Rotation and translation

This lesson introduces two new functions, rotate() and translate(). You can use the reference guide to find out more about these functions.

An important thing to note when using rotate() is that the value in the brackets has to be in radians. In radians two PI equals one full rotation back to the start. So some common numbers to use are PI, PI/2 or PI/4.

Tasks

- Type out the code below and run it. Remove the comment and run it again, what changed? Now change the value in the rotate function, what happens.

```
void setup() {
  size(600,600);
}

void draw() {
  //rotate(PI/4);
  rect(0,0,150,50);
}
```

- Rotating also has to be done about a point or what is called the origin. The default point for this being (0,0) or the top left corner as we saw from the code above. We can change where the shape rotates about by using the translate function. The code in the box below will make the rectangle rotate about the center of the screen, again change the numbers to see what happens.

```
void setup() {
  size(600,600);
}

void draw() {
  translate(300,300);
  ellipse(0,0,100,100);
  rotate(PI/4);
  rect(0,0,150,50);
}
```

- Create an image with at least 2 rotated shapes. Be careful here as translate and rotate stack, meaning having rotate(PI/4) twice is the same as having rotate(PI/2). The same is true of translate. Maybe make a snowflake or other image using rotated rectangles.
- Being able to rotate about one point is nice but when you have more than one shape on the screen it can get messy or impossible to rotate it correctly. To fix this we use two new functions, pushMatrix() and popMatrix(). If we put pushMatrix() at the start and popMatrix() at the end of what we want to rotate or translate they will only affect the shapes in between pushMatrix() and popMatrix(). An example of this is below.

```
void draw() {
  pushMatrix();
  translate(300, 300);
  rotate(PI/4);
  rect(0, 0, 150, 50);
  popMatrix();

  pushMatrix();
  translate(500, 400);
  rotate(3*PI/4);
  rect(0, 0, 150, 50);
  popMatrix();
}
```

- Using these functions we can do many new things. The last task is to use all of these functions to create something. Remember you can still use variables in these functions, so having something moving in circles is possible. If you can't think of anything else try making a moving analog clock like the one below.