PPEM 2022

Introducción a Processing

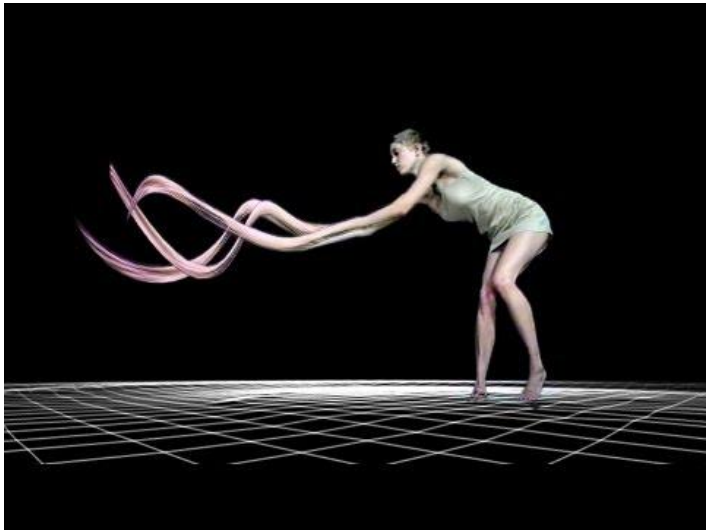
Aviso

Me olvidé que la gente de Licenciatura en Danza Contemporánea se inscribió a una materia que es de 16 a 18.

Hasta el 12.04 seguimos de 17 a 19, el **12.04** nos reunimos de **16 a 18** y vemos los horarios de todos.

Artes escénicas y tecnología

Le Sacre du Printemps – Klaus Obermaier & Ars Electronica Futurelab



Performance digital

Artes escénicas en toda su variedad (teatro, ópera, danza, performance,..) que incorporan la tecnología a la creación de la obra.

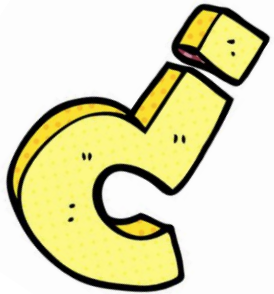
Solo se puede hablar de Performance Digital si la tecnología cumple un papel clave en el hecho artístico en vez de ser sólo una herramienta adicional.

Performance digital en el mundo

Laterna Magika https://youtu.be/gfL9hoejAn8
Wooster Group https://youtu.be/_10u984AvzE?t=116
Daito Manabe https://www.youtube.com/watch?v=_woNBiIyOKI
Chunky Moves https://www.youtube.com/watch?v=sbjOMualLVs
Klaus Obermeier https://vimeo.com/58661899
Adrien M / Claire B https://www.am-cb.net/en/projets/equinoxe
The Gertrude Stein Repertory Theatre
https://www.youtube.com/watch?v=tKL3A-JEKN0
Eeppi Ursin & Visa Oscar https://www.youtube.com/watch?v=ulsTPNre1bs
Troika Ranch https://vimeo.com/115035981

Temario

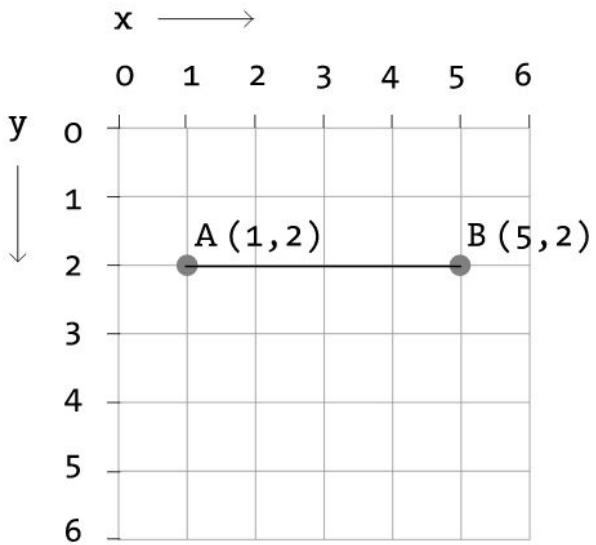
- Settings, setup y draw
- Tweak Mode
- Sistema de coordenadas (translate, rotate, scale, pushMatrix/popMatrix)
- Formas geométricas: Line, Point, Rect, Quad, Ellipse, Arc, Triangle, Vertex
- Espacios de colores (RGB, HSV, notación hexadecimal)
- Eventos de mouse y teclado
- Debug
- Librerías externas



¿Todos tienen Processing?

Settings, setup y draw

Una línea en Processing




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

Punto A Punto B

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

Cuando tengan dudas sobre las funciones

https://processing.org/reference/

Ejemplo sin settings, setup o draw

```
size(200, 200); // tamaño de la ventana de dibujo en pixeles
line(10, 10, 190, 190); // dibujamos un línea
```

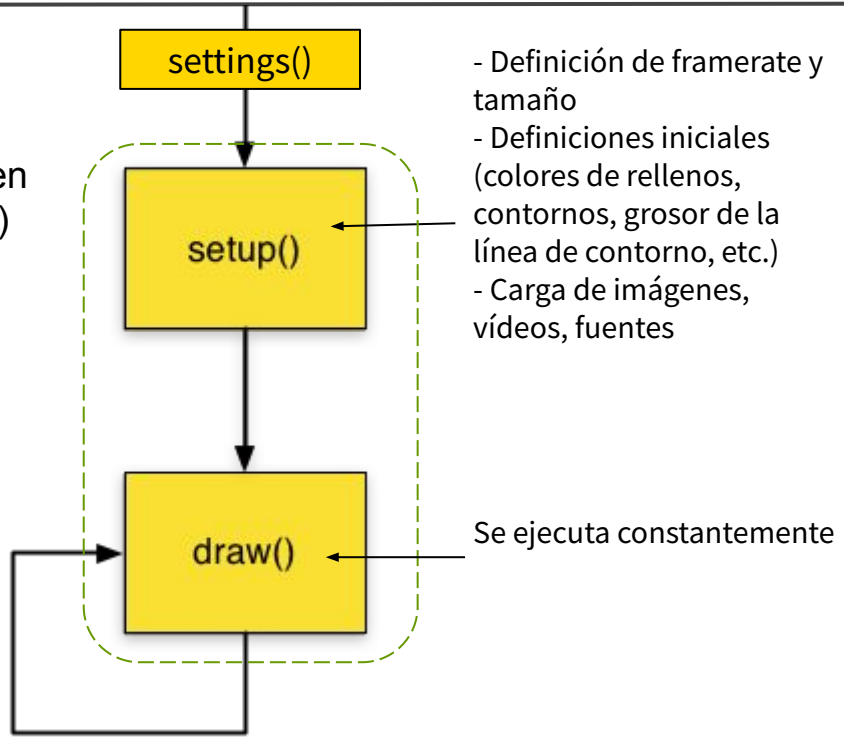
)

Setup y draw

```
int posX1 = 10;
int posX2 = 190;

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  line(posX1, 10, posX2, 190);
}
```

Variables globales

Tamaño del dibujo en píxeles (ancho, alto)

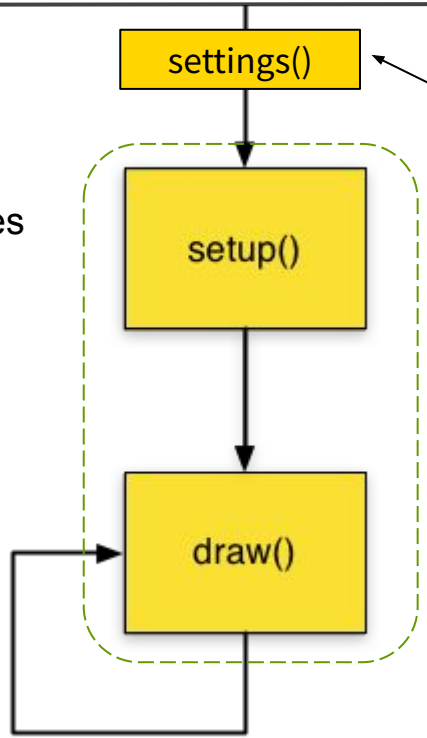


Settings y draw

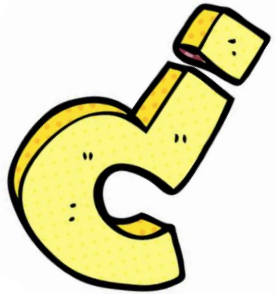
```
int posX1 = 10;
int posX2 = 190;

void settings() {
 size(displayWidth, displayHeight);
}

void draw() {
 background(255);
 line(posX1, 10, posX2, 190);
}
```

Variables globales

Tamaño del dibujo en píxeles usando variables



Generalmente no se usa, solamente si queremos definir el tamaño de la ventana de dibujo usando variables o cuando NO usamos IDE de Processing para definir valores de size() y smooth().



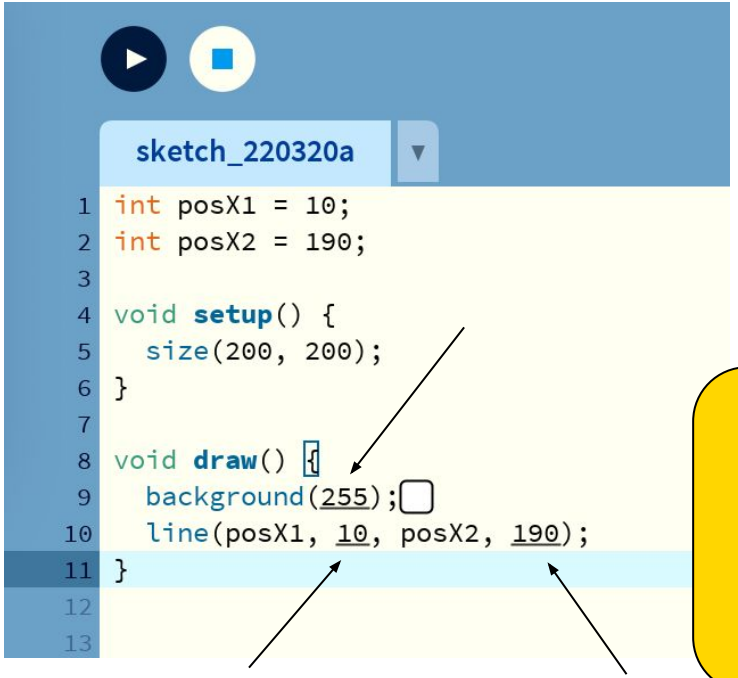
¿Dudas?

Tweak mode

Para manipular constantes dentro del *draw*

Tweak mode

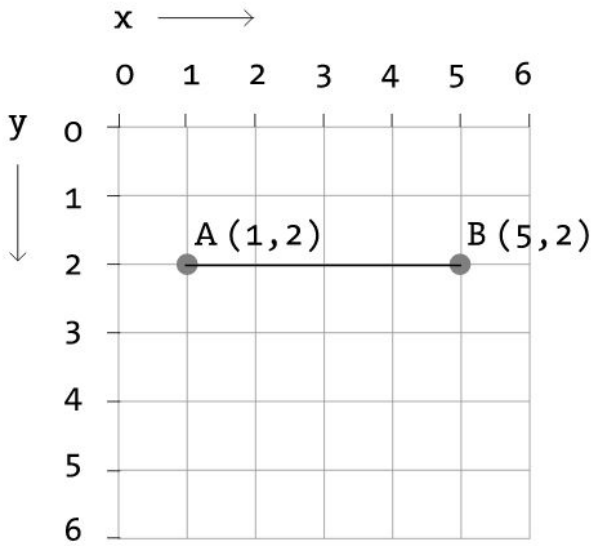
```
int posX1 = 10;
int posX2 = 190;

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  line(posX1, 10, posX2, 190);
}
```



Para usar Tweak Mode, en vez de correr el código apretando el botón “play” vamos a Sketch -> Tweak

Sistema de coordenadas

Sistema de coordenadas

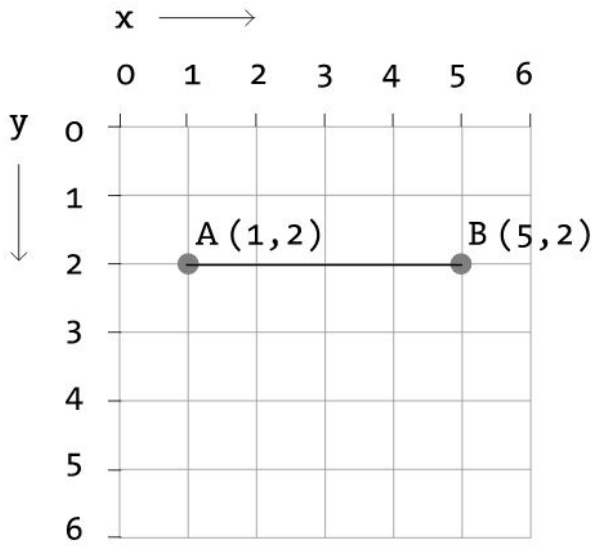



```
line(x1,y1,x2,y2);
```

Punto A Punto B

Sistema de coordenadas

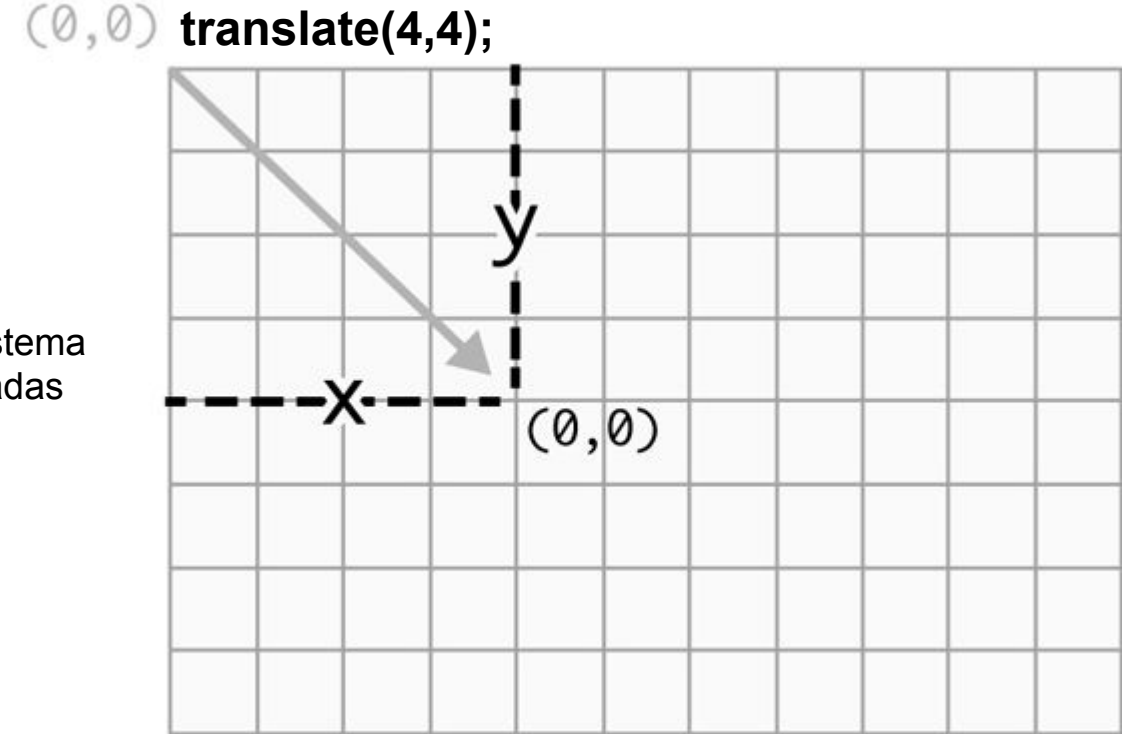
```
int posX1 = 10;
int posX2 = 190;

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  line(posX1, 10, posX2, 190);
}
```



Translate

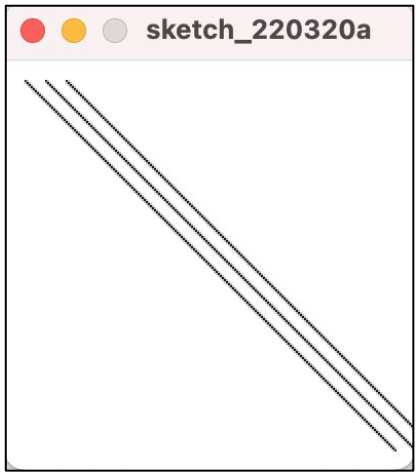
```
int posX1 = 10;
int posX2 = 190;

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  translate(10,0);
  line(posX1, 10, posX2, 190);
}
```

Muevo el sistema de coordenadas

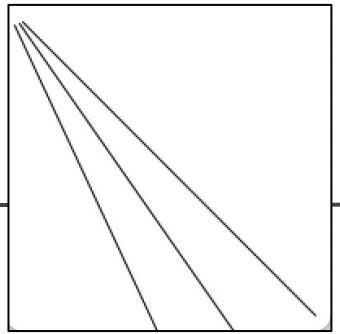


Translate

```
int posX1 = 10;
int posX2 = 190;
void setup() {
  size(200, 200);
}
void draw() {
  background(255);
  line(posX1, 10, posX2, 190);
  translate(10,0);
  line(posX1, 10, posX2, 190);
  translate(10,0);
  line(posX1, 10, posX2, 190);
}
```

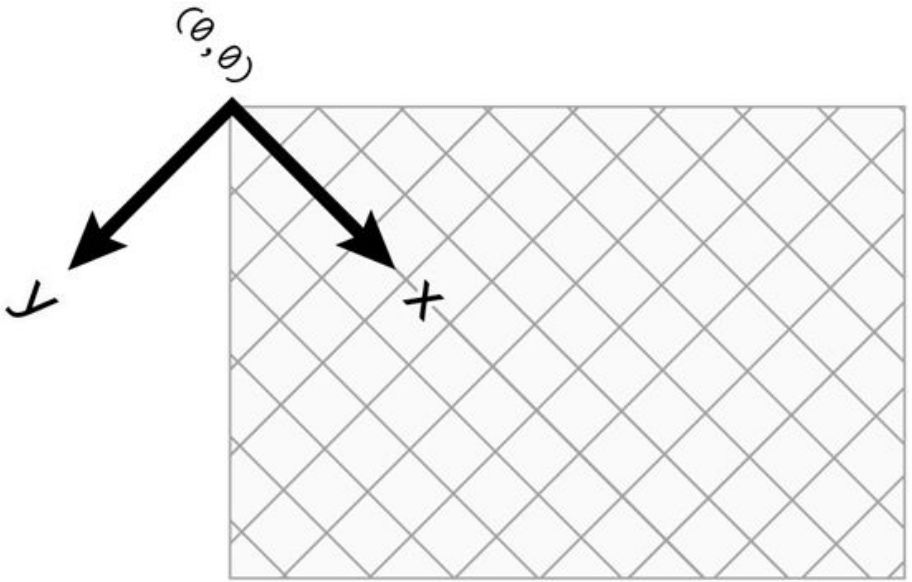


Rotate



```
int posX1 = 10;
int posX2 = 190;
void setup() {
  size(200, 200);
}
void draw() {
  background(255);
  line(posX1, 10, posX2, 190);
  rotate(radians(10)); // rotate recibe radianes (0 a TWO_PI)
  line(posX1, 10, posX2, 190);
  rotate(radians(10));
  line(posX1, 10, posX2, 190);
  rotate(radians(10));
}
```

Roto 10 grados

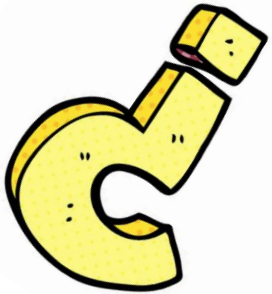


rotate(radians(45));

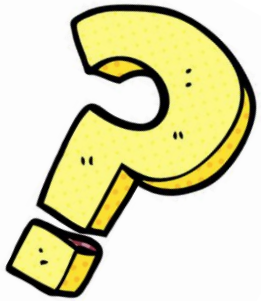
Translate + rotate + tweak mode

```
void setup() {
  size(200, 200);
}
void draw() {
  background(255);
  translate(width/2,height/2);
  rotate(radians(0));
  line(10, 10, 190, 190);
  rotate(radians(10));
  line(10, 10, 190, 190);
  rotate(radians(10));
  line(10, 10, 190, 190);
  rotate(radians(10));
}
```

A mover las líneas!



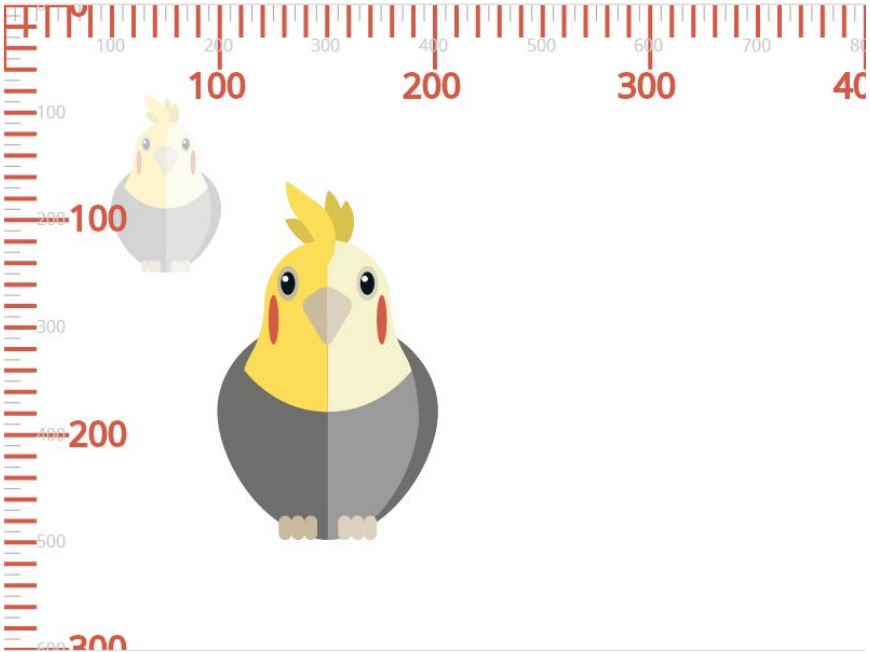
¿Dudas?
¿Cosas interesantes para compartir?



Scale

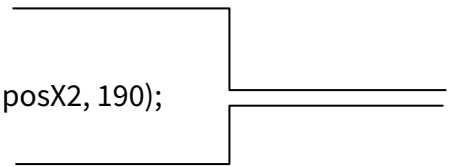
```
int posX1 = 10;
int posX2 = 190;

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  scale(10);
  line(posX1, 10, posX2, 190);
}
```

scale(2);

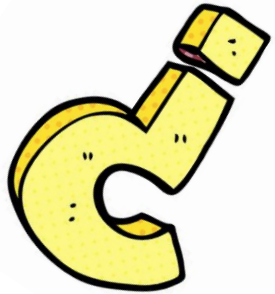


Push/pop matrix

```
int posX1 = 10;
int posX2 = 190;

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  pushMatrix();
  scale(10);
  line(posX1, 10, posX2, 190);
  popMatrix();

  line(posX1, 10, posX2, 190);
}
```

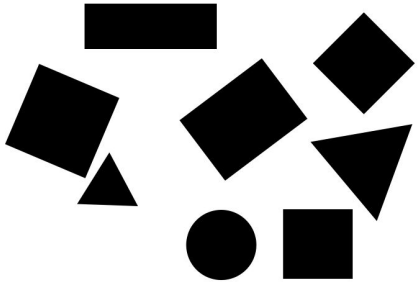


Las transformaciones definidas entre push y pop no influyen el resto del código



¿Dudas?

Formas geométricas



Formas geométricas

Line

Point

Rect

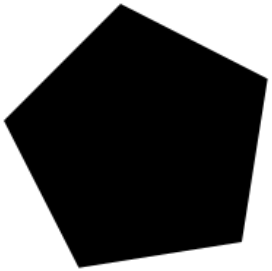
Quad

Ellipse

Arc

Triangle

Vertex



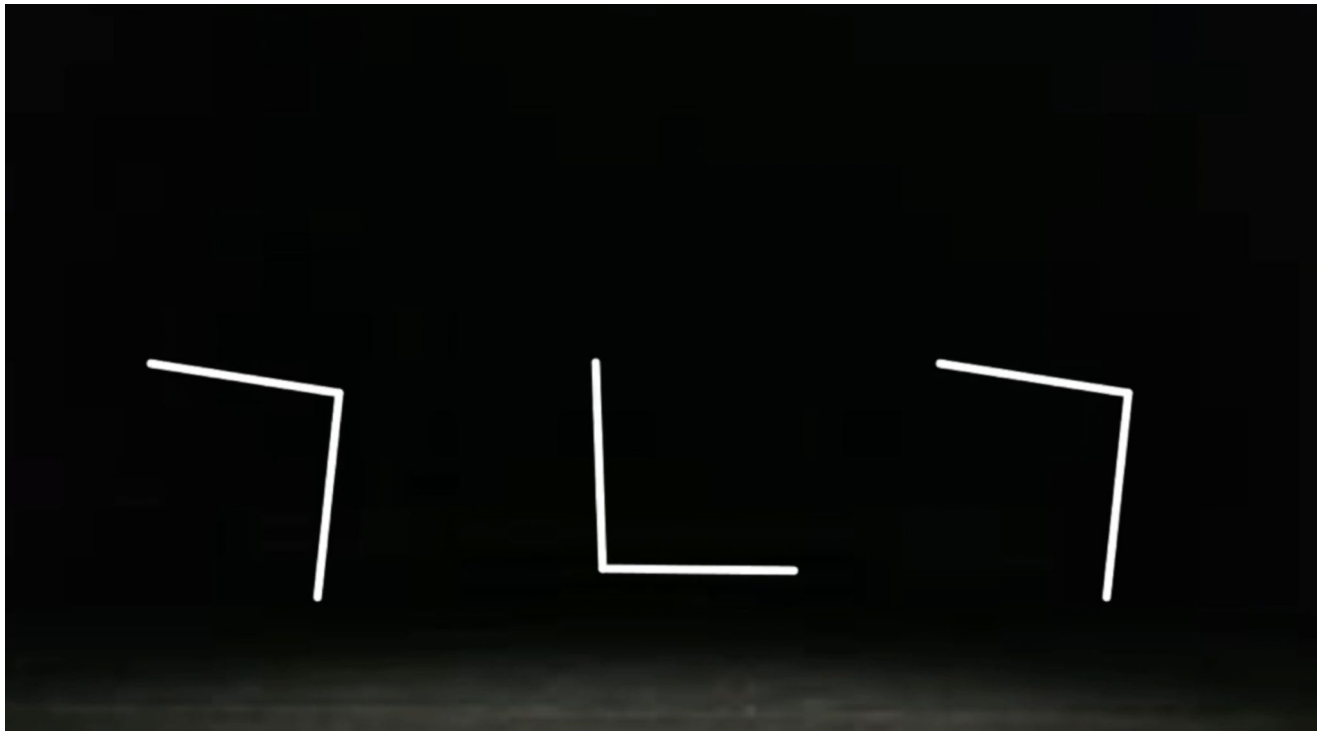
Line

Ya vimos ejemplos simples, así que veamos algo más interesante:

https://processing.org/examples/linear.html

https://processing.org/examples/continuouslines.html

Hablando de líneas...

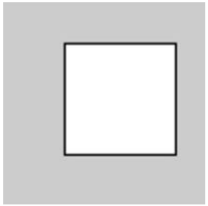


Point

```
size(400, 400);
strokeWeight(10);
point(12, 8);
point(34, 8);
point(34, 30);
point(12, 30);
```

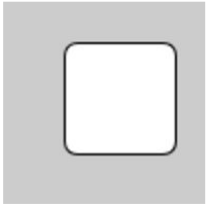
Rect

rect(x, y, w, h);



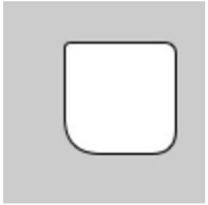
```
rect(30, 20, 55, 55);
```

rect(x, y, w, h, radio);



```
rect(30, 20, 55, 55, 7);
```

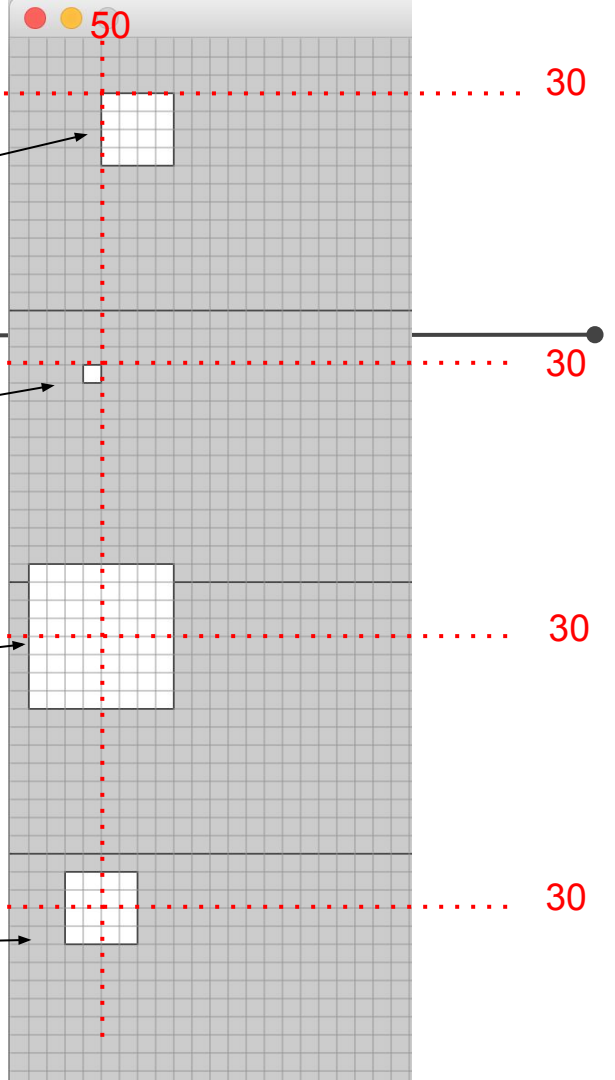
rect(x, y, w, h, topLeft, topRight, bottomRight, bottomLeft);



```
rect(30, 20, 55, 55, 3, 6, 12, 18);
```

rectMode

```
rectMode(CORNER);
rect(50, 30, 40, 40); // rect(x,y,w,h), (x,y) de la esquina superior izquierda

translate(0,150);
line(0,0,width,0);
rectMode(CORNERS);
rect(50, 30, 40, 40);  // rect(x,y,x2,y2)

translate(0,150);
line(0,0,width,0);
rectMode(RADIUS);
rect(50, 30, 40, 40); // rect(x,y, 0.5*w, 0.5*h), (x,y) del centro

translate(0,150);
line(0,0,width,0);
rectMode(CENTER);
rect(50, 30, 40, 40); // rect(x,y, w, h), (x,y) del centro
```

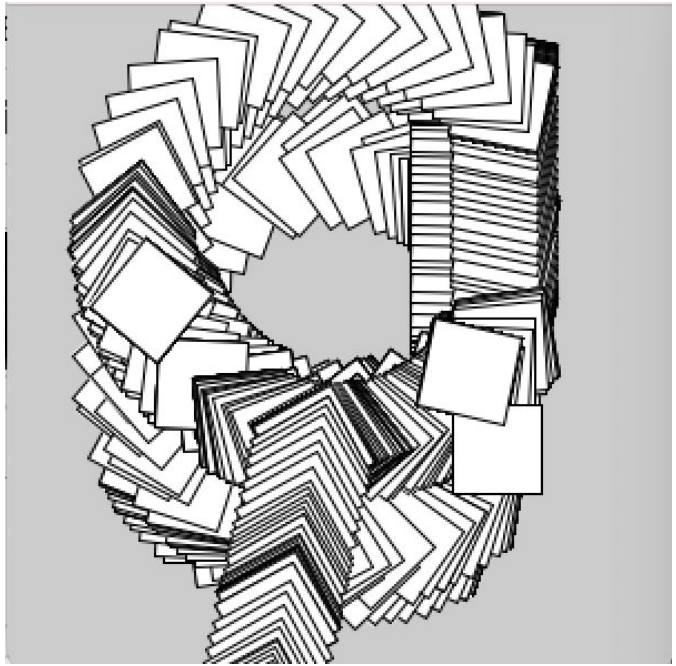


A mover los cuadrados!

rectMode + tweak mode

```
void setup() {
  size(300, 300);
  rectMode(CORNER);
  //rectMode(CORNERS);
  //rectMode(RADIUS);
  //rectMode(CENTER);
}
void draw() {
  translate(width/2, height/2);
  rect(50, 30, 40, 40);
  translate(0, 10);
  rotate(radians(10));
  rect(50, 30, 40, 40);
  translate(0, 10);
  rotate(radians(10));
  rect(50, 30, 40, 40);
  translate(0, 10);
  rotate(radians(10));
}
```

Variables predefinidas en Processing



Quad

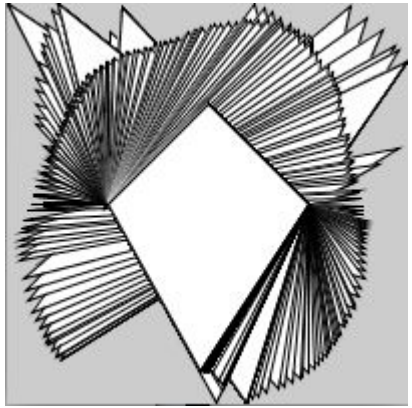
quad(x1, y1, x2, y2, x3, y3, x4, y4);

```
quad(38, 31, 86, 20, 69, 63, 30, 76);
```

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

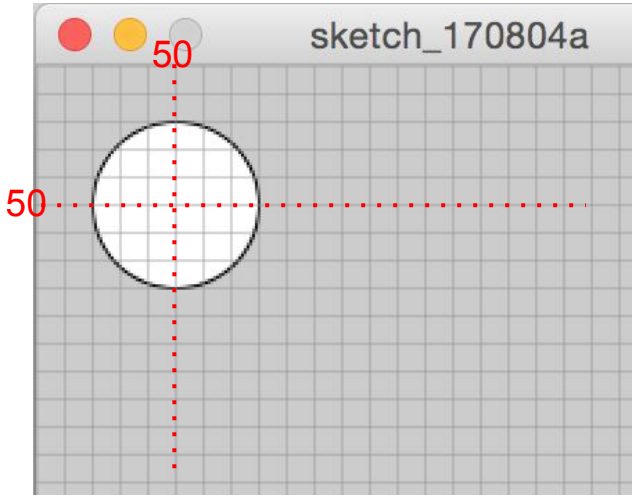
Ejemplo quad

```
void setup(){
  size(200,200);
}
void draw(){
  quad(50, 100, 100, 50, 150, 100, mouseX, mouseY);
}
```



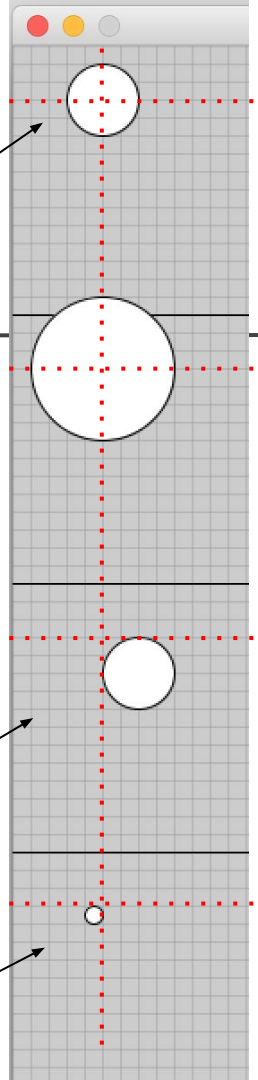
ellipse

```
ellipse(x, y, w, h);
ellipse(50, 50, 60, 60);
```



ellipseMode

```
ellipseMode(CENTER);
ellipse(50, 30, 40, 40); // ellipse(x,y, w, h), (x,y) del centro

translate(0,150);
line(0,0,width,0);
ellipseMode(RADIUS);
ellipse(50, 30, 40, 40); // ellipse(x,y, 0.5*w, 0.5*h), (x,y) del centro

translate(0,150);
line(0,0,width,0);
ellipseMode(CORNER);
ellipse(50, 30, 40, 40); // ellipse(x,y,w,h), (x,y) de la esquina superior izquierda

translate(0,150);
line(0,0,width,0);
ellipseMode(CORNERS);
ellipse(50, 30, 40, 40);  // ellipse(x,y,x2,y2) de las dos esquinas del "bounding box"
```



ellipseMode

```
void setup() {
  size(300, 300);
  ellipseMode(CENTER);
  //ellipseMode(CORNER);
  //ellipseMode(CORNERS);
  //ellipseMode(RADIUS);
}
void draw() {
  ellipse(mouseX, mouseY, 40, 40);
}
```

Círculos y el mouse más divertidos

https://processing.org/examples/storinginput.html

¿Qué está pasando?

arc

arc(x, y, w, h, start, stop);

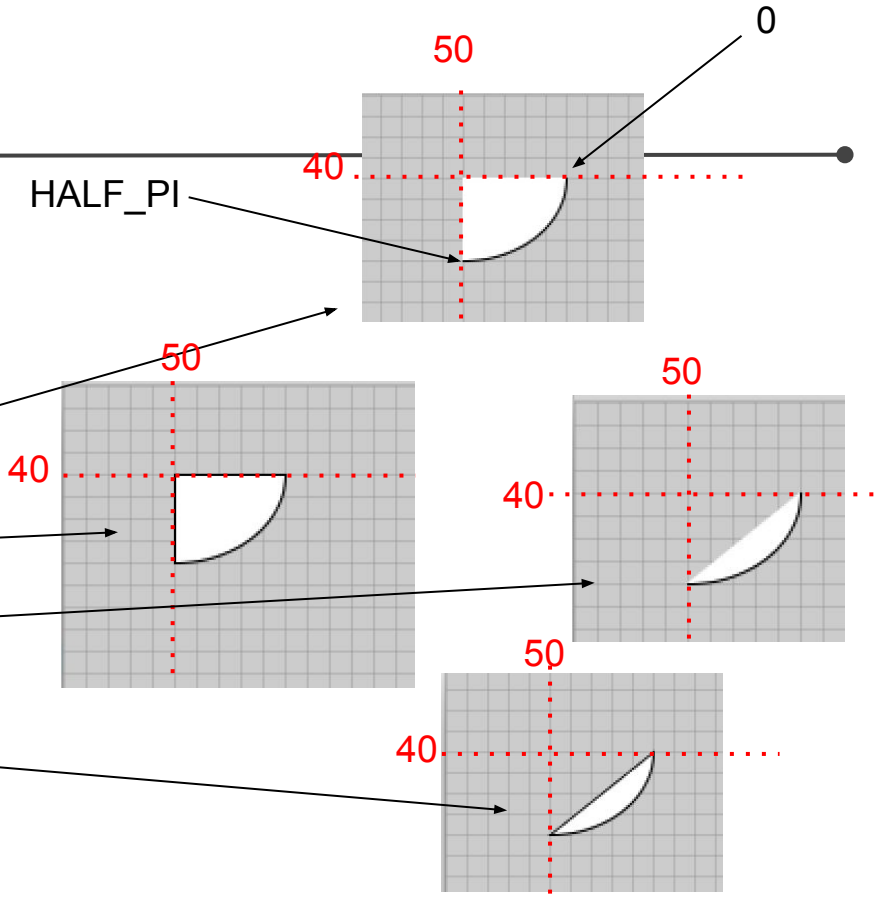
arc(x, y, w, h, start, stop, mode);

arc(50, 40, 100, 80, 0, HALF_PI); = arc(50, 40, 100, 80, 0, radians(90));

arc(50, 40, 100, 80, 0, HALF_PI, PIE);

arc(50, 40, 100, 80, 0, HALF_PI, OPEN);

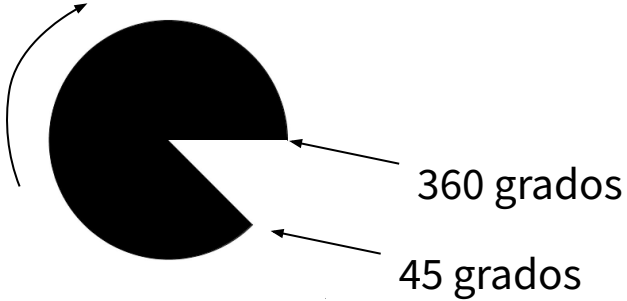
arc(50, 40, 100, 80, 0, radians(90),CHORD);



! ellipseMode para cambiar modo de dibujo !

QUARTER_PI, HALF_PI, PI, TWO_PI

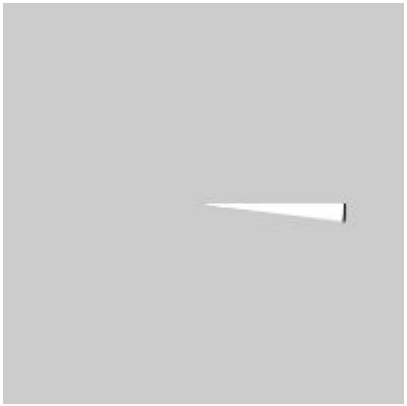
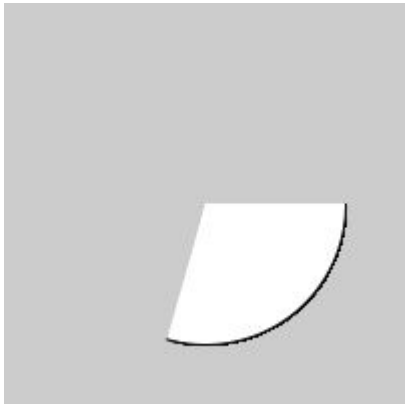
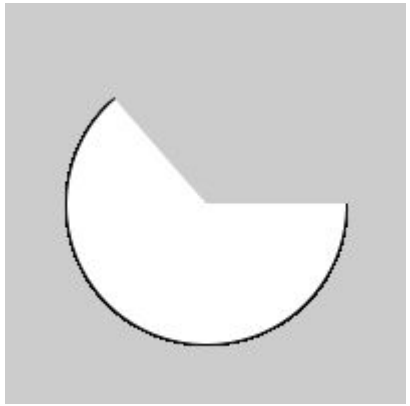
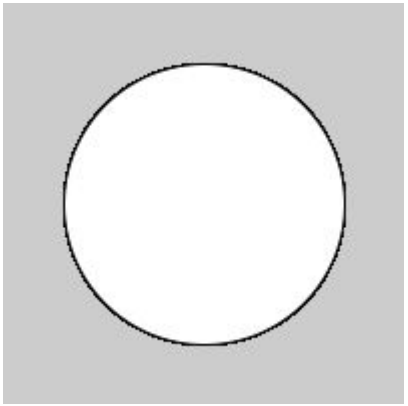
Ejemplo arc




```
int addition = 0;
void setup(){
  size(200,200);
}
void draw(){
  background(0);
  arc(width/2, height/2, 140, 140, radians(0 + addition), radians(90+addition));
  addition += 1;
  addition %= 360;
}
```

Pregunta

¿Cómo podríamos animar un círculo que va dibujándose usando arc?

Ejemplo arc - dibujar un círculo desde 0

```
int addition = 0;
void setup(){
  size(200,200);
}
void draw(){
  // background(0);
  arc(width/2, height/2, 140, 140, radians(0), radians(addition));
  addition += 1;
  addition %= 360;
}
```

Para no borrar lo ya dibujado comentamos *background()*

Para ir avanzando de a poquito

Siempre desde 0

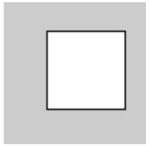
triangle

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

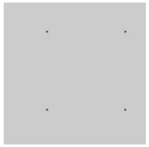
```
triangle(30, 75, 58, 20, 86, 75);
```

vertex

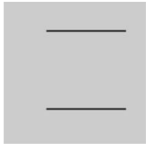
```
beginShape();
vertex(x,y);
vertex(x2,y2);
…
vertex(xn,yn);
endShape();
```



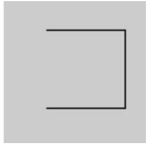
```
beginShape();
vertex(30, 20);
vertex(85, 20);
vertex(85, 75);
vertex(30, 75);
endShape(CLOSE);
```



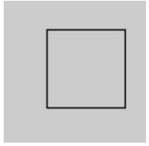
```
beginShape(POINTS);
vertex(30, 20);
vertex(85, 20);
vertex(85, 75);
vertex(30, 75);
endShape();
```



```
beginShape(LINES);
vertex(30, 20);
vertex(85, 20);
vertex(85, 75);
vertex(30, 75);
endShape();
```



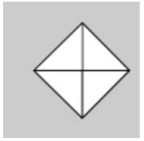
```
noFill();
beginShape();
vertex(30, 20);
vertex(85, 20);
vertex(85, 75);
vertex(30, 75);
endShape();
```



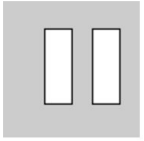
```
noFill();
beginShape();
vertex(30, 20);
vertex(85, 20);
vertex(85, 75);
vertex(30, 75);
endShape(CLOSE);
```

```
beginShape(TRIANGLES);
vertex(30, 75);
vertex(40, 20);
vertex(50, 75);
vertex(60, 20);
vertex(70, 75);
vertex(80, 20);
endShape();
```

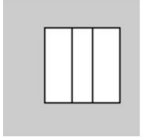
```
beginShape(TRIANGLE_STRIP);
vertex(30, 75);
vertex(40, 20);
vertex(50, 75);
vertex(60, 20);
vertex(70, 75);
vertex(80, 20);
vertex(90, 75);
endShape();
```



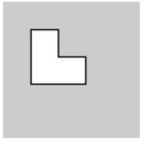
```
beginShape(TRIANGLE_FAN);
vertex(57.5, 50);
vertex(57.5, 15);
vertex(92, 50);
vertex(57.5, 85);
vertex(22, 50);
vertex(57.5, 15);
endShape();
```



```
beginShape(QUADS);
vertex(30, 20);
vertex(30, 75);
vertex(50, 75);
vertex(50, 20);
vertex(65, 20);
vertex(65, 75);
vertex(85, 75);
vertex(85, 20);
endShape();
```

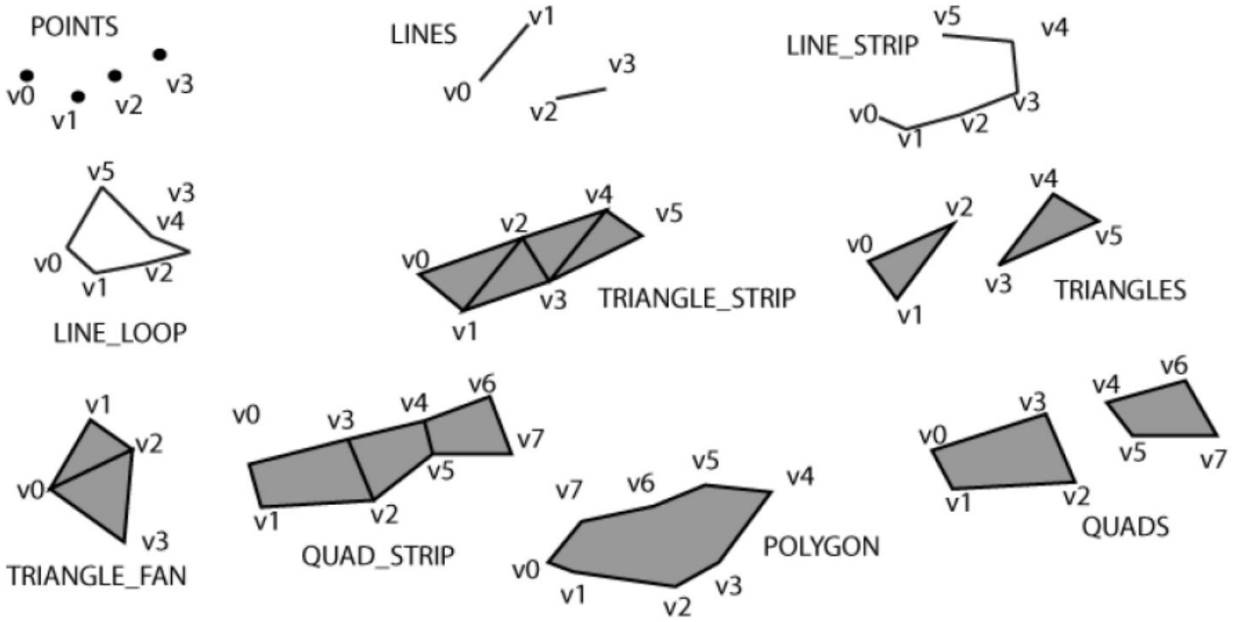


```
beginShape(QUAD_STRIP);
vertex(30, 20);
vertex(30, 75);
vertex(50, 20);
vertex(50, 75);
vertex(65, 20);
vertex(65, 75);
vertex(85, 20);
vertex(85, 75);
endShape();
```



```
beginShape();
vertex(20, 20);
vertex(40, 20);
vertex(40, 40);
vertex(60, 40);
vertex(60, 60);
vertex(20, 60);
endShape(CLOSE);
```

vertex


POINTS
v0
v1
v2
v3
LINES
v0
v1
v2
v3
LINE_STRIP
v0
v1
v2
v3
v4
v5
LINE_LOOP
v0
v1
v2
v3
v4
v5
TRIANGLE_STRIP
v0
v1
v2
v3
v4
v5
TRIANGLES
v0
v1
v2
v3
v4
v5
TRIANGLE_FAN
v0
v1
v2
v3
v4
v5
v6
v7
QUAD_STRIP
v0
v1
v2
v3
v4
v5
v6
v7
POLYGON
v0
v1
v2
v3
v4
v5
v6
v7
QUADS

Un ejemplo con lo que vimos hasta ahora

https://processing.org/examples/star.html