

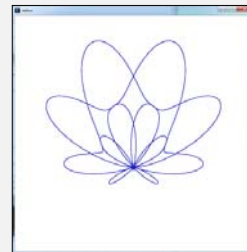
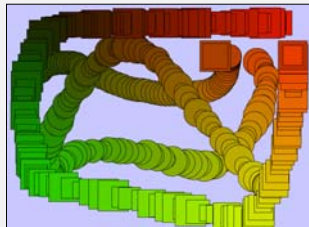
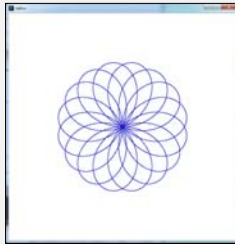
Transformations



Oregon State
University

Mike Bailey

mjb@cs.oregonstate.edu



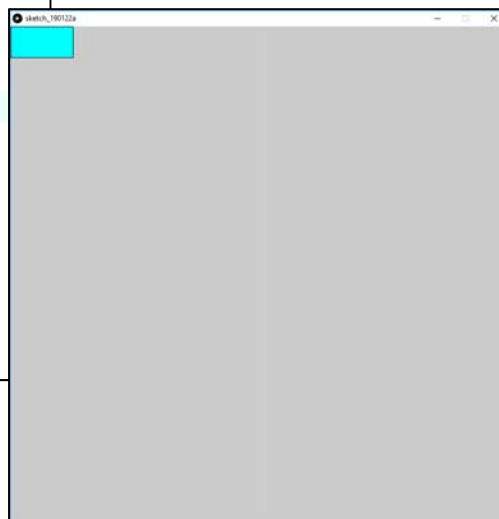
Oregon State
University
Computer Graphics

transformations.pptx

mjb - January 28, 2019

It is Often Nice to Transform Entire Objects at Once

```
void  
setup( )  
{  
  size( 800, 800 );  
  stroke( 0, 0, 0 );  
  fill( 0, 255, 255 );  
}  
  
void  
draw( )  
{  
  rect( 0, 0, 100, 50 );  
}
```



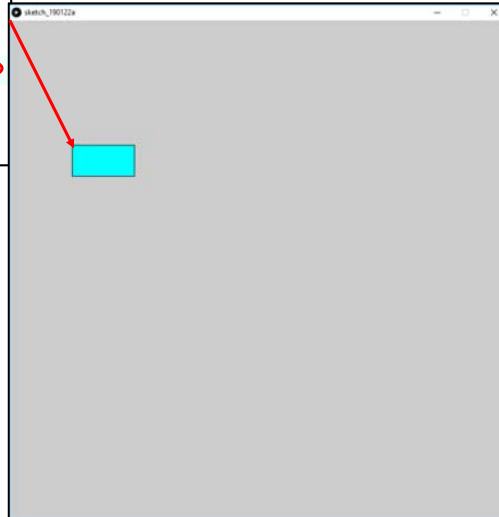
Oregon State
University
Computer Graphics

mjb - January 28, 2019

Translation

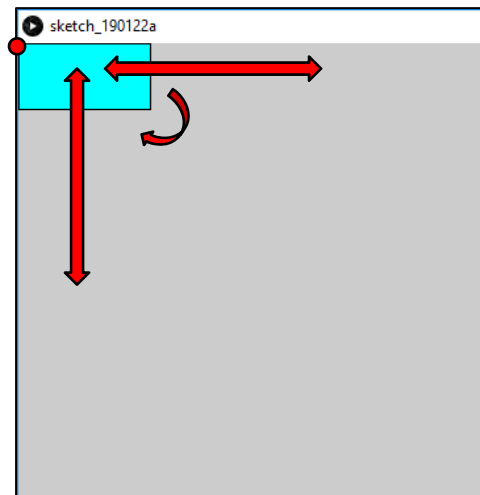
3

```
void  
draw( )  
{  
  translate( 100, 200 );  
  rect( 0, 0, 100, 50 );  
}
```



Rotations and Scaling Happen Around the Origin

4



Rotation

5

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

In math, science, and computer programming, angles are not given in degrees, they are given in *radians*.

1 radian = 0.01745 degrees

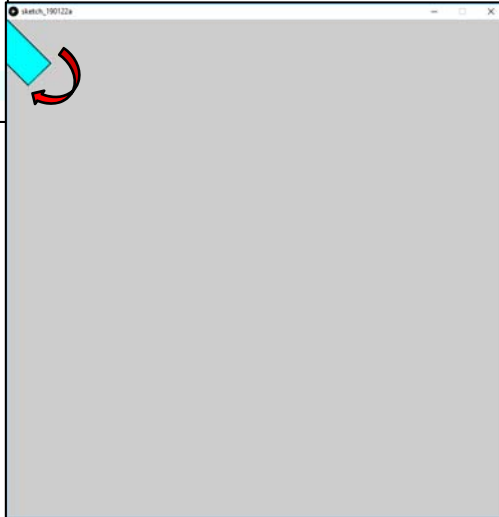
1 radian = $\pi/180$. degrees

But, don't worry about this.

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

Use it!

University
Computer Graphics

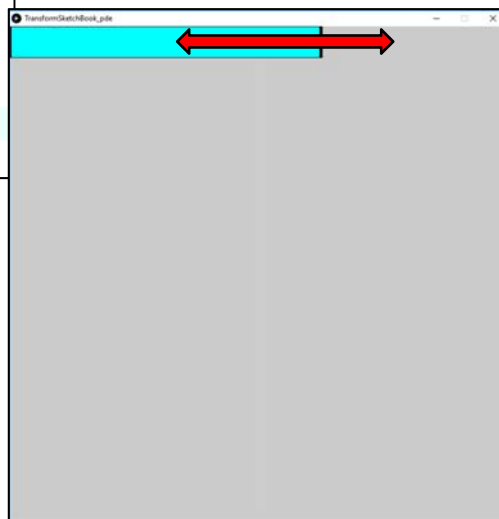


mjb - January 28, 2019

Scaling

6

```
void  
draw( )  
{  
  scale( 5., 1. );  
  rect( 0, 0, 100, 50 );  
}
```



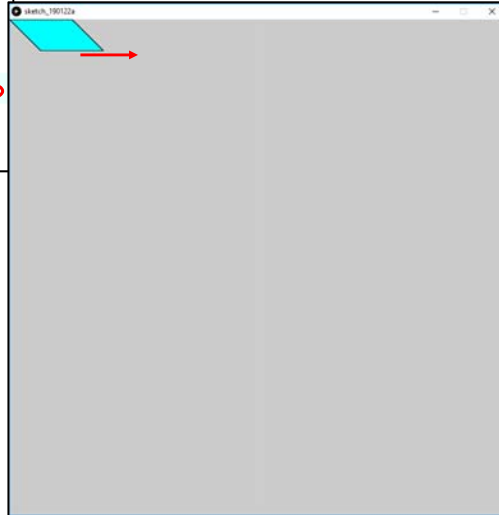
mjb - January 28, 2019

Shearing

7

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

There is also a **shearY**
transformation function



mjb - January 28, 2019

Transformations Accumulate!

8

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

is the same as:

```
void  
draw( )  
{  
    rotate( radians( 20. ) );  
    ...  
}
```

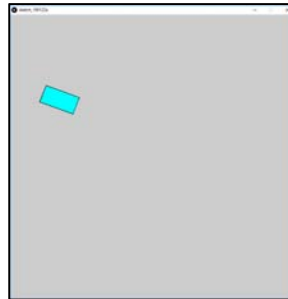


mjb - January 28, 2019

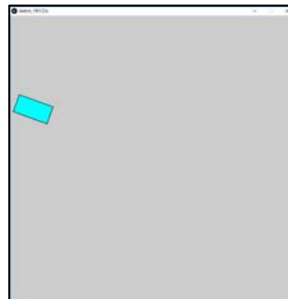
Transformation Order Matters!

9

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



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



Oregon State
University
Computer Graphics

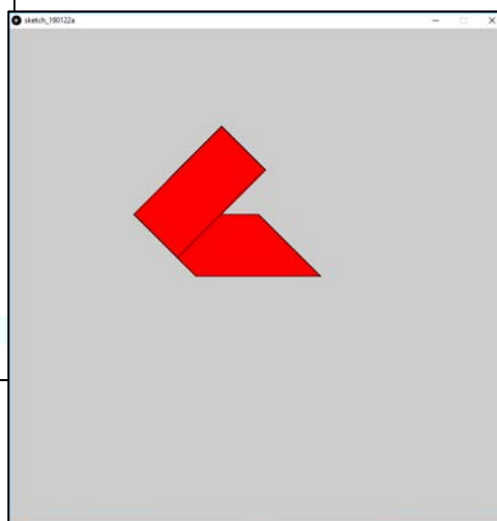
mjb - January 28, 2019

You Can Save and Un-do Transformations

10

```
void
draw( )
{
  2. translate( 200, 300 );
  pushMatrix( );
  1. shearX( radians(45.) );
  rect( 0, 0, 200, 100 );
  popMatrix( );

  fill( 255, 0, 0 );
  1. rotate( radians( -45. ) );
  rect( 0, 0, 200, 100 );
}
```



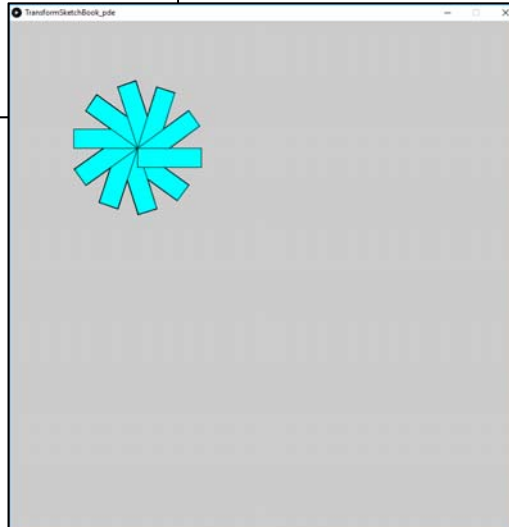
Oregon State
University
Computer Graphics

mjb - January 28, 2019

Transformations and for-loops

11

```
void
draw( )
{
  translate( 200, 200 );
  for( int degrees = 0; degrees <= 360; degrees = degrees + 36 )
  {
    pushMatrix( );
    rotate( radians( degrees ) );
    rect( 0, 0, 100, 30 );
    popMatrix( );
  }
}
```

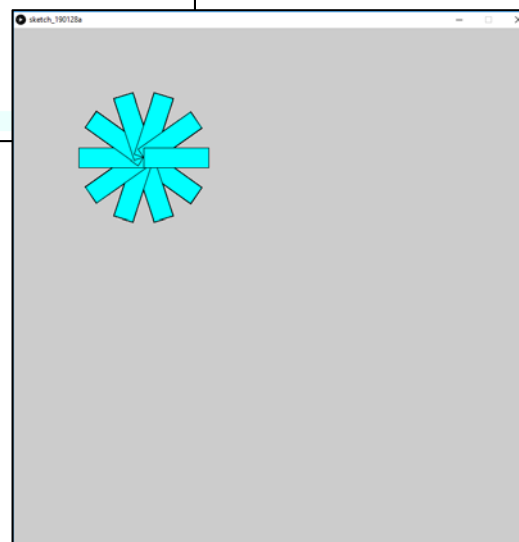


2019

Transformations and for-loops

12

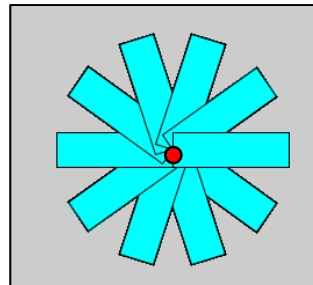
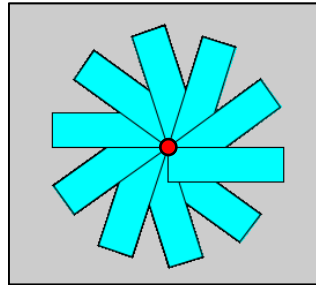
```
void
draw( )
{
  translate( 200, 200 );
  for( int degrees = 0; degrees <= 360; degrees = degrees + 36 )
  {
    pushMatrix( );
    rotate( radians( degrees ) );
    rect( 0, -15, 100, 30 );
    popMatrix( );
  }
}
```



19

What's the Difference?

13



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

```
void
draw( )
{
  translate( 200, 200 );
  for( int degrees = 0; degrees <= 360;
  {
    pushMatrix( );
    rotate( radians( degrees ) );
    rect( 0, -15, 100, 30 );
    popMatrix( );
  }
}
```

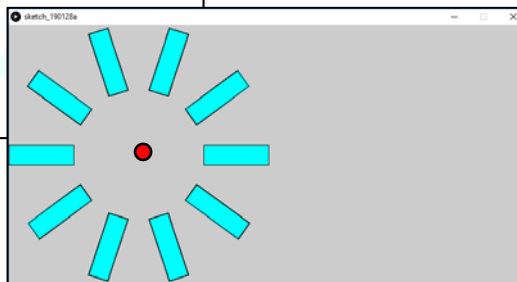
University
Computer Graphics

mjb - January 28, 2019

Transformations and for-loops

14

```
void
draw( )
{
  translate( 200, 200 );
  for( int degrees = 0; degrees <= 360; degrees = degrees + 36 )
  {
    pushMatrix( );
    rotate( radians( degrees ) );
    rect( 100, -15, 100, 30 );
    popMatrix( );
  }
}
```




Oregon State
University
Computer Graphics

19

Rotating While Changing Color and Size

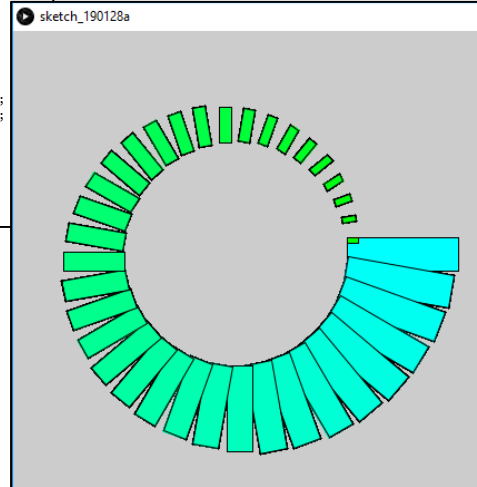
15

```
void
draw( )
{
  translate( 200, 200 );
  for( int degrees = 0; degrees <= 360; degrees = degrees + 10 )
  {
    pushMatrix();
    // map color from cyan to green
    int blue = int( map( degrees, 0, 360, 255, 0 ) );
    fill( 0, 255, blue );

    // transform by rotating:
    rotate( radians( degrees ) );

    // change rectangle size:
    int xsize = int( map( degrees, 0, 360, 100, 10 ) );
    int ysize = int( map( degrees, 0, 360, 30, 5 ) );

    // draw rectangle away from the origin:
    rect( 100, -15, xsize, ysize );
    popMatrix();
  }
}
```



And, there are even 3D Transformations

16

```
translate( x, y, z );
scale( x, y, z );
rotateX( radians );
rotateY( radians );
rotateZ( radians );
```

But, we will get to those later ...