Artwork: Aaron Koblin

Transformation Functions

VA345 Creative Coding

Course Instructor : Assist. Prof. Dr. Selcuk ARTUT

Food for thought : Aaron Koblin

http://www.aaronkoblin.com/

saveFrame()

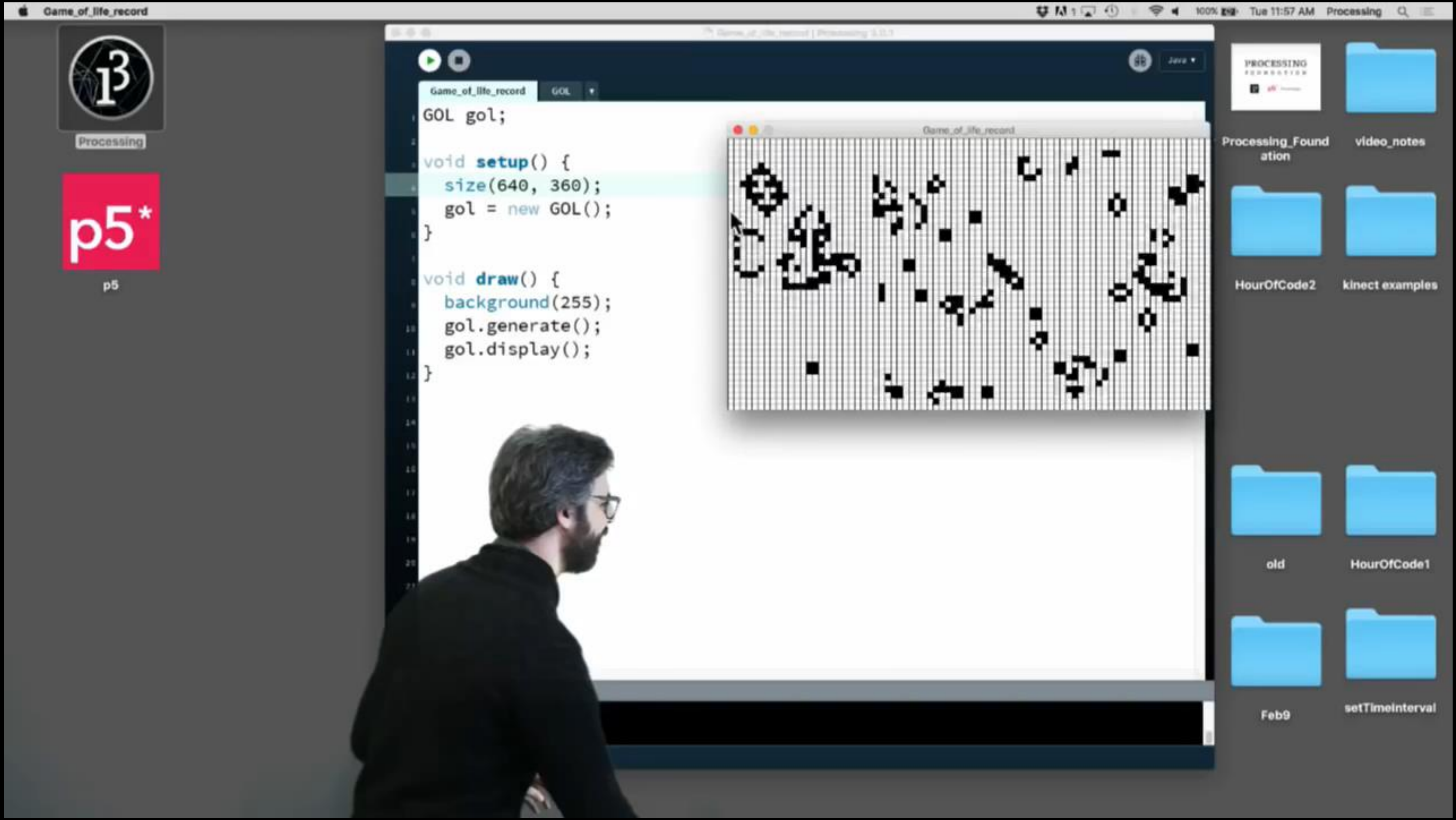
Saves a numbered sequence of images, one image each time the function is run.

Syntax

saveFrame()

saveFrame(filename)

Please save a frame from your Assignment 02 and send via email to sartut@sabanciuniv.edu https://www.youtube.com/watch?v=G2hI9XL6oyk



```
GOL gol;

void setup() {
  size(640, 360);
  gol = new GOL();
}

void draw() {
  background(255);
  gol.generate();
  gol.display();
}
```

Translate

The translate() function moves the origin from the upper-left corner of the screen to another location. It has two parameters. The first is the x-coordinate offset and the second is the y-coordinate offset:

translate(x, y)

The values of the x and y parameters are added to any shapes drawn after the function is run.

Translate

Code Examples (Translate Example 1)

Let’s try

```
rect(0, 5, 70, 30);
translate(10, 30); // Shifts 10 pixels right and 30 down
rect(0, 5, 70, 30);
```

Translate

Code Examples (Translate Example 2)

vs

```
translate(10, 30); // Shifts 10 pixels right and 30 down
rect(0, 5, 70, 30);
rect(0, 5, 70, 30);
```

Translate

The translate() function is additive. If translate(10,30) is run twice, all the elements drawn after will display with an x-offset of 20 and a y-offset of 60 (Translate Example 3)

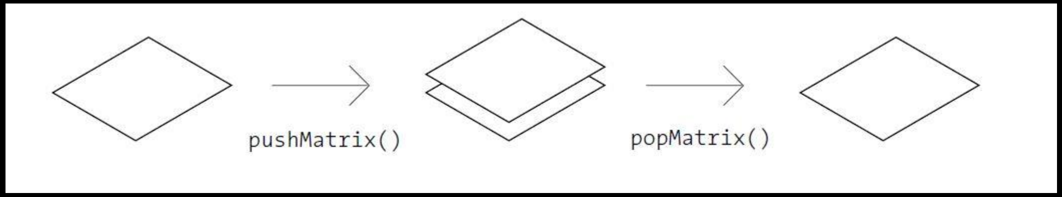
```
rect(0, 5, 70, 30);
translate(10, 30); // Shifts 10 pixels right and 30 down
rect(0, 5, 70, 30);
translate(10, 30); // Shifts everything again for a total
rect(0, 5, 70, 30); // 20 pixels right and 60 down
```

Translate

Controlling transformations

The transformation matrix is a set of numbers that defines how geometry is drawn to the screen. Transformation functions such as translate() alter the numbers in this matrix and cause the geometry to draw differently.

The pushMatrix() function records the current state of all transformations so that a program can return to it later. To return to the previous state, use popMatrix().



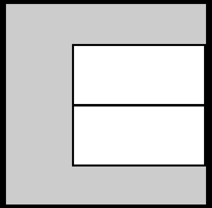
Each pushMatrix() must have a corresponding popMatrix(). The function pushMatrix() cannot be used without popMatrix(), and vice versa.

Translate

Using pushMatrix() and popMatrix()

Example 4

```
translate(33, 0); // Shift 33 pixels right
rect(0, 20, 66, 30);
rect(0, 50, 66, 30);
```

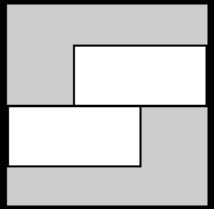


Translate

Using pushMatrix() and popMatrix()

Example 5

```
pushMatrix();
translate(33, 0); // Shift 33 pixels right
rect(0, 20, 66, 30);
popMatrix(); // Remove the shift
// This shape is not affected by translate() because
// the transformation is isolated between the pushMatrix()
// and popMatrix()
rect(0, 50, 66, 30);
```



Translate

Rotation

The rotate() function rotates the coordinate system so that shapes can be drawn to the screen at an angle. It has one parameter that sets the amount of the rotation as an angle:

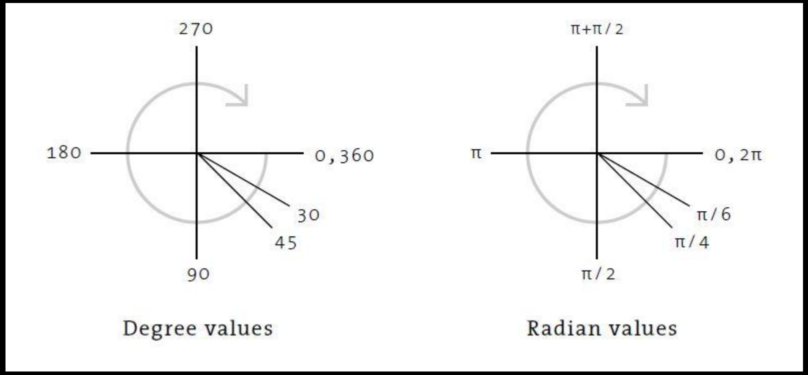
rotate(angle)

The rotate function assumes that the angle is specified in units of radians

Radian? Uh?

Degrees are a common way to measure angles. A right angle is 90°, halfway around a circle is 180°, and the full circle is 360°.

In working with trigonometry, angles are measured in units called radians. Using radians, the angle values are expressed in relation to the mathematical value π, written in Latin characters as “pi” and pronounced “pie.” In terms of radians, a right angle is π/2, halfway around a circle is simply π, and the full circle is 2π.

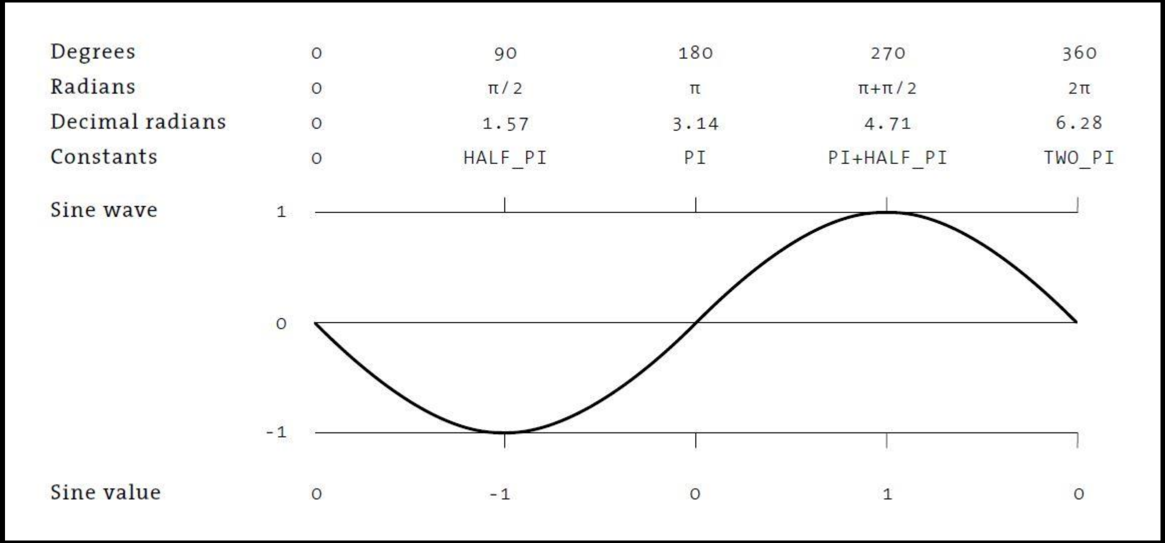


Trigonometry? Oh no!

The sin() and cos() functions are used to determine the sine and cosine value of any angle. Each of these functions requires one parameter:

sin(angle)

cos(angle)



Degrees	0	90	180	270	360
Radians	0	π/2	π	π+π/2	2π
Decimal radians	0	1.57	3.14	4.71	6.28
Constants	0	HALF_PI	PI	PI+HALF_PI	TWO_PI
Sine value	0	-1	0	1	0

Let’s code an example

Make a shape move with numbers returned from sin() and cos().

Translate

Rotation Example 1

```
smooth();
rect(55, 0, 30, 45);
rotate(PI/8);
rect(55, 0, 30, 45);
```

Translate

Scaling

The scale() function magnifies the coordinate system so that shapes are drawn larger. It has one or two parameters to set the amount of increase or decrease:

scale(size)

scale(xsize, ysize)

Scaling Example 1 :

```
smooth();
ellipse(32, 32, 30, 30);
scale(1.8);
ellipse(32, 32, 30, 30);
```

Free Coding Time (40 minutes)

Use Loop iterations, Transformation Functions to create Rich Geometries

Assignment 004

Use Rotation, Transformation, Randomness, Noise etc to create animating geometric patterns.