Transformations

It is Often Nice to Transform Entire Objects at Once

Translation

Rotations and Scaling Happen Around the Origin
In math, science, and computer programming, angles are not given in degrees, they are given in radians.

1 radian = 0.01745 degrees
1 radian = π/180. degrees

But, don’t worry about this.

Processing gives you a function, `radians()`, to automatically convert degrees into radians.

Use it!

There is also a `shearX` transformation function

```
void draw() {
    shearX( radians(45. ) );
    rect( 0, 0, 100, 50 );
}
```

```
void draw() {
    rotate( radians(10. ) );
    rotate( radians(10. ) );
    . . .
}
```

is the same as:

```
void draw() {
    rotate( radians(20. ) );
    . . .
}
```
Transformation Order Matters!

```cpp
void draw()
{
  2. translate( 100, 200 );
  1. rotate( radians(20.) );
  rect( 0, 0, 100, 50 );
}
```

You Can Save and Un-do Transformations

```cpp
void draw()
{
  2. translate( 200, 300 );
  pushMatrix();
  1. shearX( radians(45.) );
  rect( 0, 0, 100, 100 );
  popMatrix();
  fill( 255, 0, 0 );
  1. rotate( radians( -45. ) );
  rect( 0, 0, 100, 100 );
}
```

Transformations and for-loops

```cpp
void draw()
{
  translate( 200, 200 );
  for( int degrees = 0; degrees < 360; degrees += degrees + 30 )
  {
    pushMatrix();
    rotate( radians( degrees ) );
    rect( 0, 0, 100, 100 );
    popMatrix();
  }
}
```

```
```
What's the Difference?

Transformations and for-loops

void draw()
{
    translate(200, 200);
    for( int degrees = 8; degrees <= 360; 
        pushMatrix();
            rotateX( radians );
        popMatrix();
    }