Using **ArbitraryShape.js:**

Put this line in your **index.html** file, alongside the other scripts:

<script src="ArbitraryShape.js"></script>

This line tells the web browser, “Hey, there’s another file you need to get code from.” It lets your **sketch.js** access the code from **ArbitraryShape.js**In order for this to work, **ArbitraryShape.js** must be in the same folder as **index.html**

Put this line at the very top of your **sketch.js** file:

var myShape;

This line determines what your shape will be called later in the code. You can call it something other than ‘myShape,’ but you must include this line. Repeat this line (with a different name) for each shape you plan to use in your sketch.

Use this line in your **setup()** function to create the ArbitraryShape ‘object’:

myShape = new ArbitraryShape(x, y, size, theta, r, g, b, alpha);

This line determines so properties of your shape:  
X and Y are the coordinate of your shape’s position on the screen. The shape will also rotate around this point. This is called the origin of your shape.  
‘Size’ is used to scale the shape. Ninety-nine percent of the time, you should set this to 1.  
‘Theta’ is the angle of your shape. It’s best to leave it at zero here, and then change it later as needed.   
R, G, and B determine the color of your shape, while alpha determines its opacity.

Use this line in your **setup()** function to define the form of your ArbitraryShape:

myShape.addPoint(x, y);

This line must be used once for each corner of your shape.  
You can work either clockwise or counter-clockwise, but pick one and stick with it.  
Position of these points is relative to your shape, not to the entire sketch.  
Assuming the rotation of your shape is zero:  
 Positive X values are to the right of the ‘origin’ of your shape.  
 Negative X values are to the left of the ‘origin’ of your shape.  
 Positive Y values are below the ‘origin’ of your shape.  
 Negative Y values are above the ‘origin’ of your shape.  
 A value of (0, 0) will place a point directly on the ‘origin’ of your shape.

Use this line in your **draw()** function to draw your ArbitraryShape to the screen:

myShape.draw();

There is no need to set fill color before making this call, as that is handled by code inside the shape. Each ArbitraryShape objects maintains its own color.

Use this line in your **draw()** function to rotate your ArbitraryShape:

myShape.rotate(theta);

This line adds theta, a number expressed in radians, to the angle of your ArbitraryShape.  
If you would rather use degrees for rotation, the conversion is (*degrees\*pi/180 = radians*)*.*  
A positive value of theta rotates your ArbitraryShape counter-clockwise; negative, clockwise. To set Theta to a new value directly, rather than adding or subtracting a value, use the function myShape.setTheta(newTheta).

Use this line in your **draw()** function to move your ArbitraryShape:

myShape.move(x, y);

This line adds X and Y to your ArbitraryShape’s absolute position.  
Remember that positive Y values will move your shape down the screen, negative Y values the opposite. This movement does not account for the shape’s current orientation.  
To move your shape in the direction it is pointing, use the function myShape.moveAngle(velocity), with velocity being the distance you would like the shape to move.  
To totally relocate your shape, rather than moving it incrementally, use the function myShape.relocate(x, y), with X and Y behaving the same as in move().

Use this line in your **draw()** function to change the size of your ArbitraryShape:

myShape.resize(size);

This line sets the size of your ArbitraryShape to whatever value you give it.  
If you set up your object right, a size value of 1 will reset it to its default size.  
A value of 2 will double the size; a value of 0.5 will halve the size, etc.  
If you set up your shape with a default size value other than 1, you’re on your own.  
To change the size of your shape incrementally, use the function myShape.grow(dSize), where dSize is the amount by which the shape’s size should change. A negative value for dSize causes the shape to shrink.

Use this line anywhere in your sketch to change the color of your ArbitraryShape:

myShape.setColor(newRed, newGreen, newBlue, newAlpha);

This line sets the color of your ArbitraryShape to the color specified. The variables newRed, newGreen, and newBlue comprise an RGB value while newAlpha sets the shape’s opacity. The function myShape.setAlpha(newAlpha)behaves similarly but only affects opacity.

Sample Program:

var myStar;

function setup() {

//put setup code here

createCanvas(500, 300);

//creates the shape object (you must have declared it at the top of your program)

myStar = new ArbitraryShape(10, 10, 1, 0, 255, 255, 0);

//Position is (10, 10)

//Shape is at default size (size = 1)

//Shape is 'upright' (theta = 0)

//Shape is yellow (RGB = 255, 255, 0)

//right tip

myStar.addPoint(50, 0);

myStar.addPoint(10, 10);

//bottom tip

myStar.addPoint(0, 50);

myStar.addPoint(-10, 10);

//left tip

myStar.addPoint(-50, 0);

myStar.addPoint(-10, -10);

//upper tip

myStar.addPoint(0, -50);

myStar.addPoint(10, -10);

}

function draw() {

clear();

noStroke();

//this draws the shape

myStar.draw();

//this moves the shape down and to the right each frame

myStar.move(1, 1);

//this rotates the shape one tenth of a radian counterclockwise

myStar.rotate(0.1);

//this sets the shape's size multiplier to 2

myStar.resize(2);

}

Type this in and mess around to see how it works.