HS2016_Course - Java - oop_why/src/oopwhy/WithoutOOP.java - Eclipse

Drawable.java | Oop2visuals7Sma | OOP2Visuals.jav | House.java | WithoutOOP.java | OopWhy.java | *WithOOP.java

```
1  package oopwhy;
2
3  import processing.core.PApplet;
4
5  public class WithoutOOP extends PApplet {
6
7      float a;
8
9      public void setup() {
10         stroke(0);
11         a = width;
12     }
13
14     public void draw() {
15         background(255);
16         rect(width-a, width/4, height/2, height/2);
17         ellipse(a, height/2, width/2, height/3);
18         line(a,0,a,height);
19         a = (float) (a - 1);
20         if (a < 0) {
21             a = width;
22         }
23     }
24
25     public void settings(){
26         size(360,360);
27     }
28
29     public static void main(String args[]){
30         PApplet.main(new String[]{oopwhy.WithoutOOP.class.getName()});
31     }
32 }
33
```

Writable | Smart Insert | 9 : 26

HS2016_Course - Java - oop_why/src/oopwhy/WithOOP.java - Eclipse

Drawable.java | Oop2visuals7Smar | OOP2Visuals.jav | House.java | WithoutOOP.java | OopWhy.java | WithOOP.java

```
3  import processing.core.PApplet;
4
5  public class WithOOP extends PApplet {
6
7  //  public class Shape {
8  //      public void draw(){
9  //
10 //      }
11 //  }
12
13     public interface Shape {
14         public void draw();
15         public void step();
16     }
17
18     public class Triangle implements Shape{
19         int a = 0;
20         public void draw(){
21             triangle(a, a, a+10, a+20, a+20, a);
22         }
23         public void step(){
24             a++;
25             if (a > width) {
26                 a = 0;
27             }
28         }
29     }
30
31     public class Ellipse implements Shape{
32         int a = width;
33         public void draw(){
34             ellipse(a, 200, 30, 50);
35         }
36         public void step(){
37             a = a -2;
38             if (a < 0){
39                 a = width;
40             }
41         }
42     }
43
44     public class Rectangle implements Shape {
45         int a = height;
46         public void draw(){
47             rect(30, a, 40, 40);
48         }
49         public void step(){
50             a--;
51             if (a < 0) {
52                 a = height;
53             }
54         }
55     }
56
57     Shape[] shapes;
58
59     public void setup() {
60         stroke(0);
61         shapes = new Shape[]{new Triangle(), new Ellipse(), new Rectangle()};
62     }
63
64     public void draw() {
65         background(0);
66         for (Shape shape: shapes){
67             shape.step();
68             shape.draw();
69         }
70     }
71
```

Writable | Smart Insert | 40 : 14