

# Object Oriented Programming

(OOP)

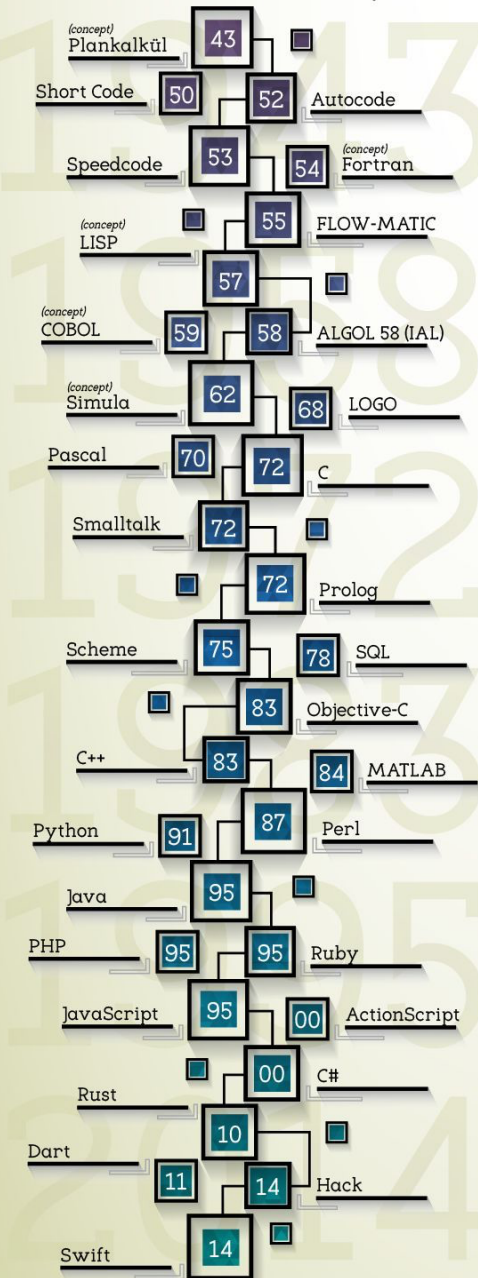


Abstraction of our world ▲



## Timeline of Programming Languages

by Alexandria O'Brien




# Paradigms

- Imperative
  - **Procedural**
  - **Object Oriented**
  - Parallel Processing
- Declarative
  - Logic
  - Functional
  - Database



# Procedural Programming

- Variables
- Methods / Functions
- Loops
- Arrays



```
// circles
int[] circles = new int[7];

for(int i = 0; i < circles.length; i++) {
    println("Position: " + circles[i]);
}
```


- How to add a color to each circle?



```
// circles
int[] positions = new int[7];
color[] colors = new color[7];

for(int i = 0; i < positions.length; i++) {
    int p = positions[i];
    color c = colors[i];
}
```

- How to add a size to each circle?



```
// circles
int[] positions = new int[7];
color[] colors = new color[7];
float[] sizes = new float[7];

for(int i = 0; i < positions.length; i++) {
    int p = positions[i];
    color c = colors[i];
    float s = sizes[i];
}
```

- It's getting messy & complicated 🙄



A new paradigm.

# Object Oriented Programming

- Class & Object
- Abstraction\*
- Inheritance\*
- Polymorphism\*





# Bicycle 🚲

- It has...
  - Frame Color
  - Wheel Size
  - Type
- It can...
  - Accelerate
  - Brake

# Object

- Attributes (Properties)
  - Variables that describe the state of the object.
- Methods
  - Things we can do with the object.

# Pen

- Attributes
  - Color
  - Thickness
- Methods
  - Write
  - Refill

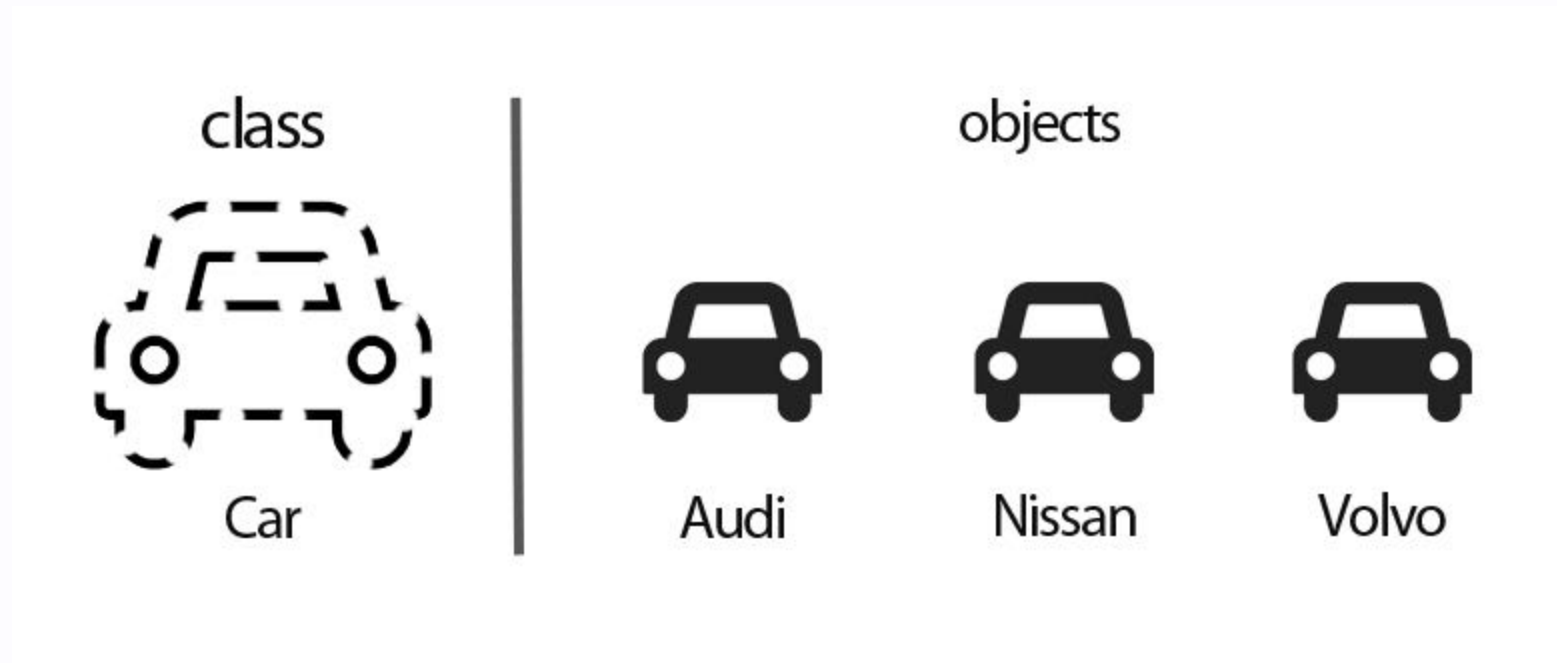




# Circle

- Attributes
  - Position
  - Color
  - Size
- Methods
  - Draw

# Class vs Object



- Very important to understand! ⚡

# How to define a class?

```
class Circle {  
    int position = 0;  
    color c = color(255);  
    float size = 30;  
  
    void paint() {  
        fill(c);  
        circle(position, 0, size);  
    }  
}
```

- Capital Letter 🙌

# How to use a class?

- Create a new instance of a class

```
Circle c = new Circle();  
  
c.size = 40;  
c.position = 10;  
  
c.paint();
```

# Using Multiple Instances

```
Circle circleA = new Circle();  
Circle circleB = new Circle();  
Circle circleC = new Circle();  
  
circleA.size = 40;  
circleC.position = 20;  
  
circleA.paint();  
circleB.paint();
```



# Task Poly Walker (30min)

Adapt the Random Walker Task to combine `position` and `velocity` values into a **Walker** class.

Give each walker a different `color` and a `paint()` method to draw it.

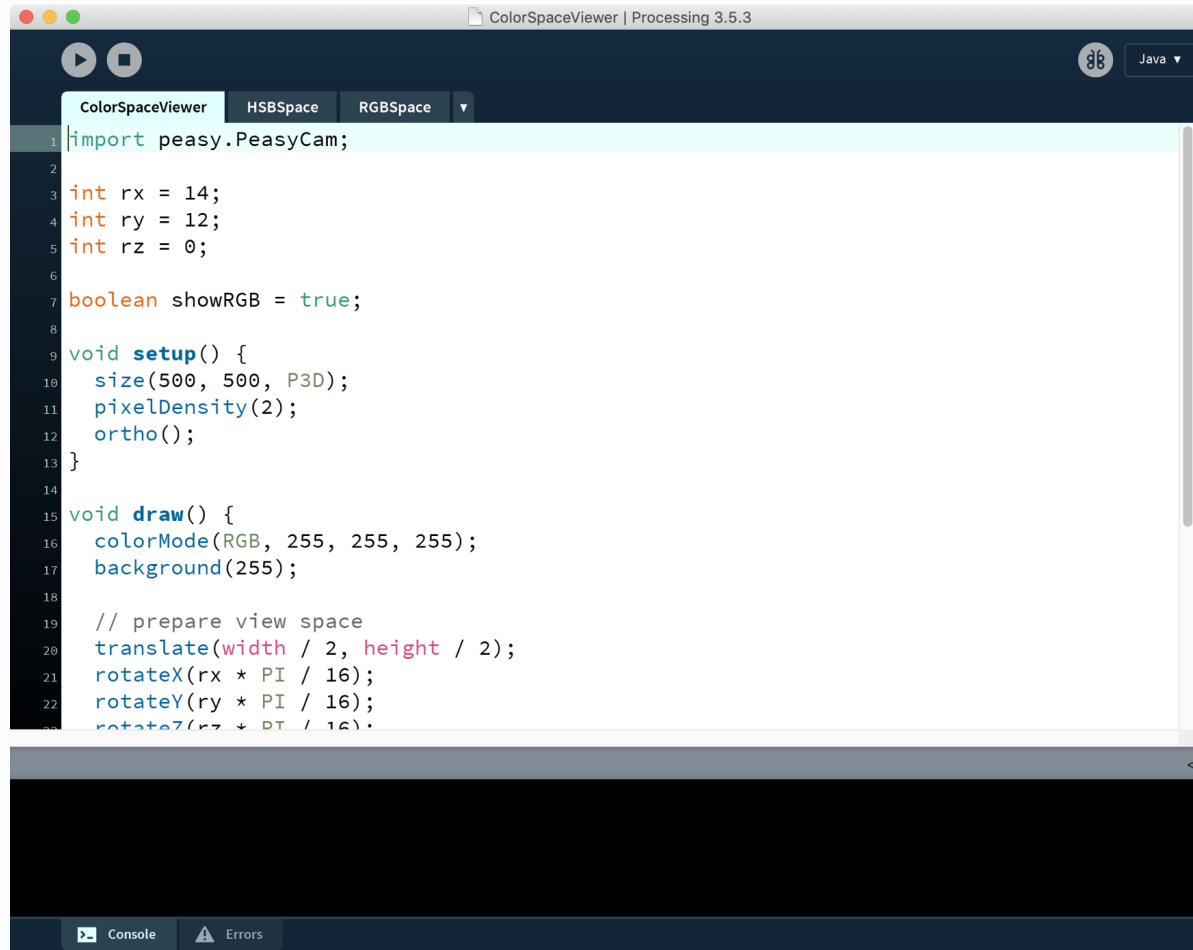
Hint: [processing.org/tutorials/objects/](https://processing.org/tutorials/objects/)

# Using Multiple Instances

```
Circle[] circles = new Circle[10];  
  
circles[0] = new Circle();  
circles[1] = new Circle();  
  
circles[1].size = 30;
```

- `ArrayList<Circle> circles = new ArrayList<>();`

# Processing Tabs



```
ColorSpaceViewer | Processing 3.5.3
ColorSpaceViewer HSBspace RGBspace
1 import peasy.PeasyCam;
2
3 int rx = 14;
4 int ry = 12;
5 int rz = 0;
6
7 boolean showRGB = true;
8
9 void setup() {
10   size(500, 500, P3D);
11   pixelDensity(2);
12   ortho();
13 }
14
15 void draw() {
16   colorMode(RGB, 255, 255, 255);
17   background(255);
18
19   // prepare view space
20   translate(width / 2, height / 2);
21   rotateX(rx * PI / 16);
22   rotateY(ry * PI / 16);
23   rotateZ(rz * PI / 16);
24 }
```

# Questions?