

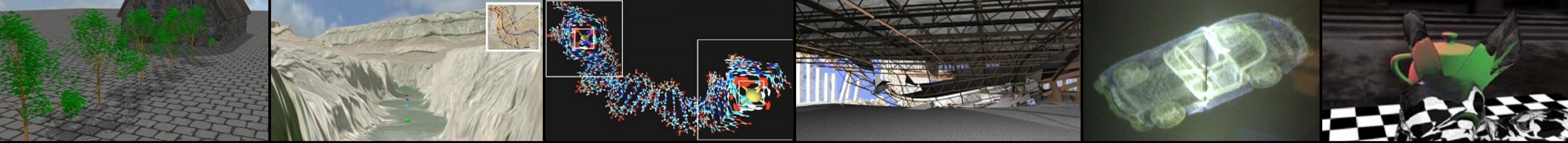
COT 452 I: INTRODUCTION TO COMPUTATIONAL GEOMETRY



INTRODUCTION TO PROCESSING

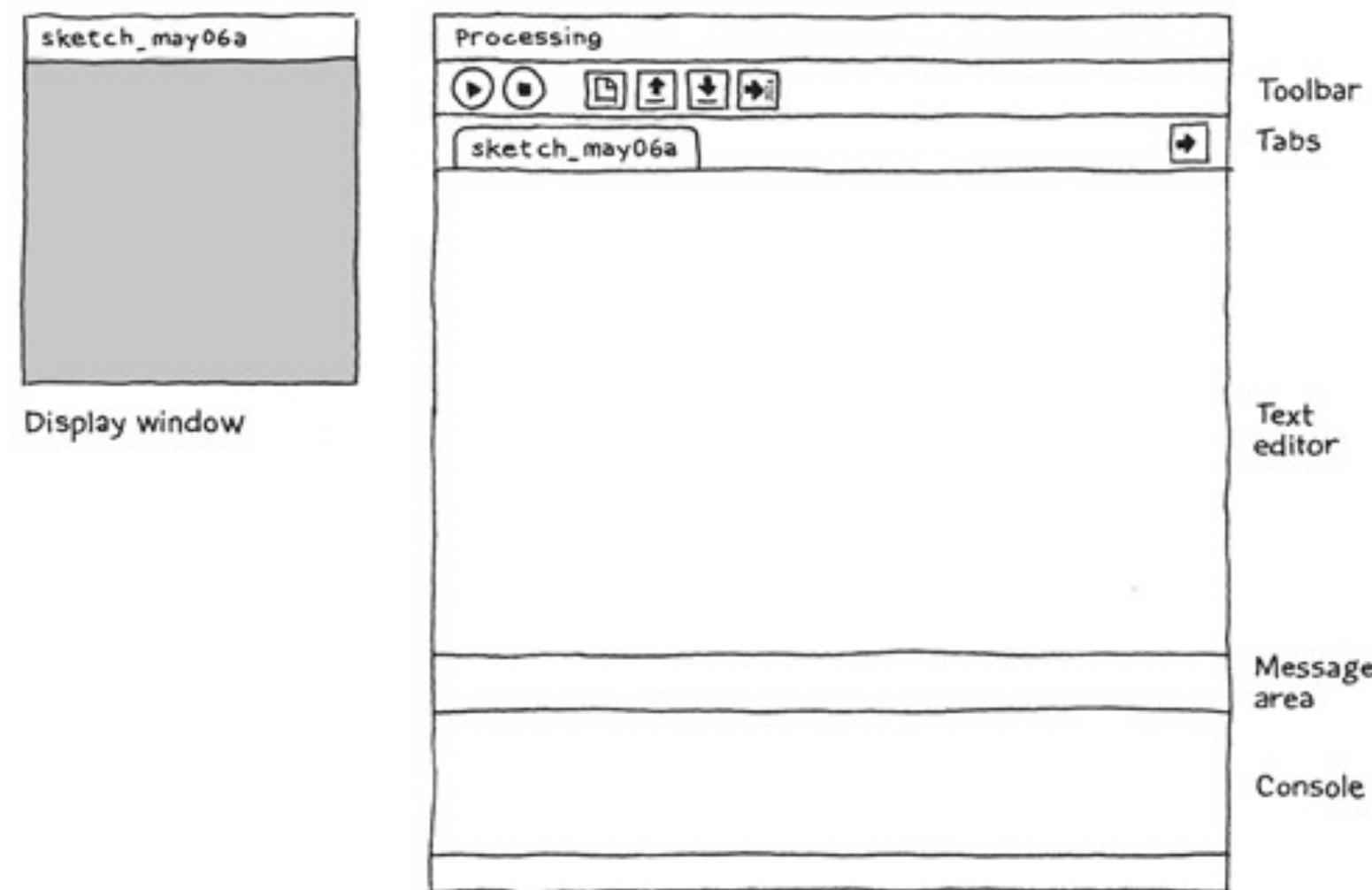
Paul Rosen
Assistant Professor
University of South Florida

Some slides from Hitesh Raju



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	Shape	Color
() (parentheses)	createShape()	Setting
, (comma)	loadShape()	background()
. (dot)	PShape	clear()
/* */ (multiline comment)		colorMode()
/** */ (doc comment)	2D Primitives	fill()
// (comment)	arc()	noFill()
; (semicolon)	ellipse()	noStroke()
= (assign)	line()	stroke()
[] (array access)	point()	
{ } (curly braces)	quad()	Creating & Reading
catch	rect()	alpha()
class	triangle()	blue()
draw()		brightness()
exit()	Curves	color()
extends	bezier()	green()
false	bezierDetail()	hue()
final	bezierPoint()	lerpColor()
implements	bezierTangent()	red()
import	curve()	saturation()
loop()	curveDetail()	
new	curvePoint()	Image
noLoop()	curveTangent()	createImage()
null	curveTightness()	PI mage
popStyle()		
private	3D Primitives	
public	box()	

WHAT IS IT?

open-source, online community

<https://processing.org/>

<http://forum.processing.org/>

<https://github.com/processing>

WHY NOT OTHER LANGUGES?

difficulty to sketch with other languages

complicated setup

portability challenging

not easy to learn

WHY PROCESSING?

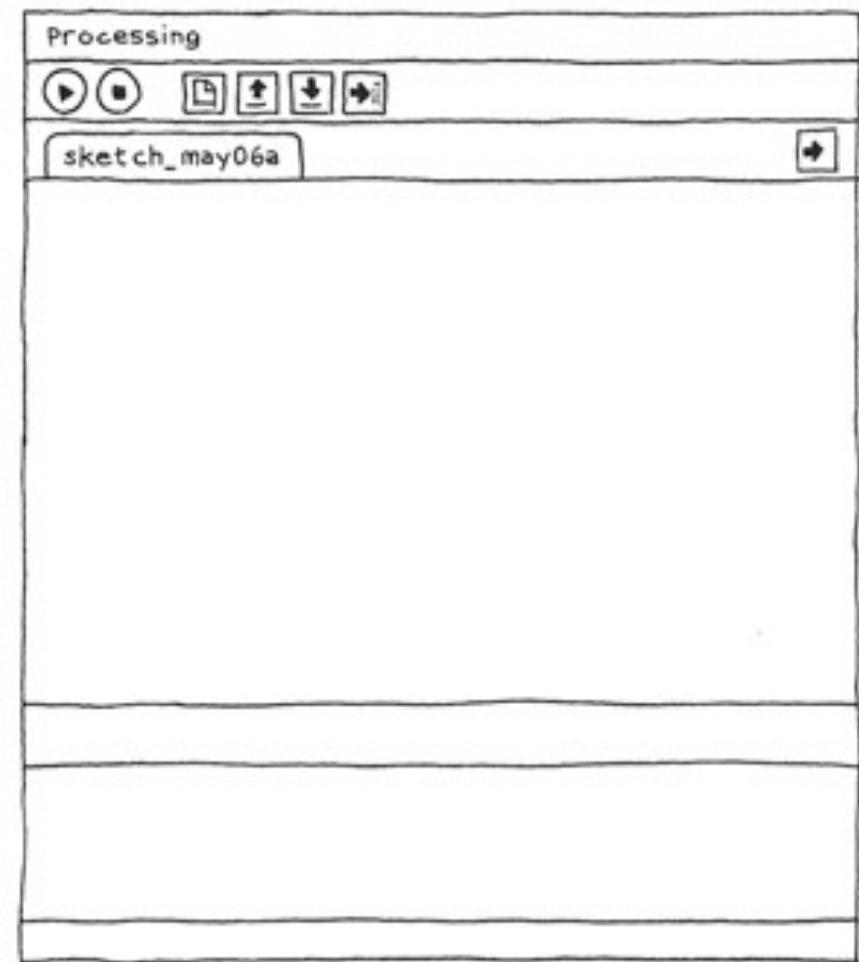
- +/- Java-based
 - complexity
 - + big standard library
 - + similar syntax & portability
- + lots of user-contributed libraries
 - + fast to startup

WHY PROCESSING?

program = sketch



Display window



Toolbar
Tabs

Text
editor

Message
area

Console

WHY PROCESSING?

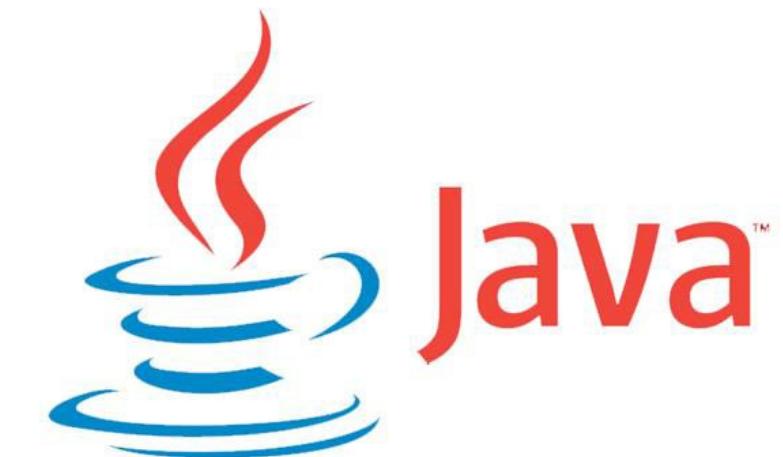
programming syntax

```
void setup() {  
    size(480, 120);  
}  
  
void draw() {  
    if (mousePressed) {  
        fill(0);  
    } else {  
        fill(255);  
    }  
    ellipse(mouseX, mouseY, 80, 80);  
}
```

WHY PROCESSING?

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

```
javac Hello.java
java Hello
```



WHY PROCESSING?

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



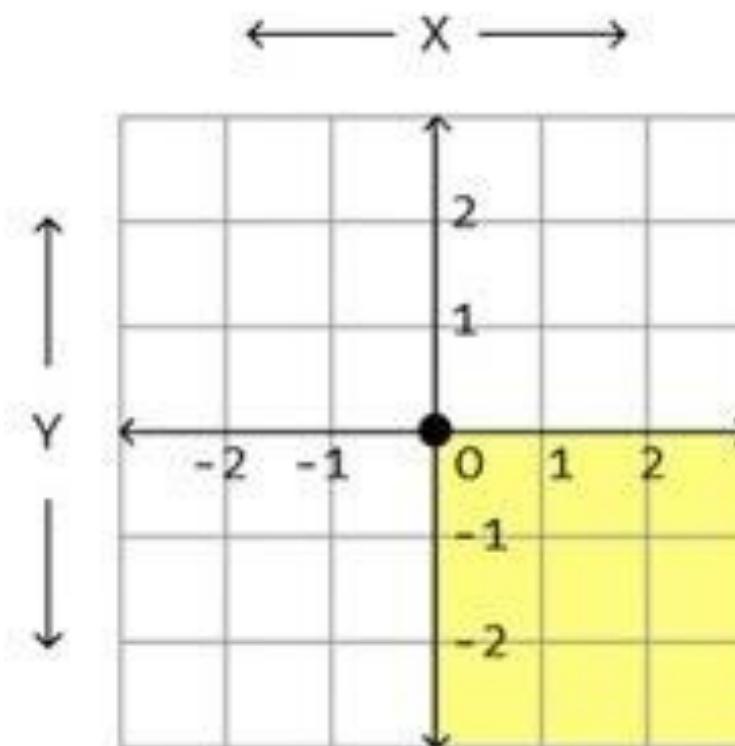
The image shows a screenshot of the Processing 2.2.1 IDE. The window title is "sketch_140912a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for play, stop, step, and run, along with a "Java" dropdown menu. The code editor window displays the line of code: `println("Hello, World!");`. Below the code editor is a preview window showing the text "Hello, World!". The bottom status bar indicates the line number "1".

GRAPHICS

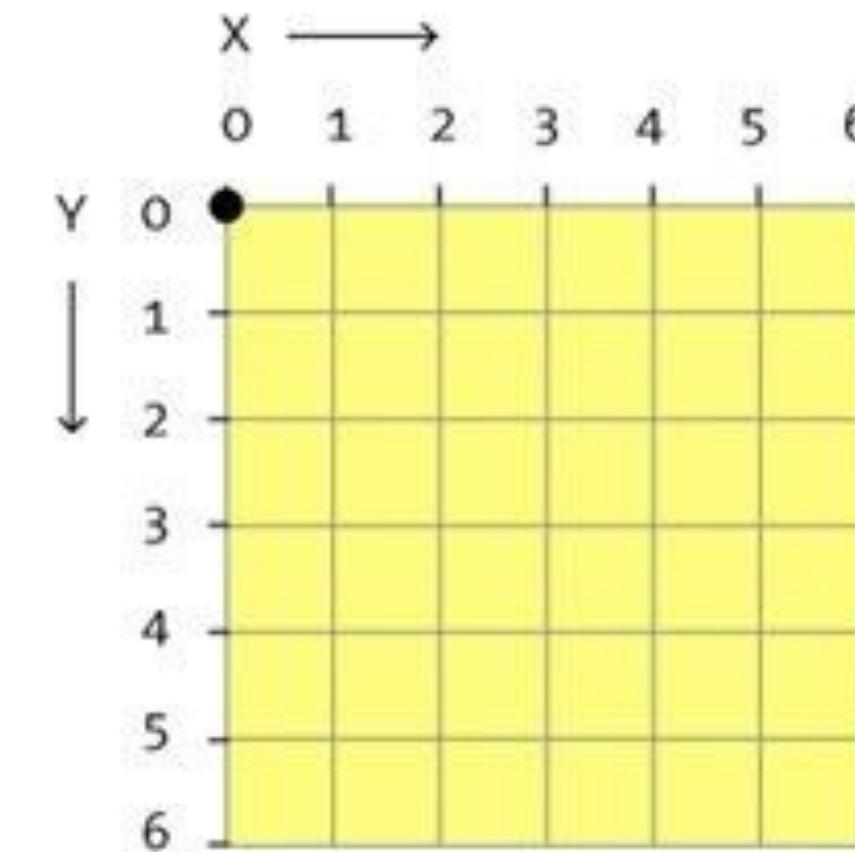


MONITORS

grid of pixels



Right Hand Coordinate System



Monitor Coordinate System
(Left Hand Coordinate System)

RIGHT HAND (RHC)

VS

LEFT HAND (LHC)

COORDINATE SYSTEMS

Everything we do in this class assumes RHC (not in processing)

I've included some code in the skeleton to convert Processing into a RHC



LHC System (Processing default)

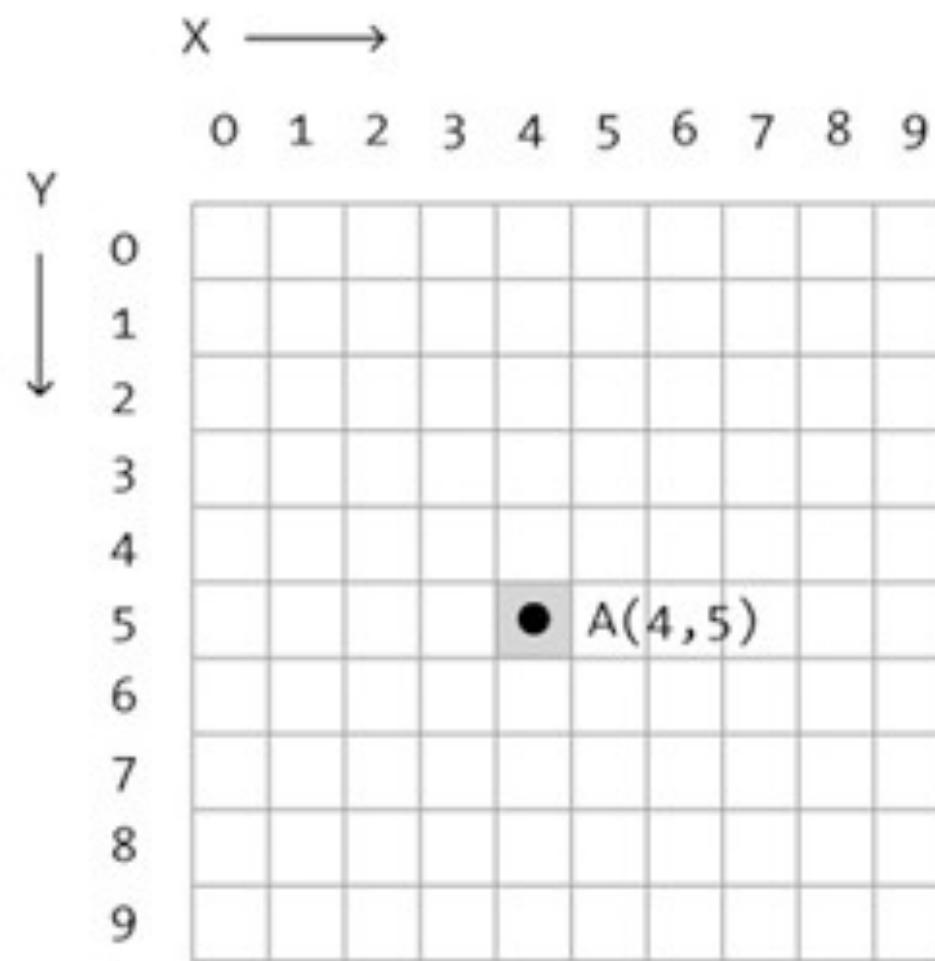


RHC System (provided modification)



SHAPE

point (x, y);

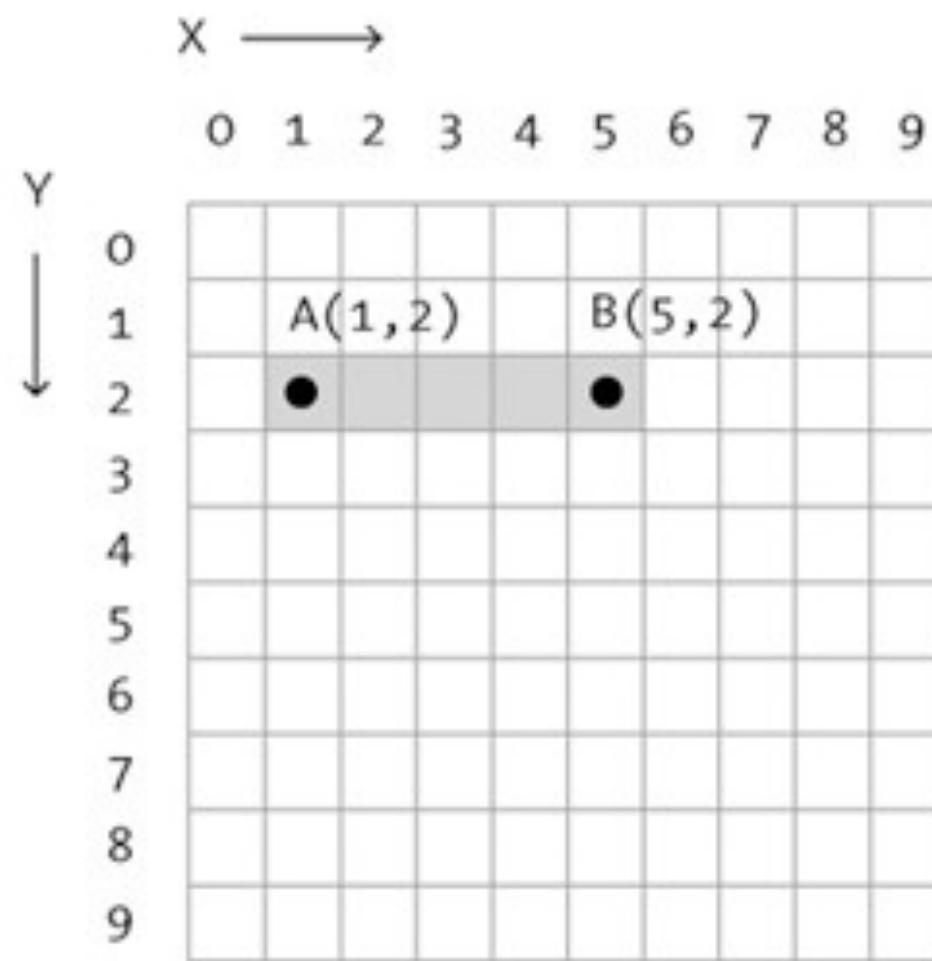


point(x,y);

Example:
A(4,5);

SHAPE

line(x1, y1, x2, y2);

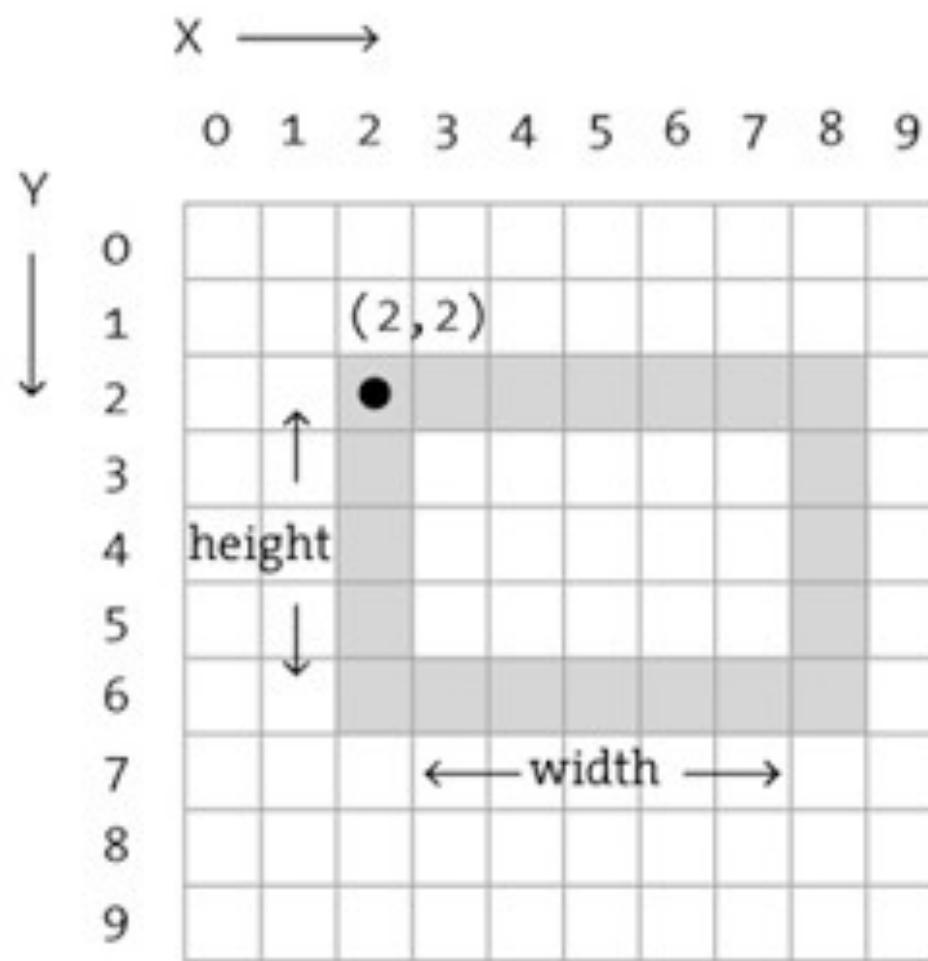


line(x1,y1,x2,y2);
 Point A Point B

Example:
line(1,2,5,2);

SHAPE

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



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

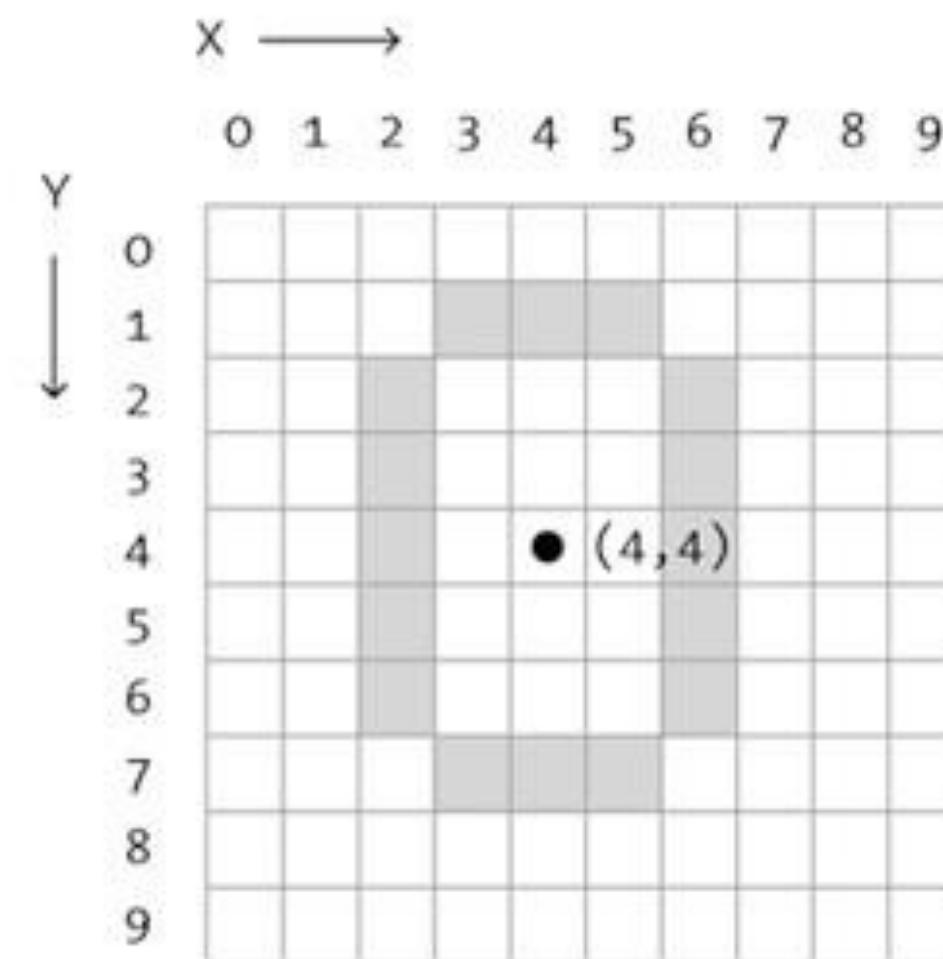
Example:

`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);

SHAPE

More complex shapes available with:

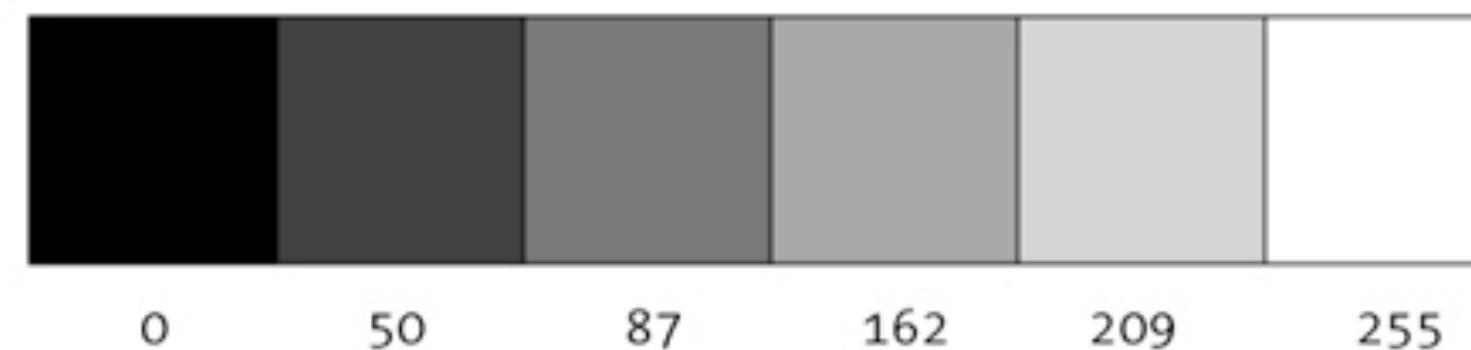
[beginShape\(\)](#) / [vertex\(\)](#) / [endShape\(\)](#)

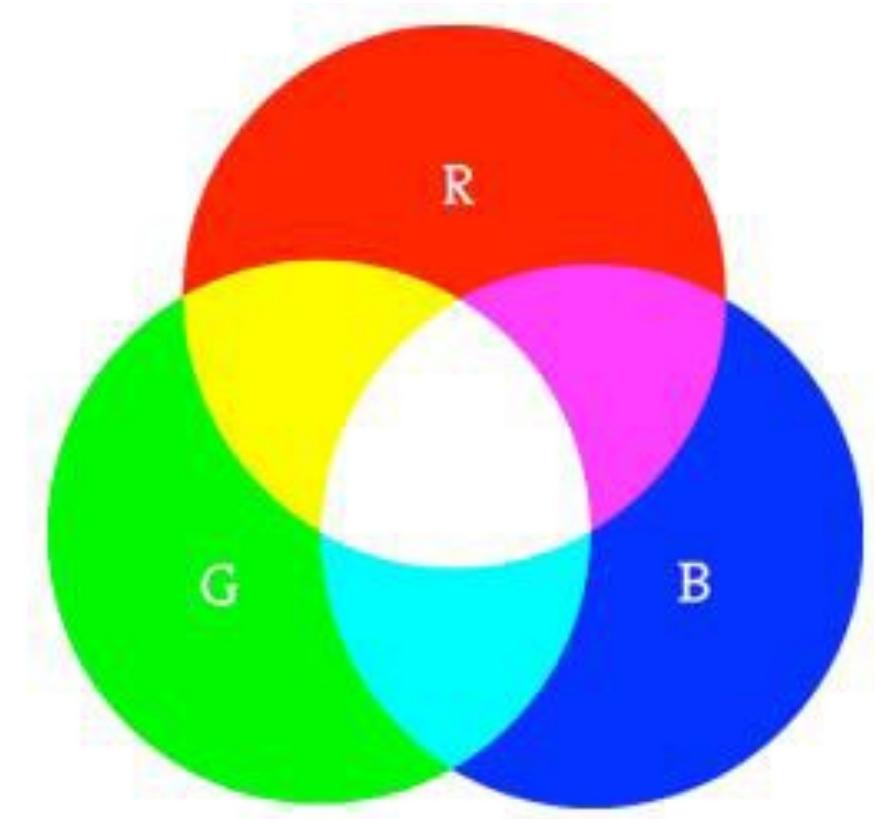


COLOR

luminance (black & white; grayscale)

background(255); // clear background





COLOR

RGB (default)

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

COLOR

RGBA

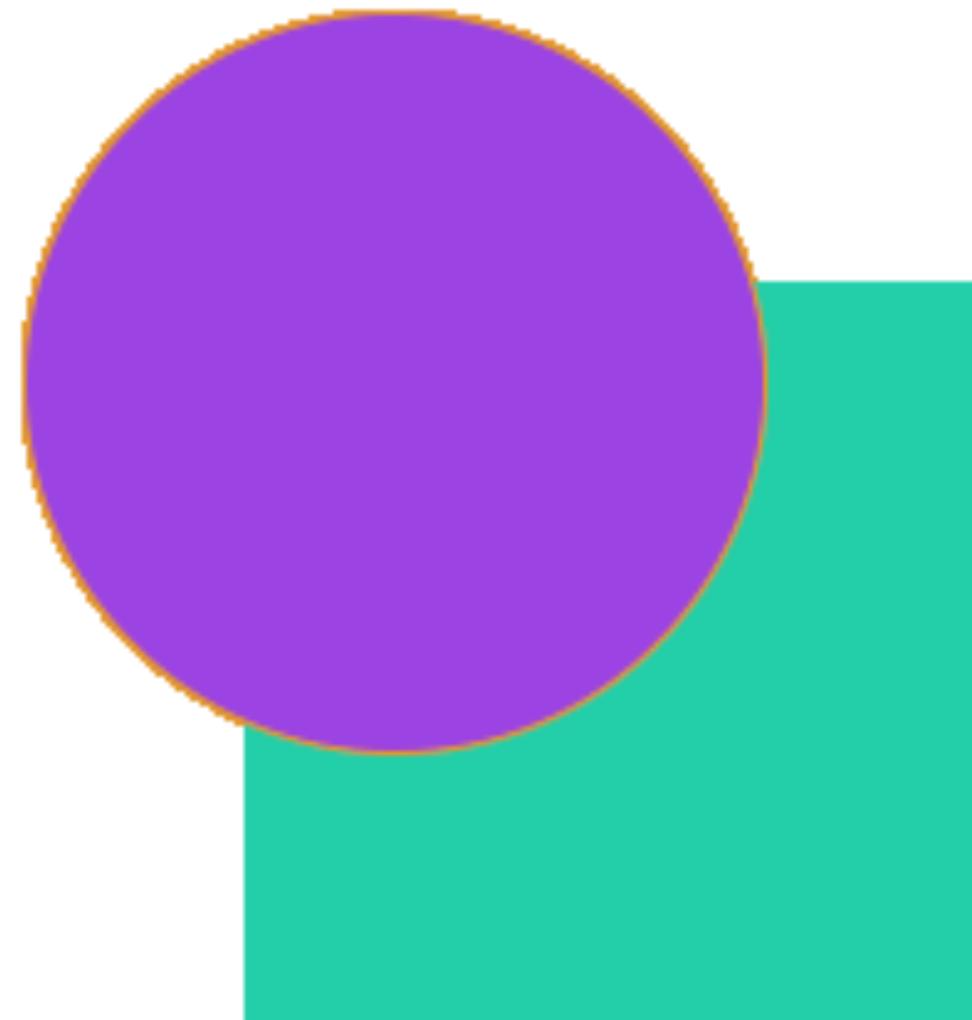
a = alpha / transparency / opacity

0 = transparent; 255 = opaque (solid)

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

PROPERTIES

```
noStroke () ;  
fill (c1) ;  
rect ( . . . ) ;  
  
fill (c2) ;  
stroke (c3) ;  
ellipse ( . . . ) ;
```



PROPERTIES

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

PROPERTIES

Graphics is a STATE MACHINE.

When you set a state (no matter where in the code),
that state will remain until changed

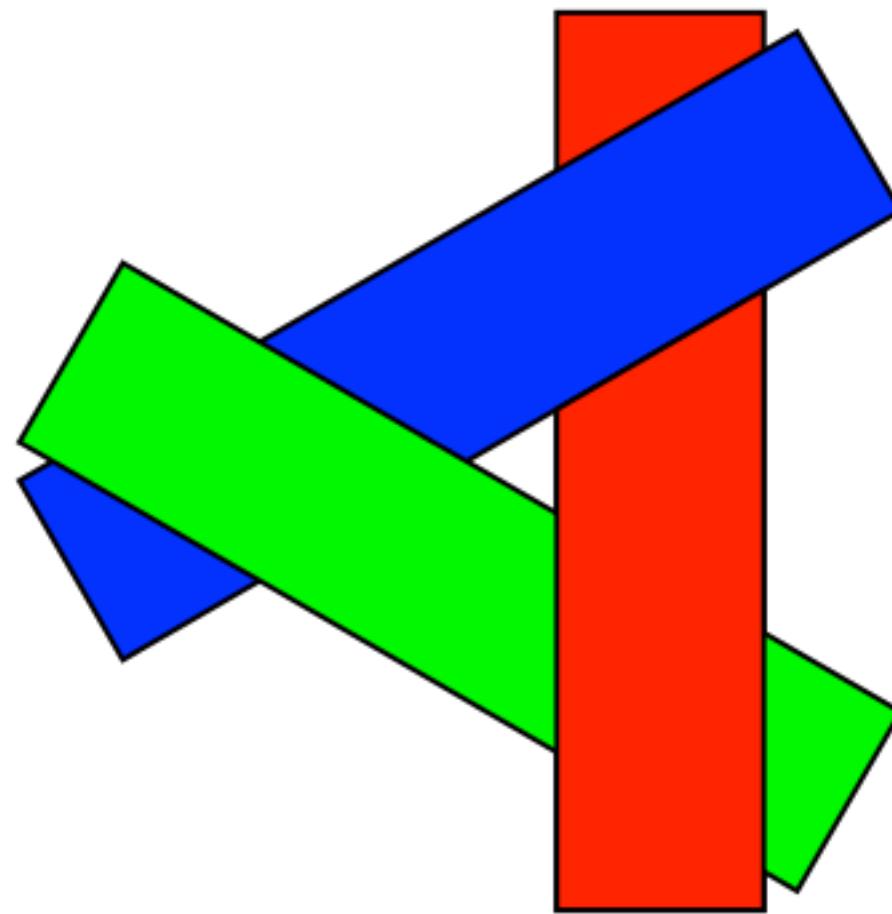
This is particularly noticeable with color



ORDER

shapes are painted one at a time
overlap can occur

some shapes are not supported



ANIMATION

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

runs once

Cycles (automatically)



TEXT

```
// in setup()  
PFont myFont;  
myFont = createFont("Georgia", 32);  
  
// in draw()  
textFont(myFont);  
textAlign(CENTER, CENTER);  
text("Hello, World!", width/2, height/2);
```

PROGRAMMING



STRUCTURE

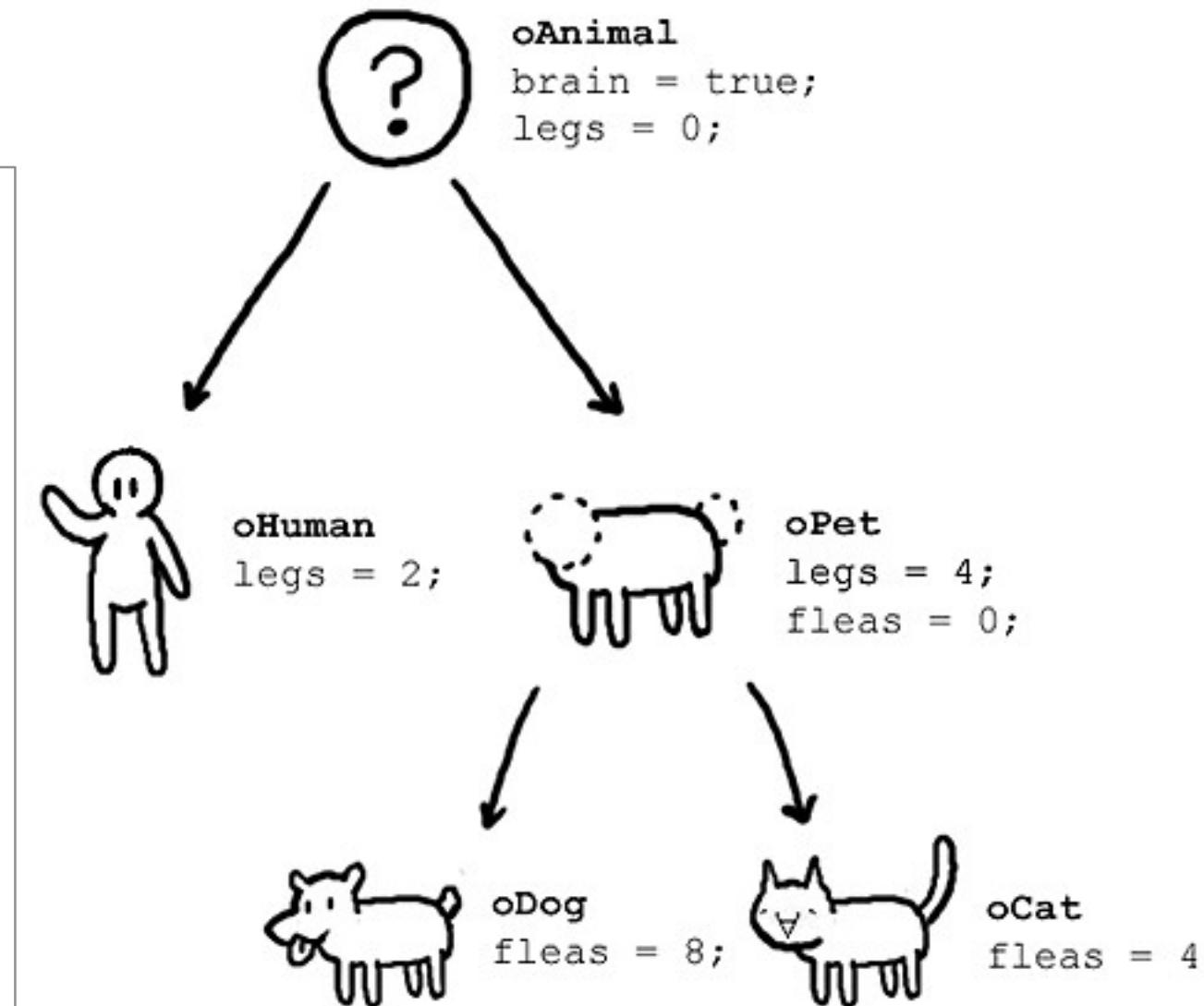
comments, variables, arrays, loops



OBJECT-ORIENTED

with classes

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



BUILT-IN TRIGONOMETRY FUNCTIONS

[acos\(\)](#) / [asin\(\)](#) / [atan\(\)](#) / [atan2\(\)](#)

[degrees\(\)](#) / [radians\(\)](#)

[cos\(\)](#) / [sin\(\)](#) / [tan\(\)](#)



BUILT-IN CALCULATION FUNCTIONS

CALCULATION

max() / min()

abs() / ceil() / constrain() / floor() / round()

lerp() / map() / norm()

dist() / mag()

exp() / log() / pow() / sq() / sqrt()



PVECTOR

Container for points and vectors

Variety of functionalities—vector arithmetic, dot/cross products, angles, normalization, and interpolation

<https://processing.org/reference/PVector.html>

<http://processing.github.io/processing-javadocs/core/processing/core/PVector.html>

STUDY AND USE THIS CLASS!



OBJECT STORAGE

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

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

Table, XML, JSON

(and anything else Java!)



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()
```

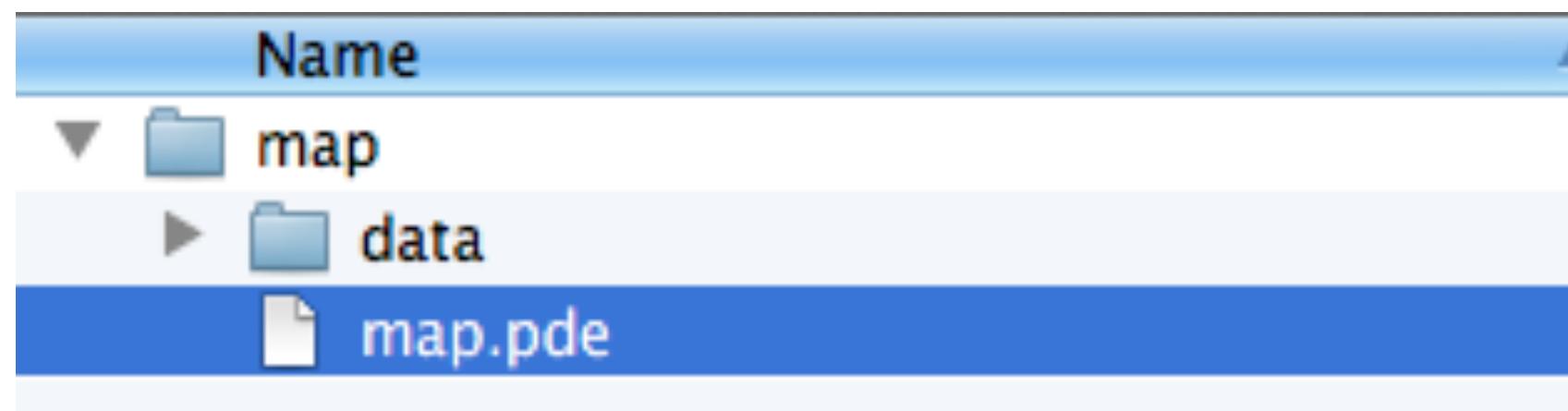
```
void keyTyped()
```

keyPressed
key
keyCode



FOLDER STRUCTURE

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

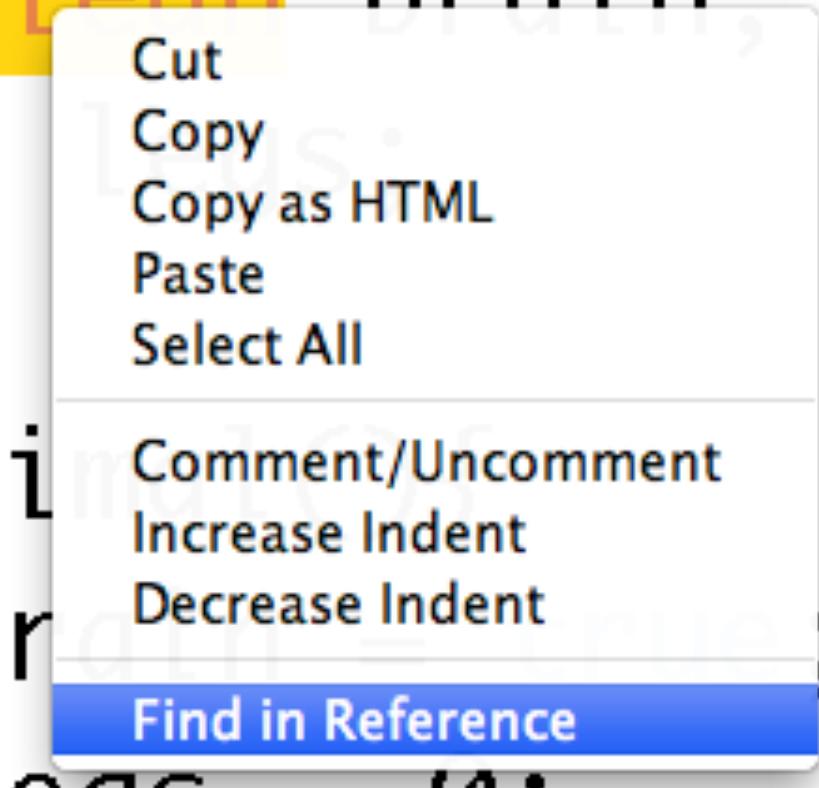


optional data folder (for images, input)

DOCUMENTATION
available online
also in the PDE

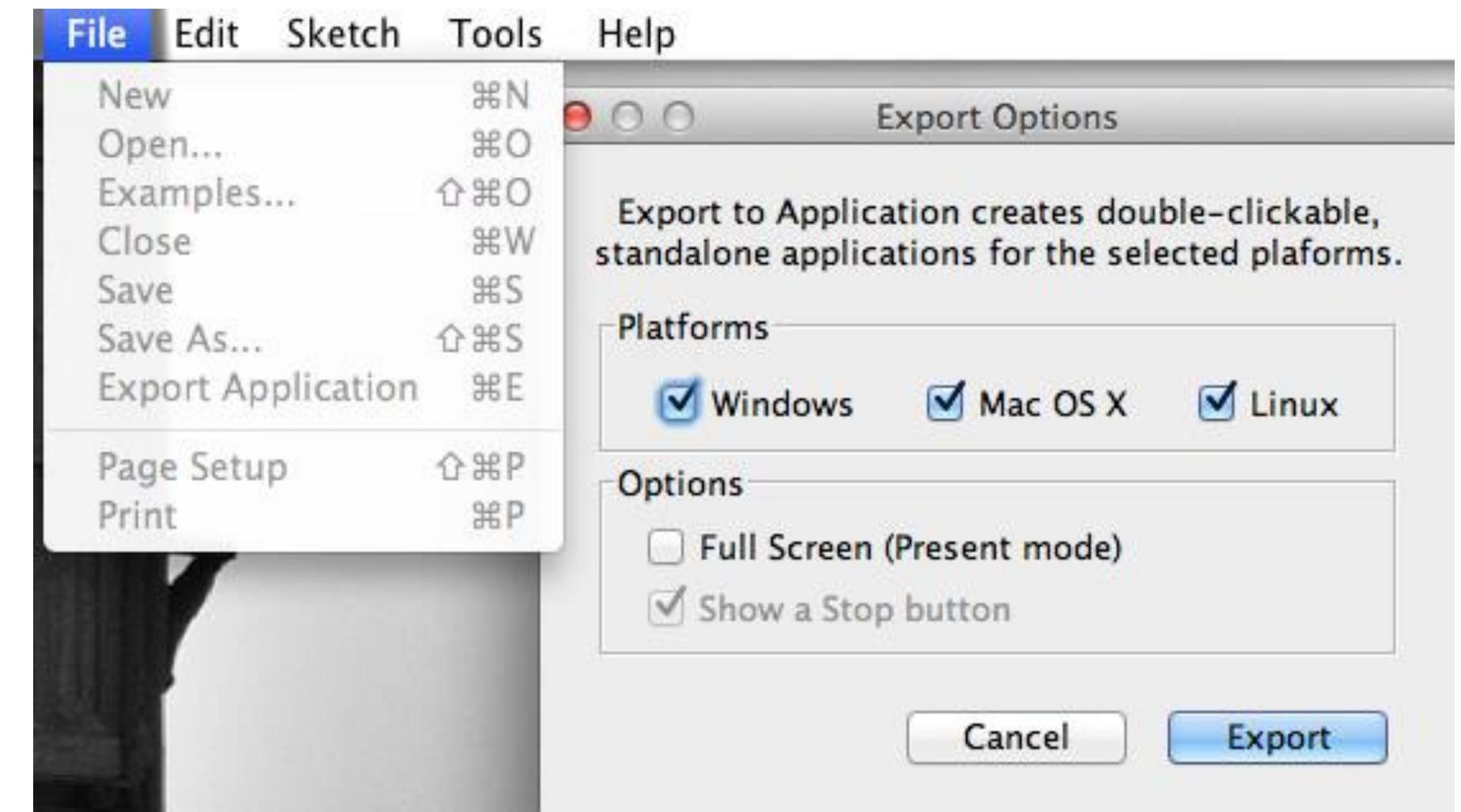
<http://processing.org/reference/>

```
class oAnimal{  
    boolean brain.  
    int  
    oAnimal  
    brain;  
    brain;  
    brain - 100 - 100;
```



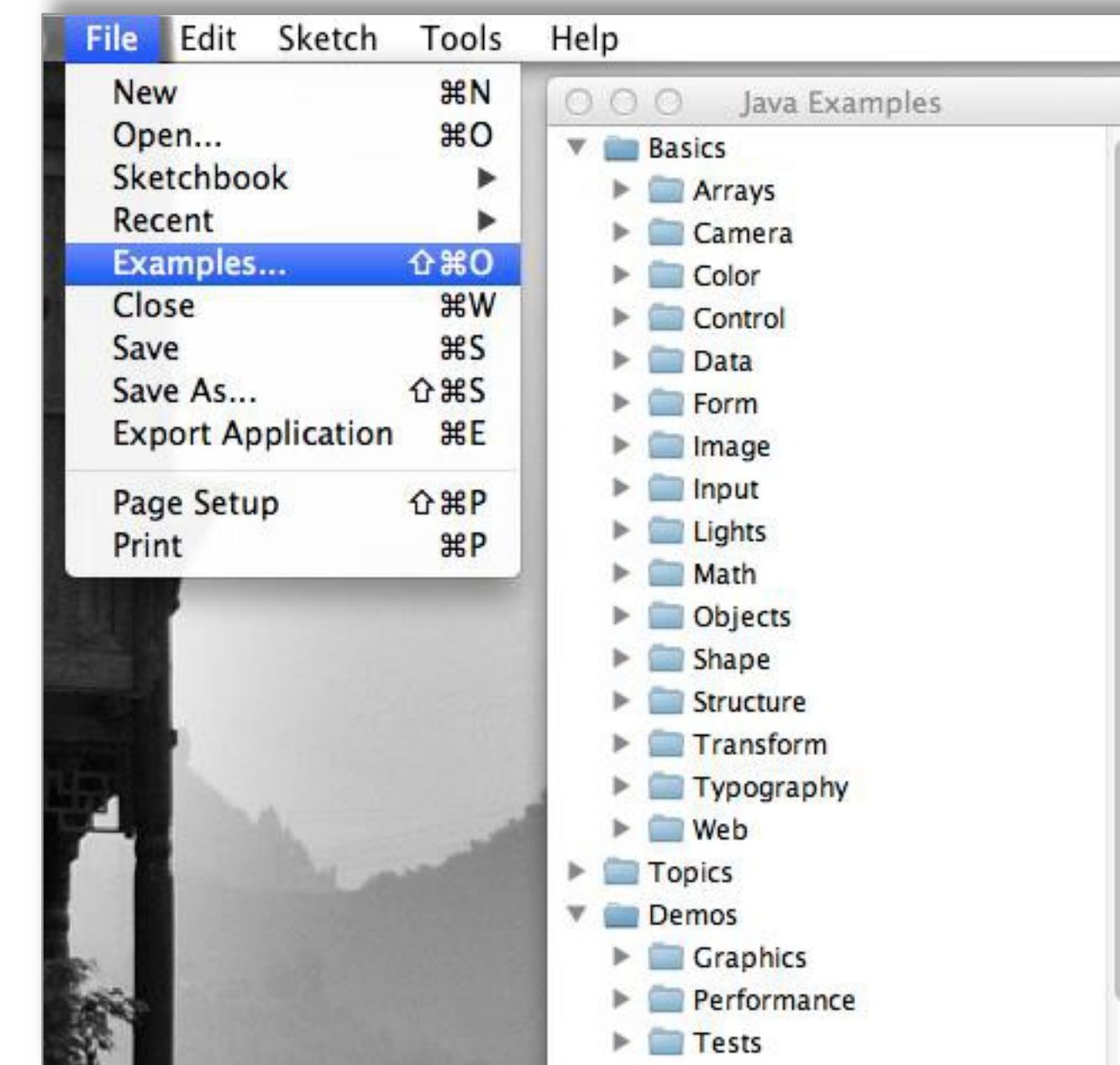
EXPORTING

creating applications is simple



EXAMPLES

variety of samples



DEMO

