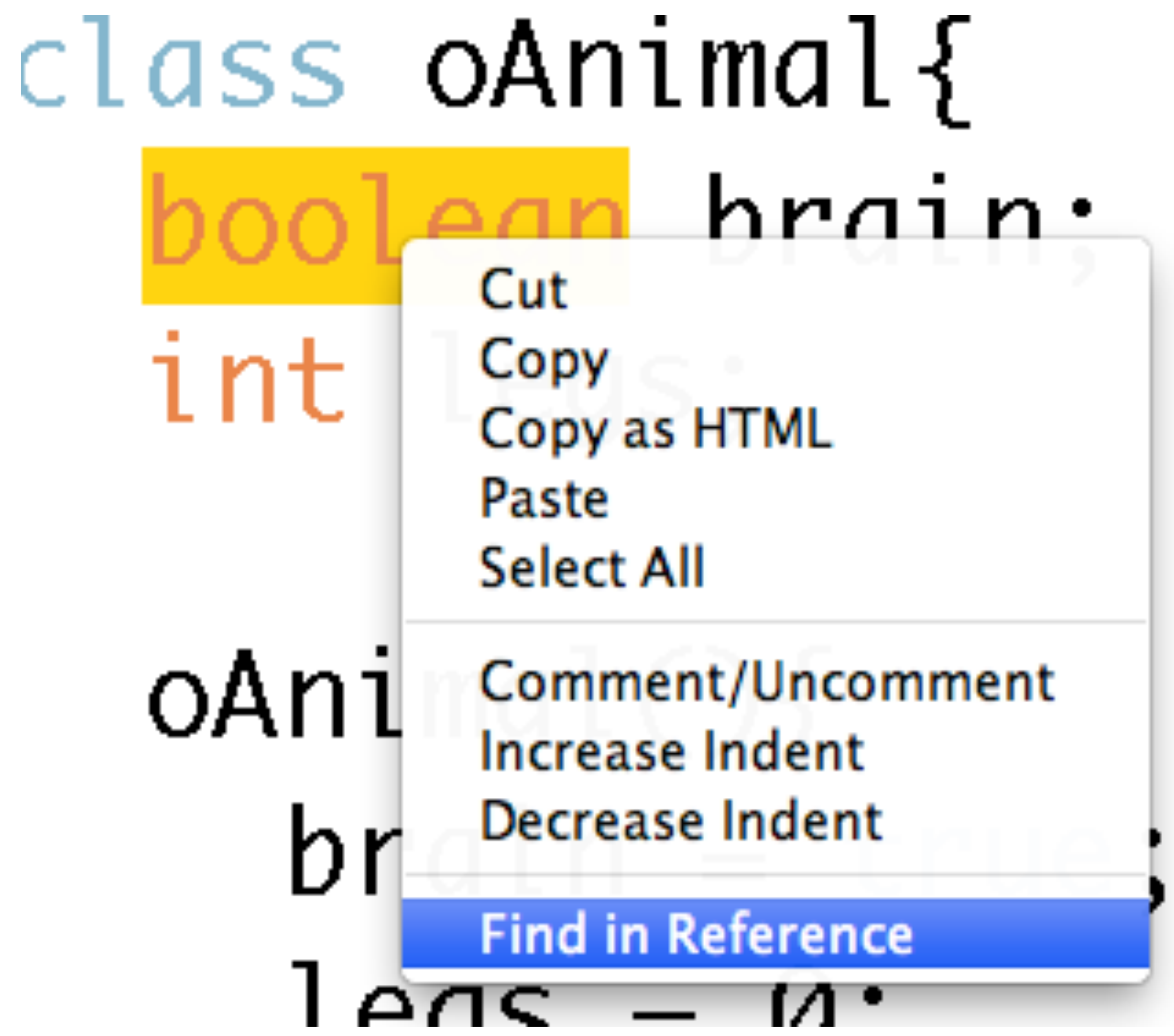
BlinkStick by Arvydas Juskevicius
Interface BlinkStick - smart USB RGB LED.

DOCUMENTATION

available online

also in the PDE

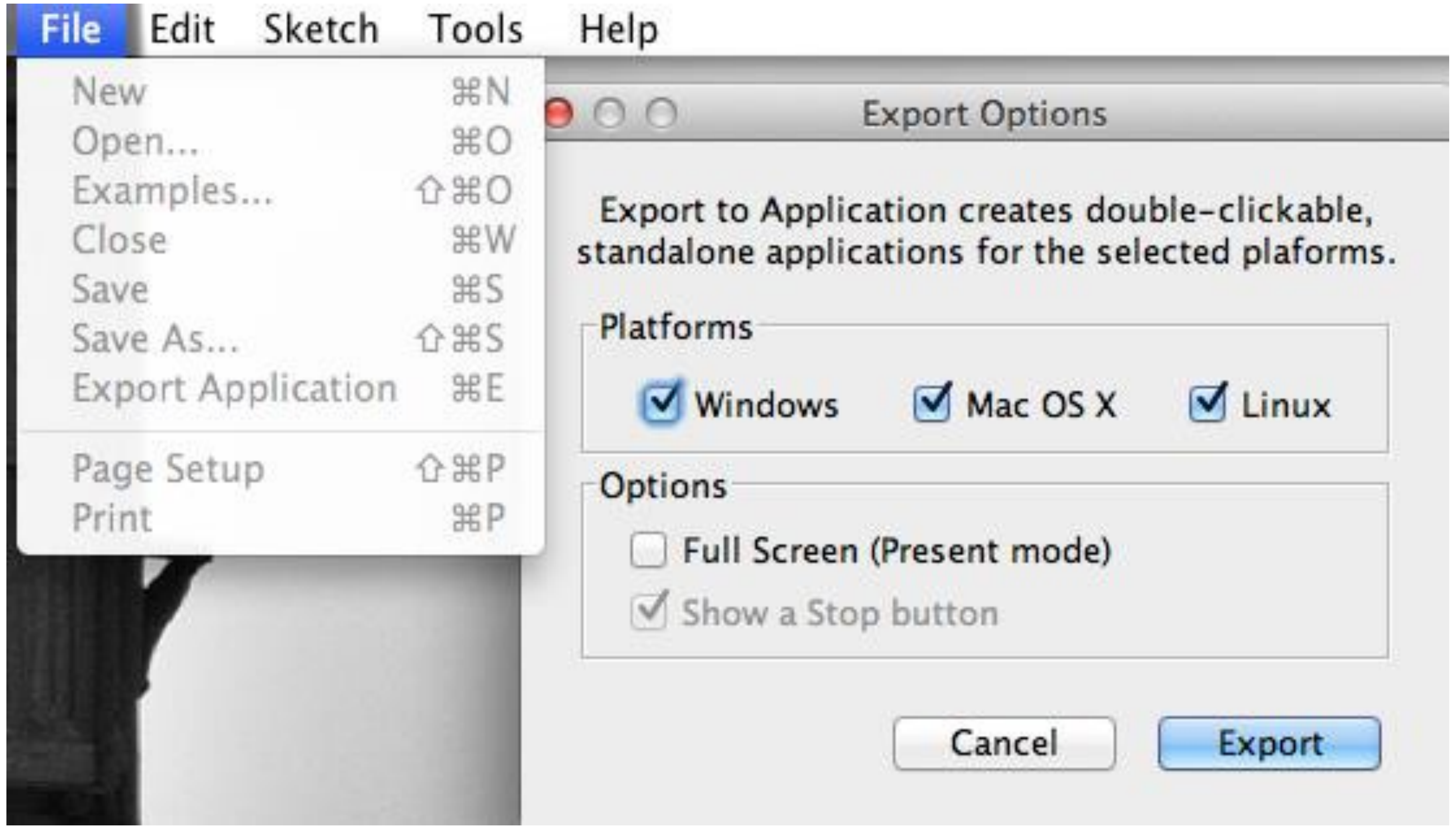
http://processing.org/reference/



```
class oAnimal{
  boolean brain;
  int
  
  oAni
    br
    legs = 4;
```

Exporting

creating applications is simple



Export to Application creates double-clickable, standalone applications for the selected plaforms.

Platforms

[✓] Windows [✓] Mac OS X [✓] Linux

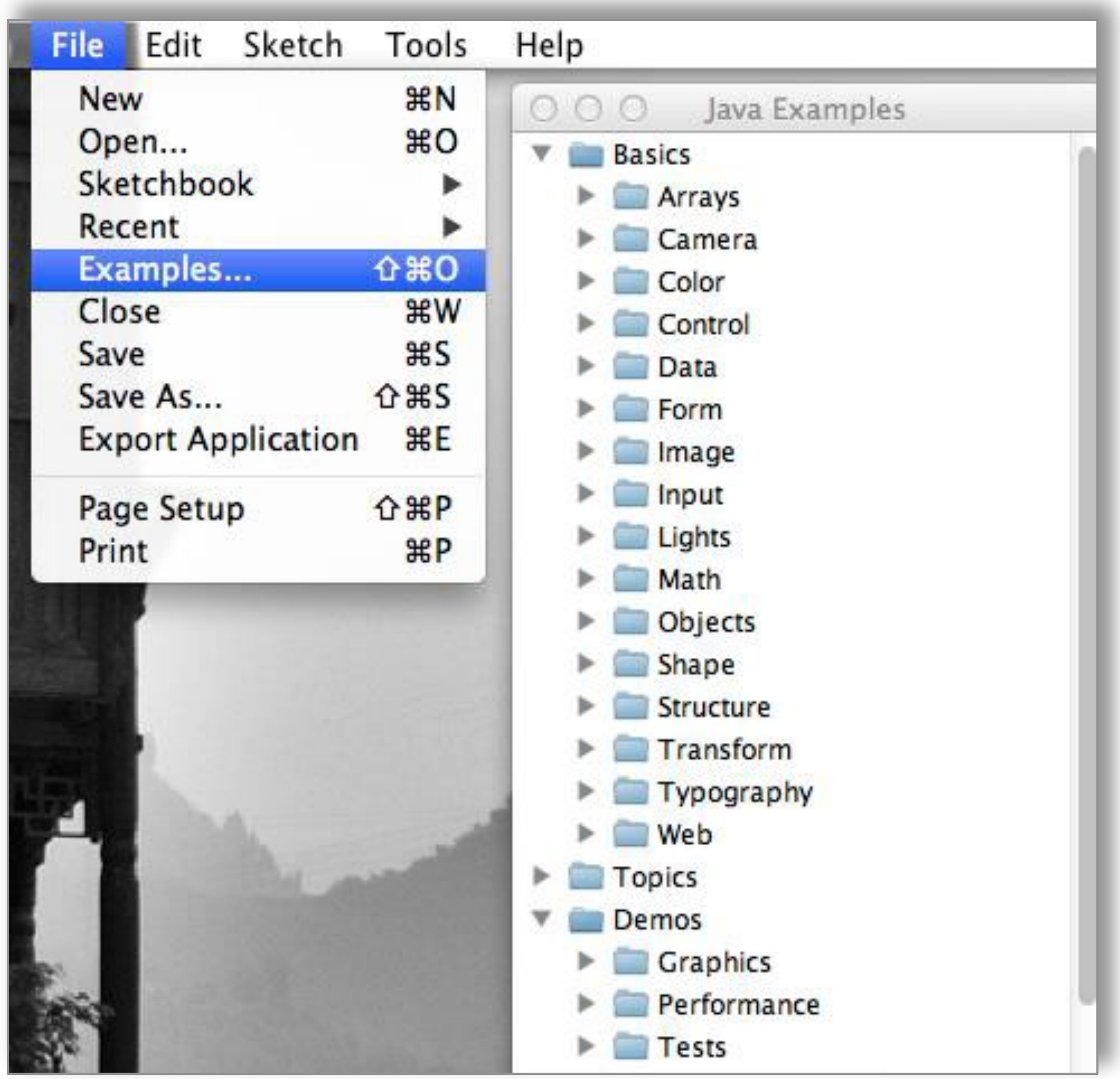
Options

[ ] Full Screen (Present mode)

[✓] Show a Stop button

Examples

variety of samples



Demo