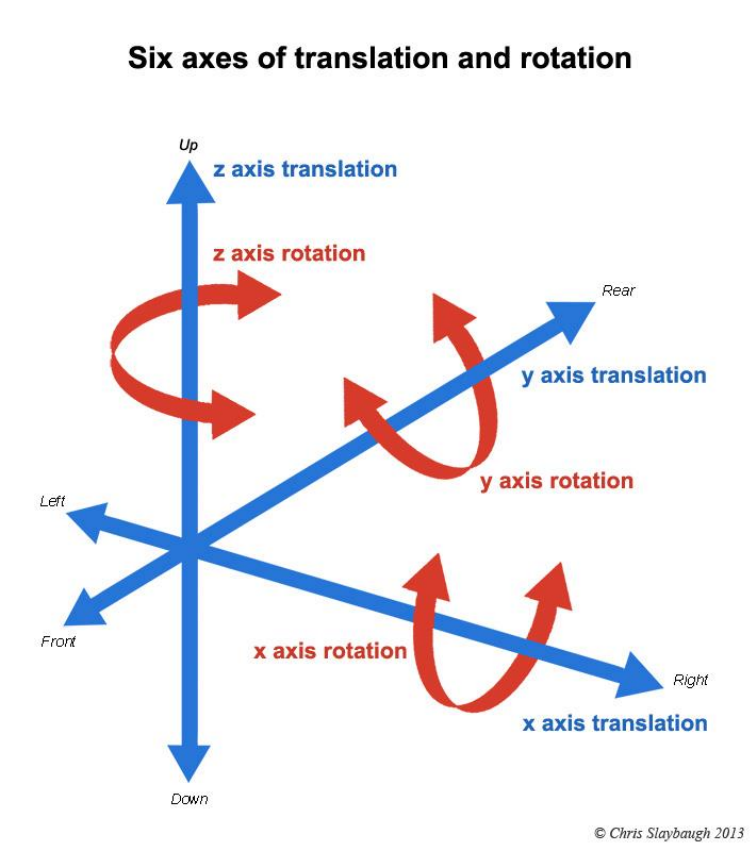
2018 Ajou

Jonghwa Park
suakii@gmail.com

GYEONGGI SCIENCE HIGH SCHOOL

3D: Translation and Rotation



- http://photomacrography.net/

Z Axis

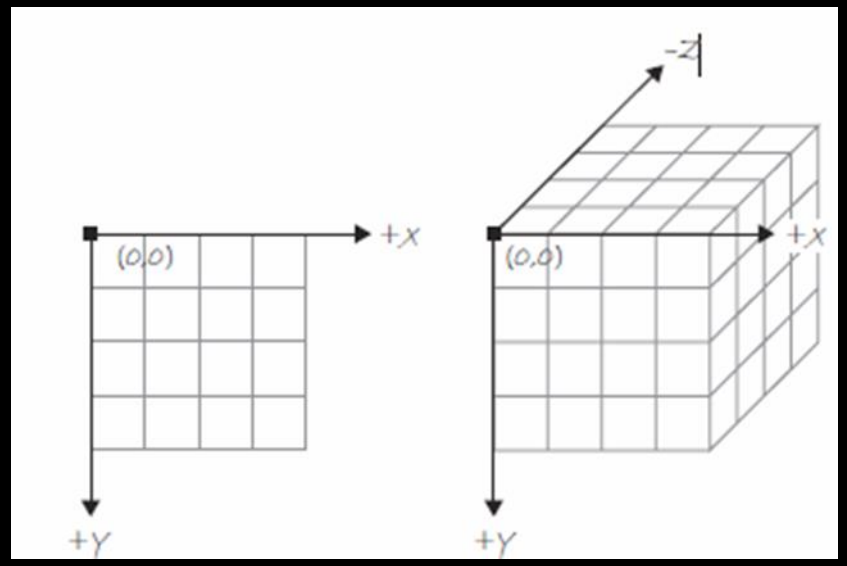
- In three-dimensional space a third axis refers to the depth of any given point.
- In a Processing sketch's window, a coordinate along the Z-axis indicates how far in front or behind the window a pixel lives.
- The Z-axis will create the illusion of three-dimensional space in your Processing window.

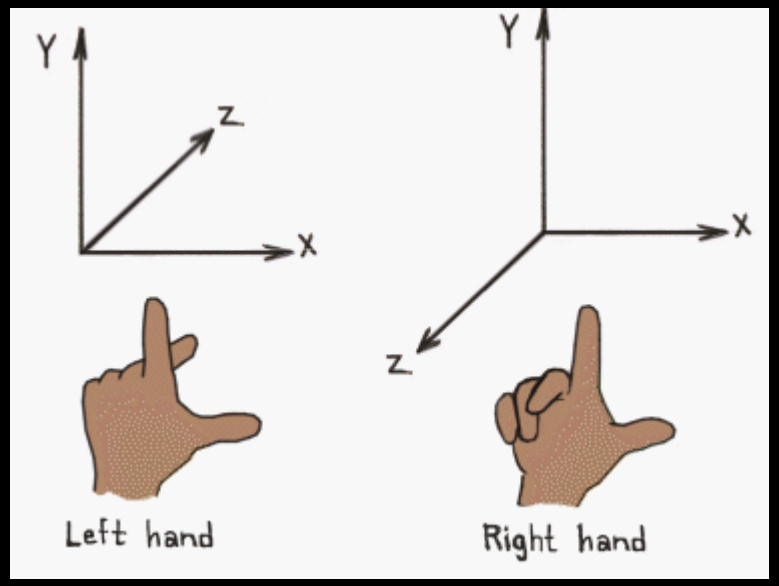
A growing rectangle

```
float r = 8;
void setup() {
  size(200,200);
}
void draw() {
  background(255);
  // Display a rectangle in the middle of the screen
  stroke(0);
  fill(175);
  rectMode(CENTER);
  rect(width/2,height/2,r,r);
  // Increase the rectangle size
  r++ ;
}
```

3D

- If we choose to use 3D coordinates, Processing will create the illusion for us.





Translate() function

- There is no rect(x, y, z…) in Processing.
- In order to user the z axis, we need to learn translate().

- The function translate() moves the origin point (0,0) relative to its previous state.

Translate

```
void setup() {
  size(200, 200);
}
void draw() {
  background(255);
  stroke(0);
  fill(175);
  // Grab mouse coordinates, constrained to window
  int mx = constrain(mouseX, 0, width);
  int my = constrain(mouseY, 0, height);
```

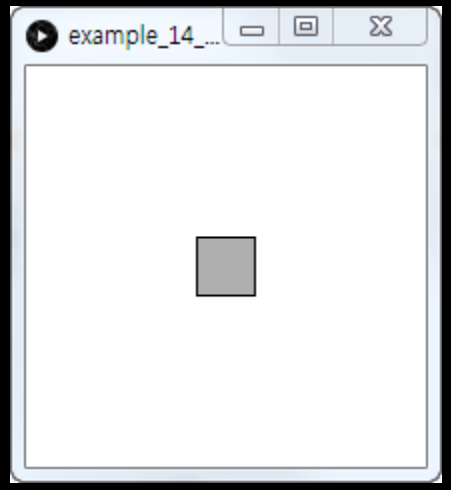
Translate

```
 // Translate to the mouse location
  translate(mx, my);
  ellipse(0, 0, 8, 8);
  // Translate 100 pixels to the right
  translate(100, 0);
  ellipse(0, 0, 8, 8);
  // Translate 100 pixels down
  translate(0, 100);
  ellipse(0, 0, 8, 8);
  // Translate 100 pixels left
  translate(-100, 0);
  ellipse(0, 0, 8, 8);
}
```

3D

```
float z = 0; // a variable for the Z (depth) coordinate
void setup() {
  size(200,200,P3D);
}
void draw() {
  background(0);
  stroke(255);
  fill(100);
  // Translate to a point before displaying a shape there
  translate(width/2,height/2,z);
  rectMode(CENTER);
  rect(0,0,8,8);
  z++; // Increment Z (i.e. move the shape toward the viewer)
}
```

3D

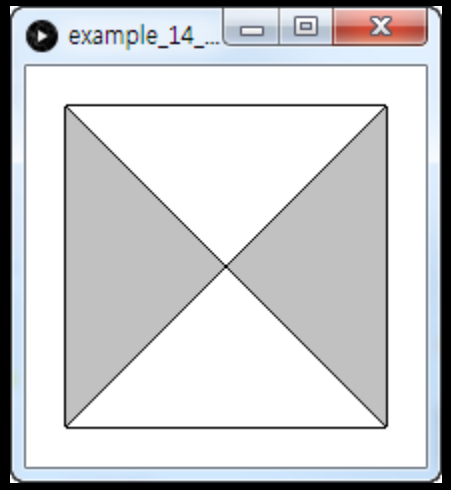


Vertex Shapes

- beginShape()
- vertex()
- endShape()

```
beginShape();
vertex(50,50);
vertex(150,50);
vertex(150,150);
vertex(50,150);
endShape(CLOSE);
```

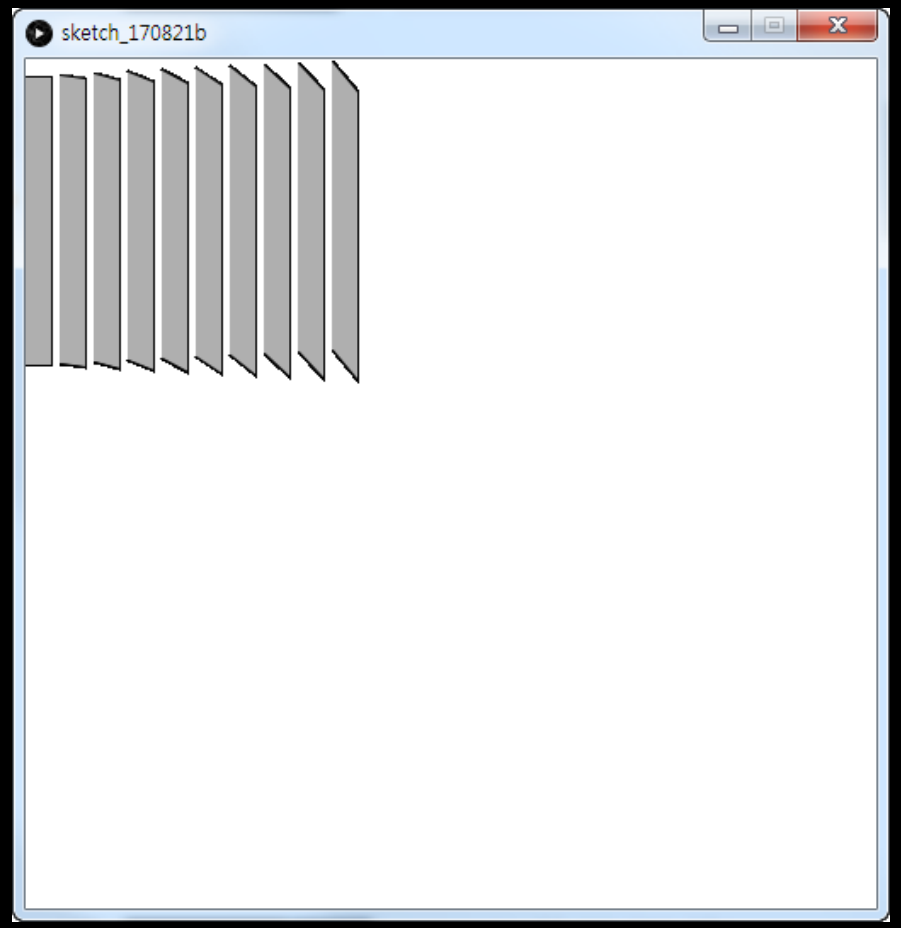
Vertex



Vertex Shapes in loops

```
stroke(0);
for (int i = 0; i < 10; i++) {
  beginShape();
  fill(175);
  vertex(i*20,10-i);
  vertex(i*20 + 15,10 + i);
  vertex(i*20 + 15,180 + i);
  vertex(i*20,180-i);
  endShape(CLOSE);
}
```

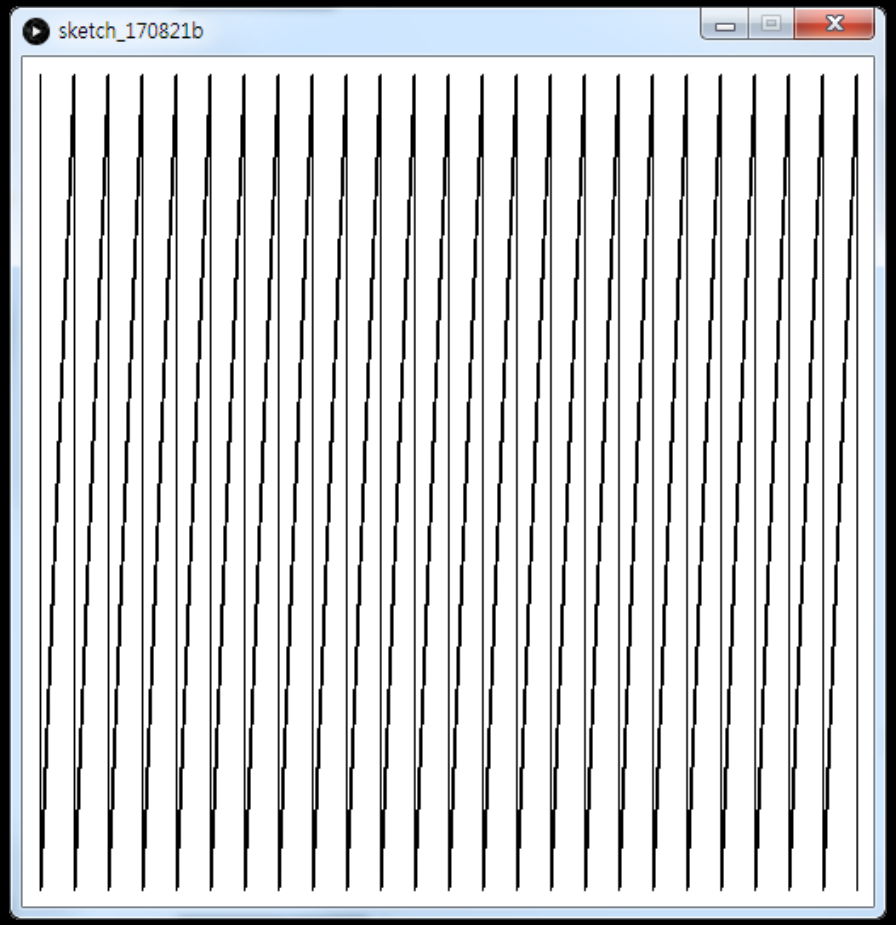
Vertex Shapes in loops



Continuous Shape

- noFill()
- beginShape before loop(no argument)
- endShape after loop

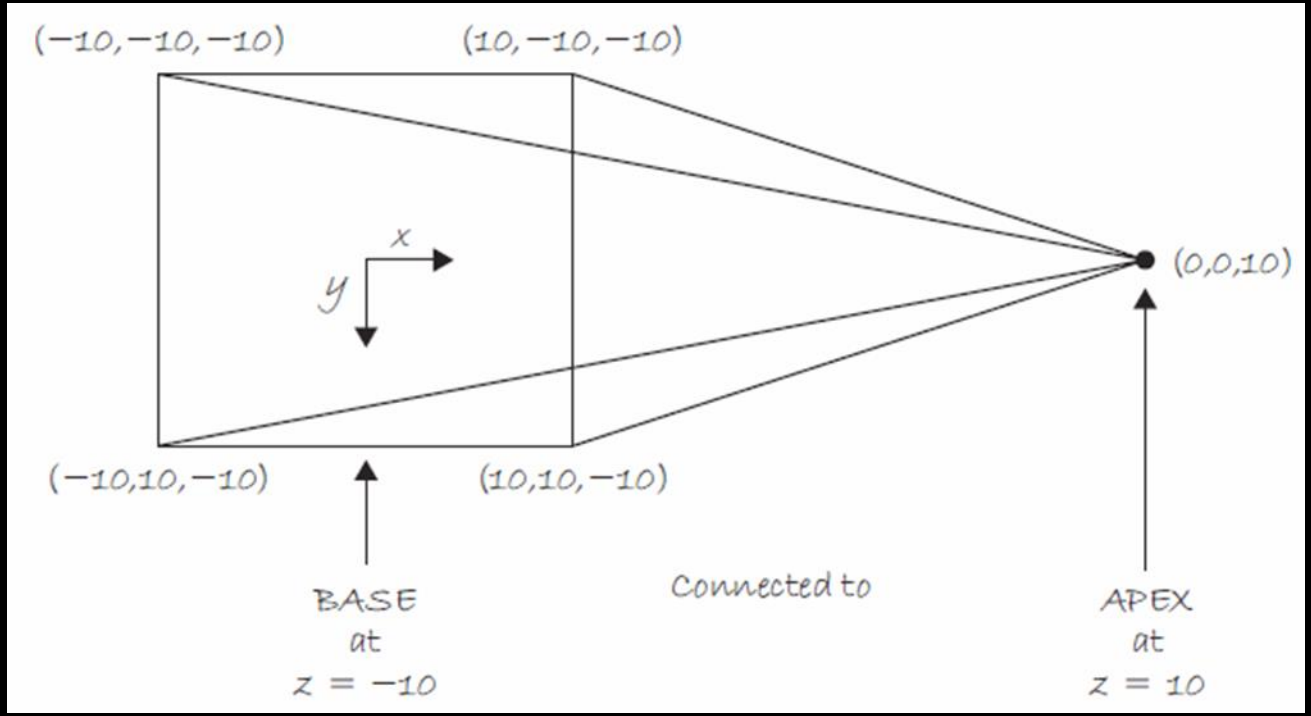
Continuous Shape



Continuous Shape

```
noFill();
stroke(0);
beginShape();  // No argument
for (int i = 10; i < width; i+= 20) {
  vertex(i,10);
  vertex(i,height–10);
}
endShape();    // No CLOSE
```

Custom 3D Vector Shapes



Pyramid

Four Triangles

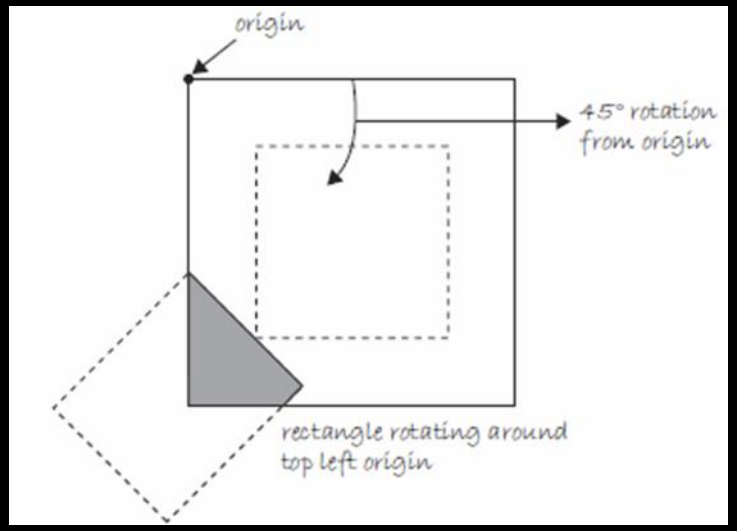
```
"Back"        x       y       z
vertex(–10,–10,–10);
vertex(–10,  10,–10);
vertex(   0,     0,   10);   // apex
"Top"
vertex(–10,–10,–10);
vertex(  10,–10,–10);
vertex(   0,     0,   10);   // apex
"Bottom"
vertex(–10,  10,–10);
vertex(  10,  10,–10);
vertex(   0,     0,   10);   // apex
"Right"
vertex( 10,–10,–10);
vertex( 10,  10,–10);
vertex(   0,    0,   10);   // apex
```

Simple Rotation

- 1. rotate() function
- 2. rotate() function takes one argument, an angle measured in radians.
- 3. rotate() will rotate the shaped in the clockwise direction.

Simple Rotation

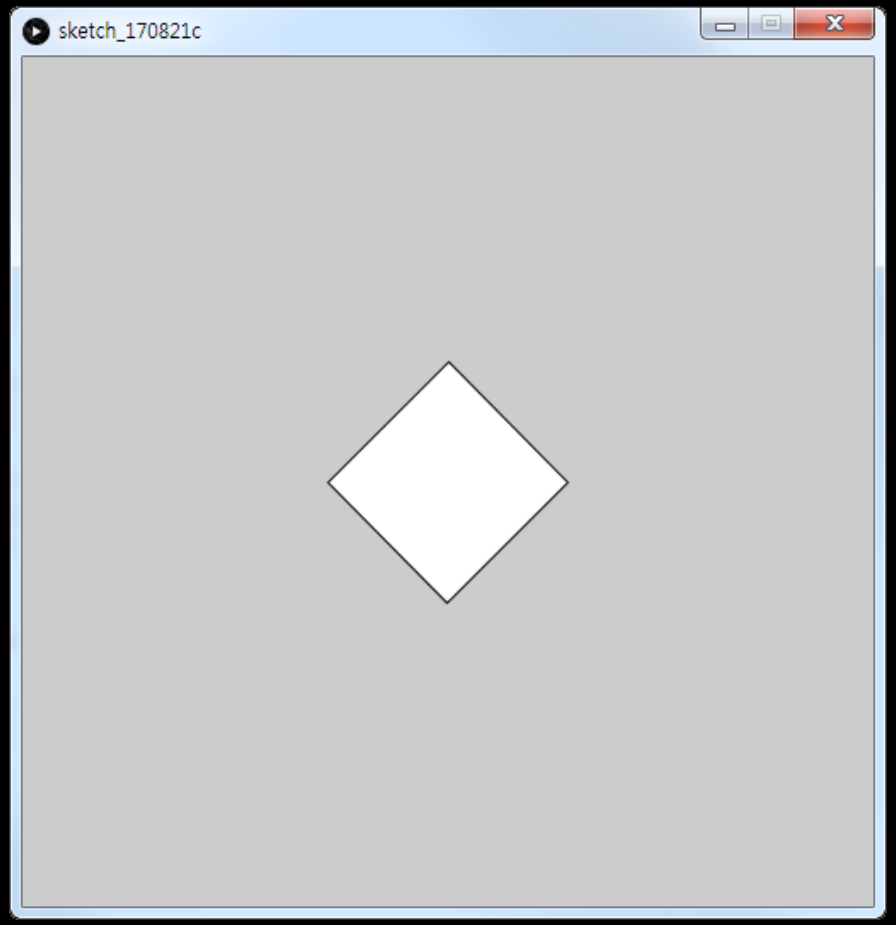
- rotate(radians(45));
- rectMode(CENTER);
- rect(width/2,height/2,100,100);



Rotating after translating

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

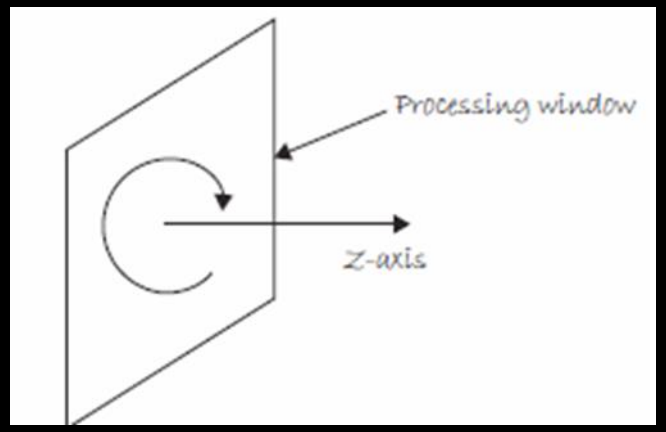
Rotating after translating



Mouse controlled rotation

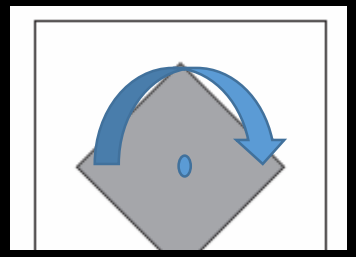
```
void setup() {
  size(200,200);
}
void draw() {
background(255);
stroke(0);
fill(175);
translate(width/2, height/2);
float theta = map(mouseX, 0, width, 0, TWO_PI);
rotate(theta);
rectMode(CENTER);
 rect(0, 0, 100, 100);
}
```

Rotation around different axes



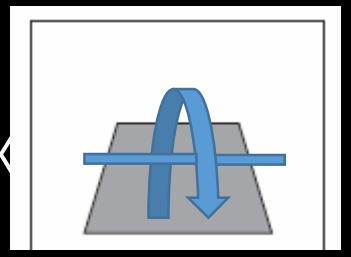
Rotation Axes

Z



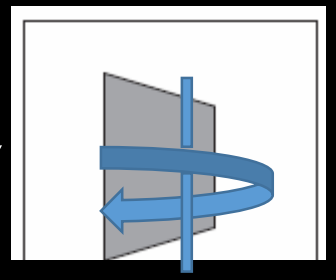
Around center

X



Flip (Rotisserie)

Y



Spin (like a top)

```
float theta = 0.0;
void setup() {
  size(200,200,P3D);
}
void draw() {
  background(255);
  stroke(0);
  fill(175);
  translate(width/2,
            height/2);
  rotateZ(theta);
  rectMode(CENTER);
  rect(0,0,100,100);
  theta + = 0.02;
}
```

```
float theta = 0.0;
void setup() {
  size(200,200,P3D);
}
void draw() {
  background(255);
  stroke(0);
  fill(175);
  translate(width/2,
            height/2);
  rotateX(theta);
  rectMode(CENTER);
  rect(0,0,100,100);
  theta + = 0.02;
}
```

```
float theta = 0.0;
void setup() {
  size(200,200,P3D);
}
void draw() {
  background(255);
  stroke(0);
  fill(175);
  translate(width/2,
            height/2);
  rotateY(theta);
  rectMode(CENTER);
  rect(0,0,100,100);
  theta + = 0.02;
}
```

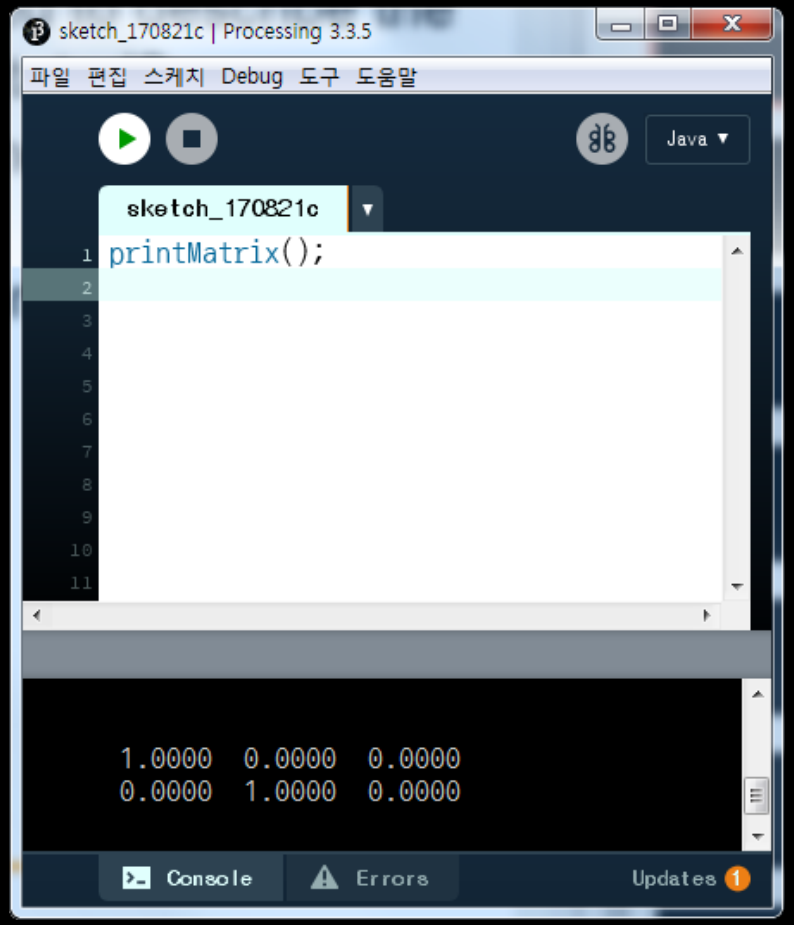
Rotation Around Multiple axes

```
void setup() {
  size(200,200,P3D);
}
void draw() {
  background(255);
  stroke(0);
  fill(175);
  translate(width/2,height/2);
  rotateX(PI*mouseY/height);
  rotateY(PI*mouseX/width);
  rectMode(CENTER);
  rect(0,0,100,100);
}
```

Matrix

- In order to keep track of rotations and translations and how to display the shapes according to different transformations, Processing uses a matrix.

Matrix


sketch_170821c | Processing 3.3.5
파일 편집 스케치 Debug 도구 도움말
Java
sketch_170821c
printMatrix();
1.0000 0.0000 0.0000
0.0000 1.0000 0.0000
Console
Errors
Updates 1

ModelX, Y, Z

```
void setup() {
  size(500, 500, P3D);
  noFill();
}

void draw() {
  background(0);

  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);

  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

Model X, Y, Z

sketch_170822a | Processing 3.3.5

파일 편집 스케치 Debug 도구 도움말

Java

sketch_170822a

```
void setup() {
  size(500, 500, P3D);
  noFill();
}

void draw() {
  background(0);

  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);

  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

```
x : 0.0, y : 0.0, z : 0.0
x : 0.0, y : 0.0, z : 0.0
x : 0.0, y : 0.0, z : 0.0
x : 0.0, y : 0.0, z : 0.0
x : 0.0, y : 0.0, z : 0.0
```

Console Errors Updates

Model X, Y, Z

```
void setup() {
  size(500, 500, P3D);
  noFill();
}
void draw() {
  background(0);
  translate(50, 100, 50);
  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);

  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

Model X, Y, Z

sketch_170822a | Processing 3.3.5

파일 편집 스케치 Debug 도구 도움말

Java

sketch_170822a

```
void setup() {
  size(500, 500, P3D);
  noFill();
}

void draw() {
  background(0);

  translate(50, 100, 50);

  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);

  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

```
x : 50.0, y : 100.0, z : 50.0
x : 50.0, y : 100.0, z : 50.0
x : 50.0, y : 100.0, z : 50.0
x : 50.0, y : 100.0, z : 50.0
x : 50.0, y : 100.0, z : 50.0
```

Console Errors Updates 1

Model X, Y, Z

```
void setup() {
  size(500, 500, P3D);  noFill();
}
void draw() {
  background(0);
  translate(50, 100, 50);
  rotateX(PI);
  translate(1, 2, 3);
  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);
  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

Model X, Y, Z

sketch_170822a | Processing 3.3.5

파일 편집 스케치 Debug 도구 도움말

Java

sketch_170822a

```
void setup() {
  size(500, 500, P3D);
  noFill();
}

void draw() {
  background(0);

  translate(50, 100, 50);
  rotateX(PI);
  translate(1, 2, 3);

  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);

  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

```
x : 51.0, y : 98.0, z : 47.0
x : 51.0, y : 98.0, z : 47.0
x : 51.0, y : 98.0, z : 47.0
x : 51.0, y : 98.0, z : 47.0
x : 51.0, y : 98.0, z : 47.0
```

Console Errors Updates 1

pushMatrix, popMatrix

```
void setup() {
  size(500, 500, P3D);
  noFill();
}

void draw() {
  background(0);

  pushMatrix();
    translate(50, 100, 50);
    rotateX(PI);
```

pushMatrix, popMatrix

```
    translate(1, 2, 3);
  popMatrix();

  translate(10,10,10);

  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);

  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

pushMatrix, popMatrix

sketch_170822a | Processing 3.3.5

파일 편집 스케치 Debug 도구 도움말

Java

sketch_170822a

```
void setup() {
  size(500, 500, P3D);
  noFill();
}

void draw() {
  background(0);

  pushMatrix();
    translate(50, 100, 50);
    rotateX(PI);
    translate(1, 2, 3);
  popMatrix();

  translate(10,10,10);

  float x = modelX(0, 0, 0);
  float y = modelY(0, 0, 0);
  float z = modelZ(0, 0, 0);

  println("x : " + x + ", y : " + y + ", z : " + z);
}
```

```
x : 10.0, y : 10.0, z : 10.0
x : 10.0, y : 10.0, z : 10.0
x : 10.0, y : 10.0, z : 10.0
x : 10.0, y : 10.0, z : 10.0
x : 10.0, y : 10.0, z : 10.0
```

Console Errors Updates 1

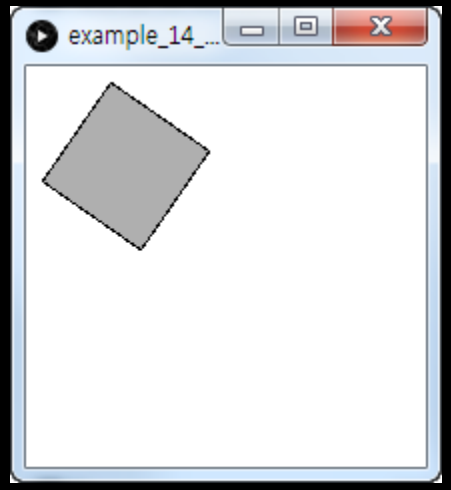
Rotating one Square

```
float theta1 = 0;

void setup() {
  size(200, 200, P3D);
}

void draw() {
  background(255);
  stroke(0);
```

Rotating one Square

```
  fill(175);
  rectMode(CENTER);

  translate(50, 50);
  rotateZ(theta1);
  rect(0, 0, 60, 60);

  theta1 += 0.02;
}
```

RotateZ



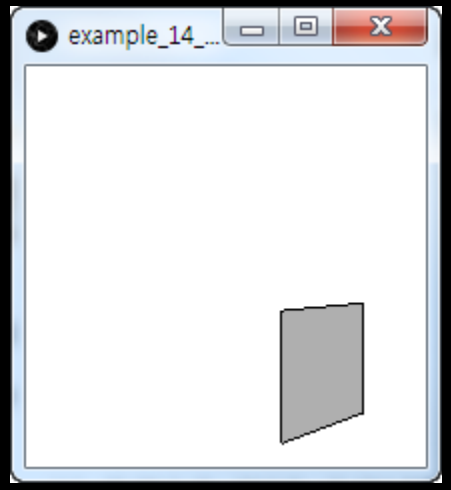
Rotating another square

```
float theta2 = 0;

void setup() {
  size(200, 200, P3D);
}

void draw() {

  background(255);
```

Rotating another square

```
  stroke(0);
  fill(175);
  rectMode(CENTER);

  translate(150, 150);
  rotateY(theta2);
  rect(0, 0, 60, 60);

  theta2 += 0.02;
}
```

RotateY



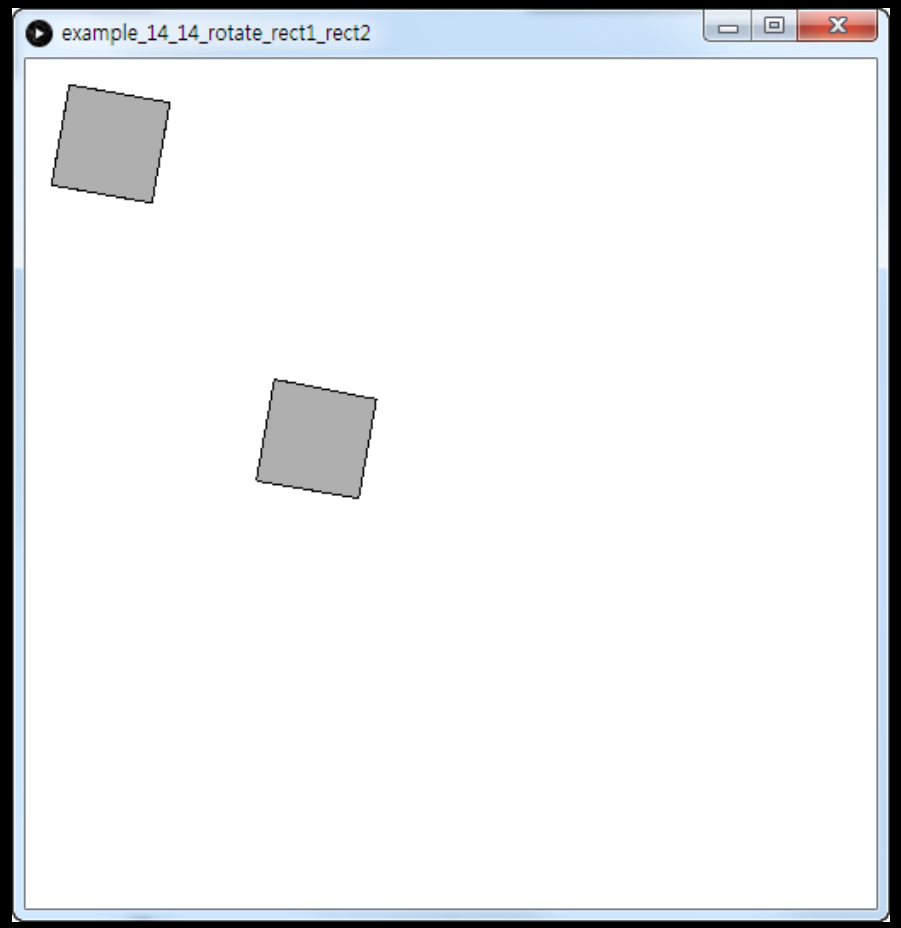
Combine

```
float theta1 = 0;
float theta2 = 0;

void setup() {
  size(500, 500, P3D);
}

void draw() {
  background(255);
  stroke(0);
  fill(175);
  rectMode(CENTER);
```

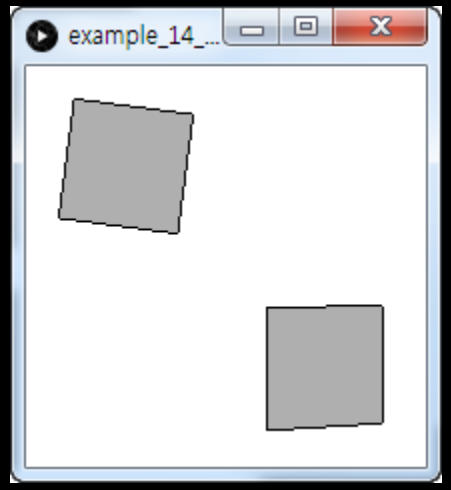
Combine

```
  translate(50, 50);
  rotateZ(theta1);
  rect(0, 0, 60, 60);

  translate(150, 150);
  rotateY(theta2);
  rect(0, 0, 60, 60);

  theta1 += 0.02;
  theta2 += 0.02;
}
```

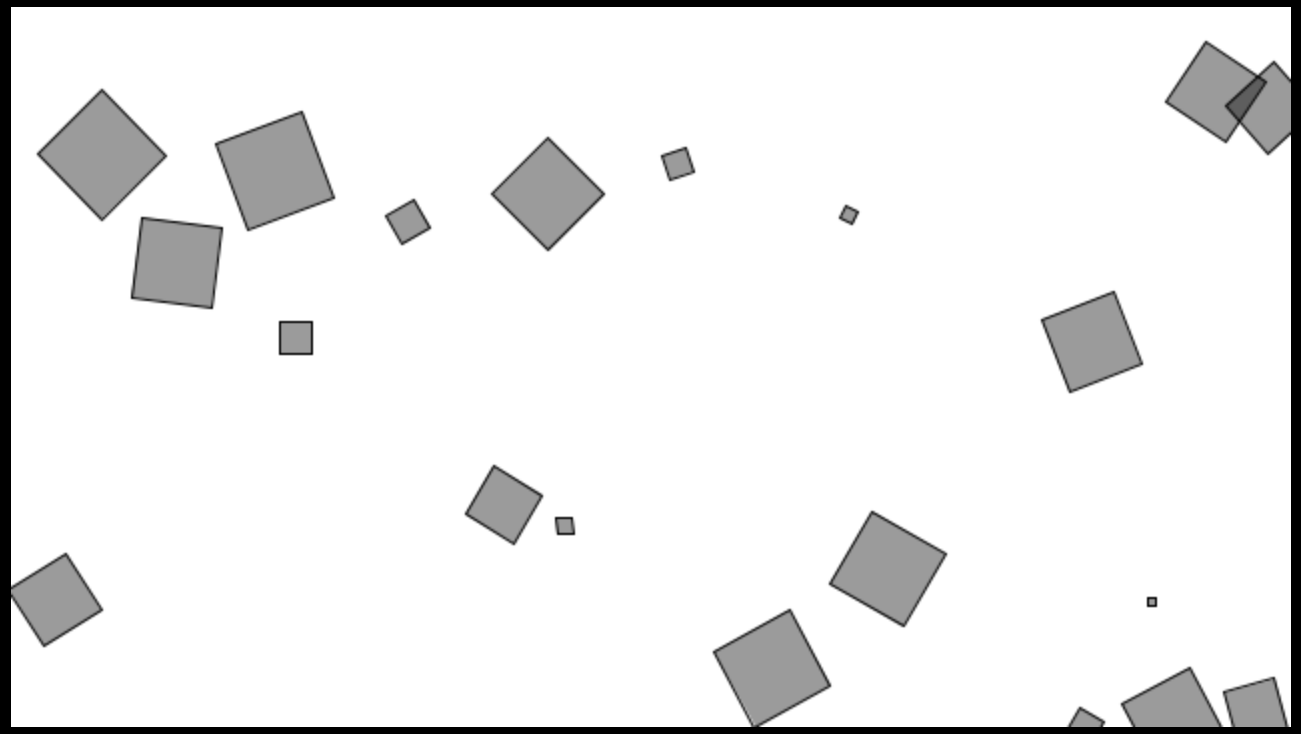
Combine



Your turn



Spinning objects



Rotater

```
class Rotater {

  float x, y;    // x,y location
  float theta;  // angle of rotation
  float speed;  // speed of rotation
  float w;       // size of rectangle

  Rotater(float tempX, float tempY, float tempSpeed, float
tempW) {
```

Rotater

```
    x = tempX;
    y = tempY;
    // Angle is always initialized to 0
    theta = 0;
    speed = tempSpeed;
    w = tempW;
  }

  // Increment angle
```

Rotater

```
void spin() {
  theta += speed;
}

// Display rectangle
void display() {
  rectMode(CENTER);
  stroke(0);
  fill(0, 100);
```

Rotater

```
    pushMatrix();
    translate(x, y);
    rotate(theta);
    rect(0, 0, w, w);
    popMatrix();
  }
}
```

Spinning objects

```
Rotater[] rotaters;

void setup() {
  size(640, 360);

  rotaters = new Rotater[20];

  for (int i = 0; i < rotaters.length; i++ ) {
    rotaters[i] = new Rotater(random(width), random(height), random(-0.1, 0.1), random(48));
  }
```

Spinning objects

```
}

void draw() {
  background(255);

for (int i = 0; i < rotaters.length; i++ ) {
    rotaters[i].spin();
    rotaters[i].display();
  }
}
```

Spinner

PSpinner

Spinner

```
float x0,   y0,
      x1,   y1,
      x2,   y2,
      ang,  freq,
      r1,   r2;
final float rad = TWO_PI/3;

void setup() {
  size(500,500);
  r1 = 140;
  r2 = 120;
  ang = freq = 0;
  stroke(#eeeeee);
}
```

Spinner

```
void draw() {
  background(#222222);
  fill(#eeeeee);
  text("Frequency: " + nf(freq, 1, 3), 20, 20);
  translate(mouseX, mouseY);

  x0 = r1 * cos(ang);
  y0 = r1 * sin(ang);
```

Spinner

```
ang += freq;
if (freq > 0) freq -= 0.0005;
else freq = 0;

x1 = r1 * cos(ang + rad);
y1 = r1 * sin(ang + rad);

x2 = r1 * cos(ang + rad * 2);
```

Spinner

```
y2 = r1 * sin(ang + rad * 2);

fill(#222222);
strokeWeight(100);
line(0, 0, x0, y0);
line(0, 0, x1, y1);
line(0, 0, x2, y2);

strokeWeight(35);
```

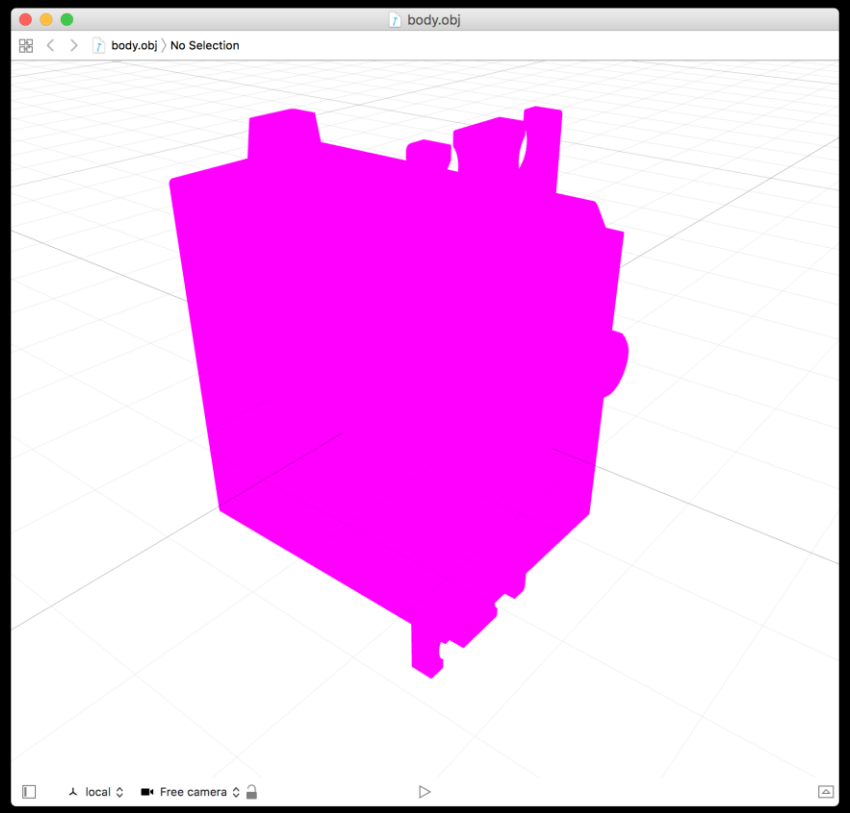
Spinner

```
  ellipse(0, 0, r2, r2);
  ellipse(x0, y0, r2, r2);
  ellipse(x1, y1, r2, r2);
  ellipse(x2, y2, r2, r2);
}

void mouseClicked() {
  freq = 0.6;
}
```

Obj file

- The OBJ file format is a simple data-format that represents 3D geometry alone — namely, the position of each vertex, the UV position of each texture coordinate vertex, vertex normals, and the faces that make each polygon defined as a list of vertices, and texture vertices.

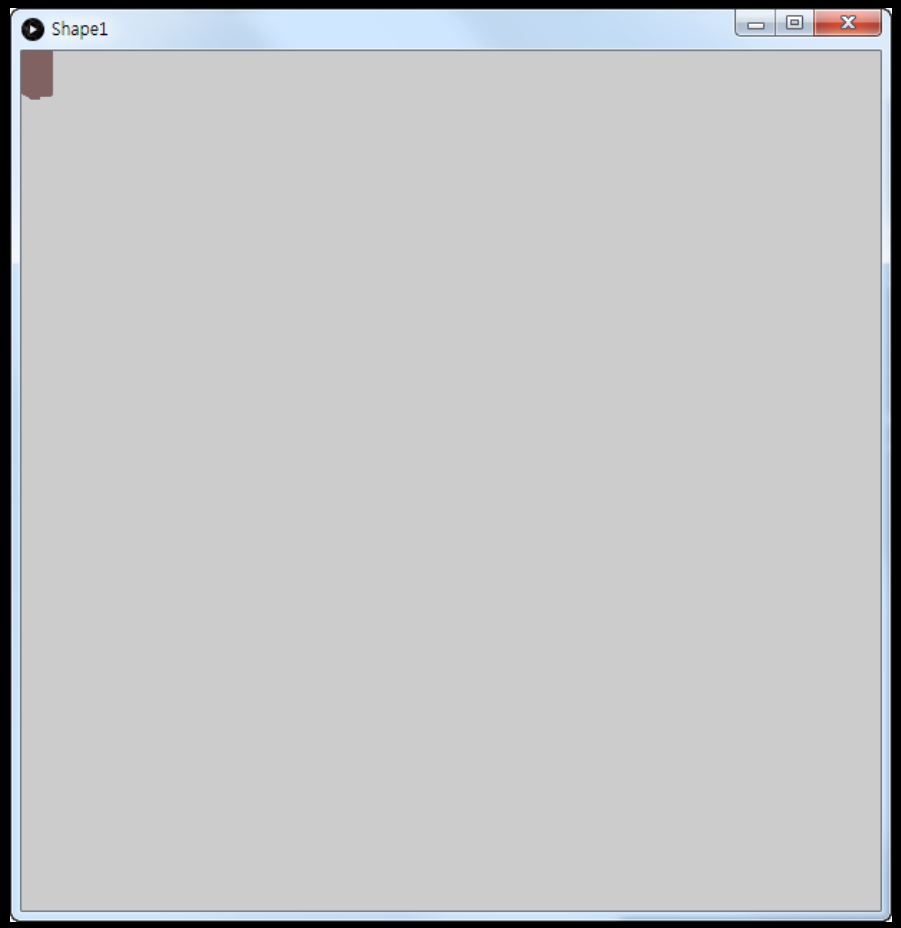
Servo Body



Servo Shape

```
PShape body;

void setup() {
  size(600,600,P3D);
  smooth();

  body = loadShape("body.obj");
}

void draw() {

  shape(body);

}
```

But…



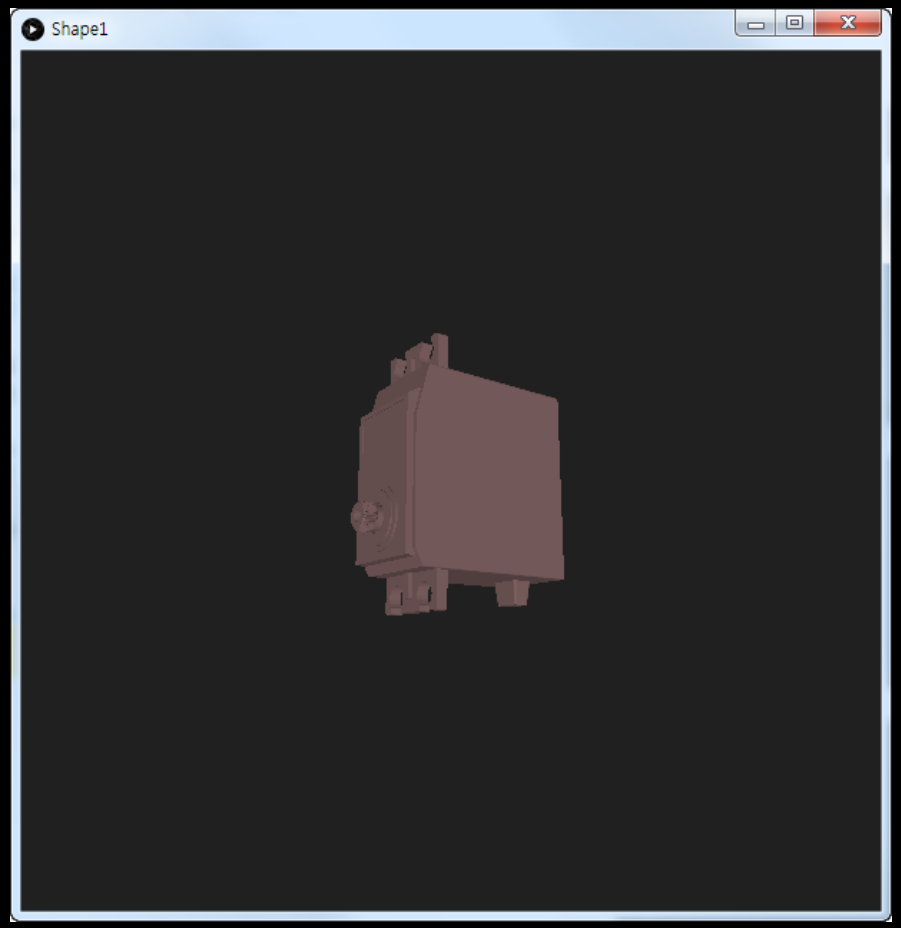
Servo body

```
PShape body;

float rotX, rotY;

void setup() {
  size(600,600,P3D);
  smooth();

  body = loadShape("body.obj");
}

void draw() {
  lights();
  background(32);
```

Servo Body

```
  translate(width/2, height/2, -100);

  rotateX(rotX);
  rotateY(-rotY);


  scale(4);
  shape(body);

}

void mouseDragged() {
    rotY -= (mouseX - pmouseX) * 0.01;
    rotX -= (mouseY - pmouseY) * 0.01;
}
```

Servo body



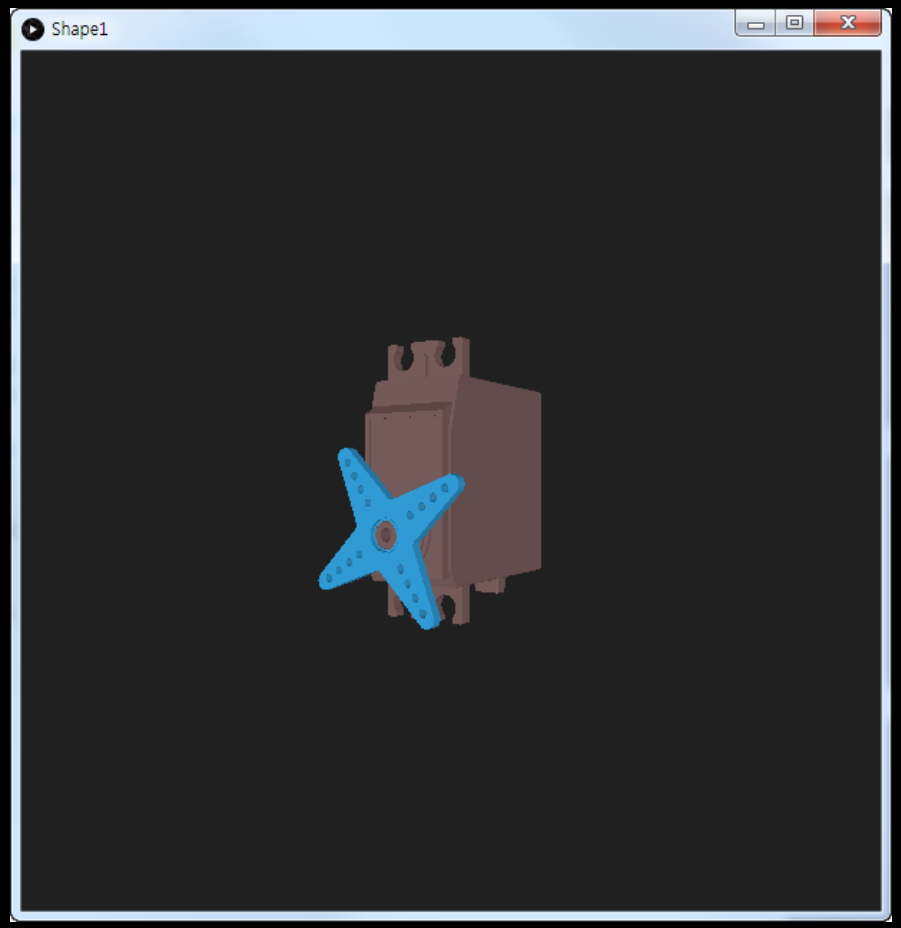
Together

```
PShape body, horn;

float rotX, rotY;

void setup() {
  size(600,600,P3D);
  smooth();

  body = loadShape("body.obj");
  horn = loadShape("tmp.obj");

}

void draw() {
  lights();
  background(32);
```

Together

```
  translate(width/2, height/2, -100);

  rotateX(rotX);
  rotateY(-rotY);



  scale(4);
  shape(body);

  translate(0, 10, 20);
  rotateZ(radians(frameCount));
  shape(horn);
}

void mouseDragged() {
    rotY -= (mouseX - pmouseX) * 0.01;
    rotX -= (mouseY - pmouseY) * 0.01;
}
```

Together



Break…

- CU..