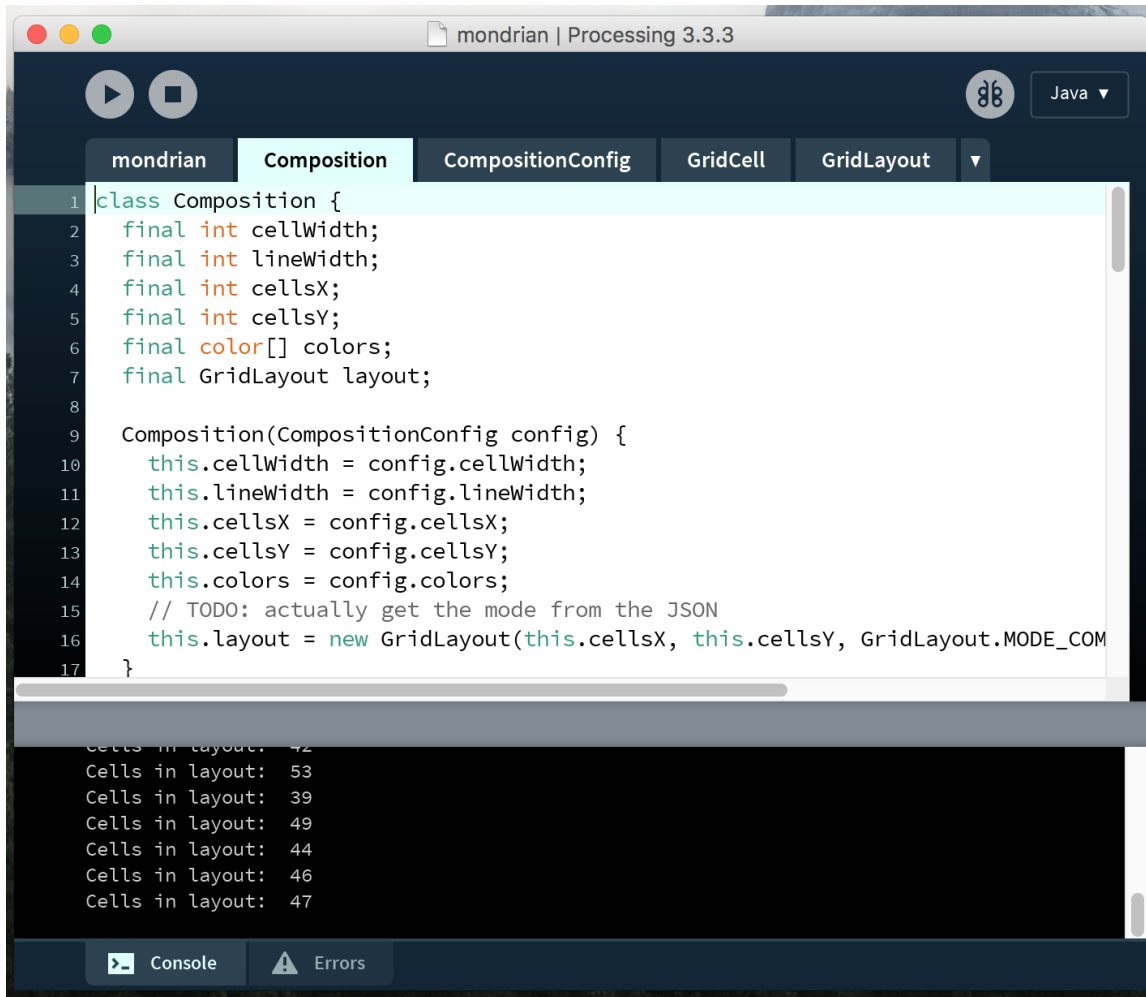


Processing

Make graphics with code!



The screenshot shows the Processing IDE interface. The title bar indicates the file is named 'mondrian' and the version is 'Processing 3.3.3'. The interface includes a toolbar with play and stop buttons, a language dropdown set to 'Java', and a tabbed editor with tabs for 'mondrian', 'Composition', 'CompositionConfig', 'GridCell', and 'GridLayout'. The 'Composition' tab is active, displaying the following Java code:

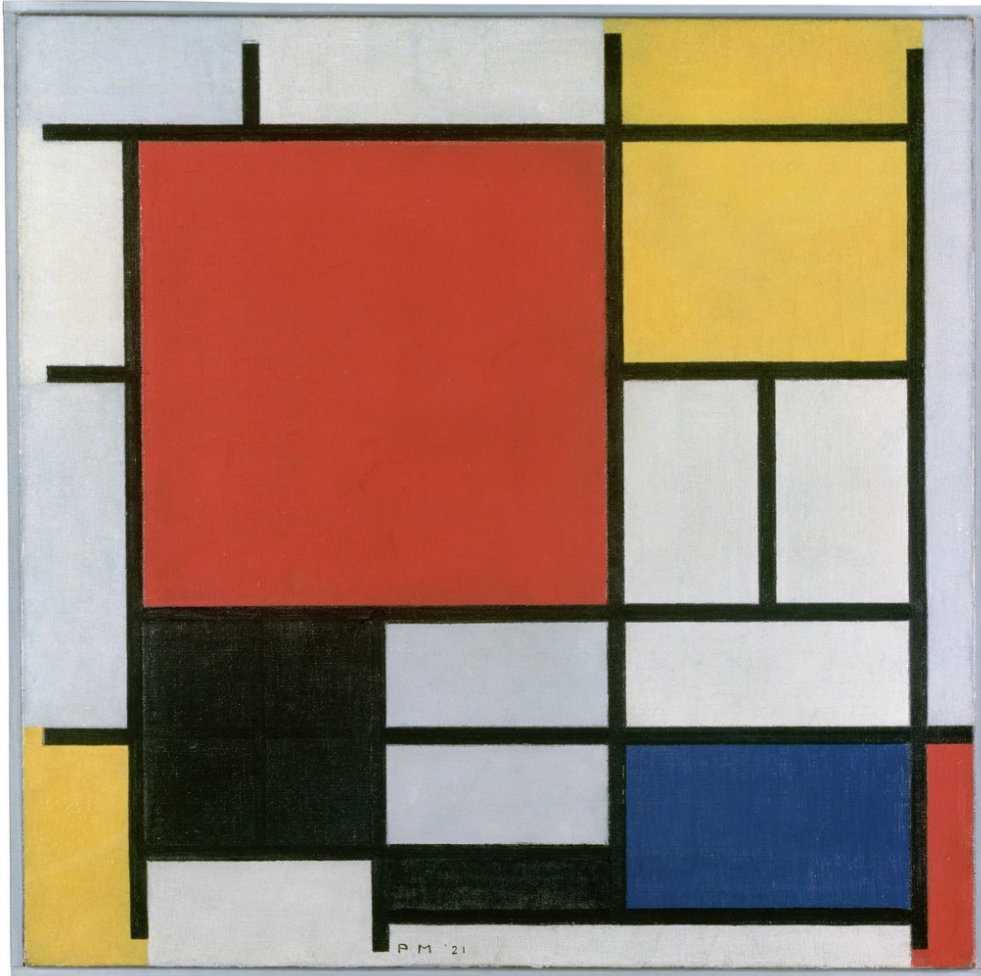
```
1 class Composition {
2   final int cellWidth;
3   final int lineWidth;
4   final int cellsX;
5   final int cellsY;
6   final color[] colors;
7   final GridLayout layout;
8
9   Composition(CompositionConfig config) {
10    this.cellWidth = config.cellWidth;
11    this.lineWidth = config.lineWidth;
12    this.cellsX = config.cellsX;
13    this.cellsY = config.cellsY;
14    this.colors = config.colors;
15    // TODO: actually get the mode from the JSON
16    this.layout = new GridLayout(this.cellsX, this.cellsY, GridLayout.MODE_COM
17  }
```

Below the code editor is a console window showing the output of the program:

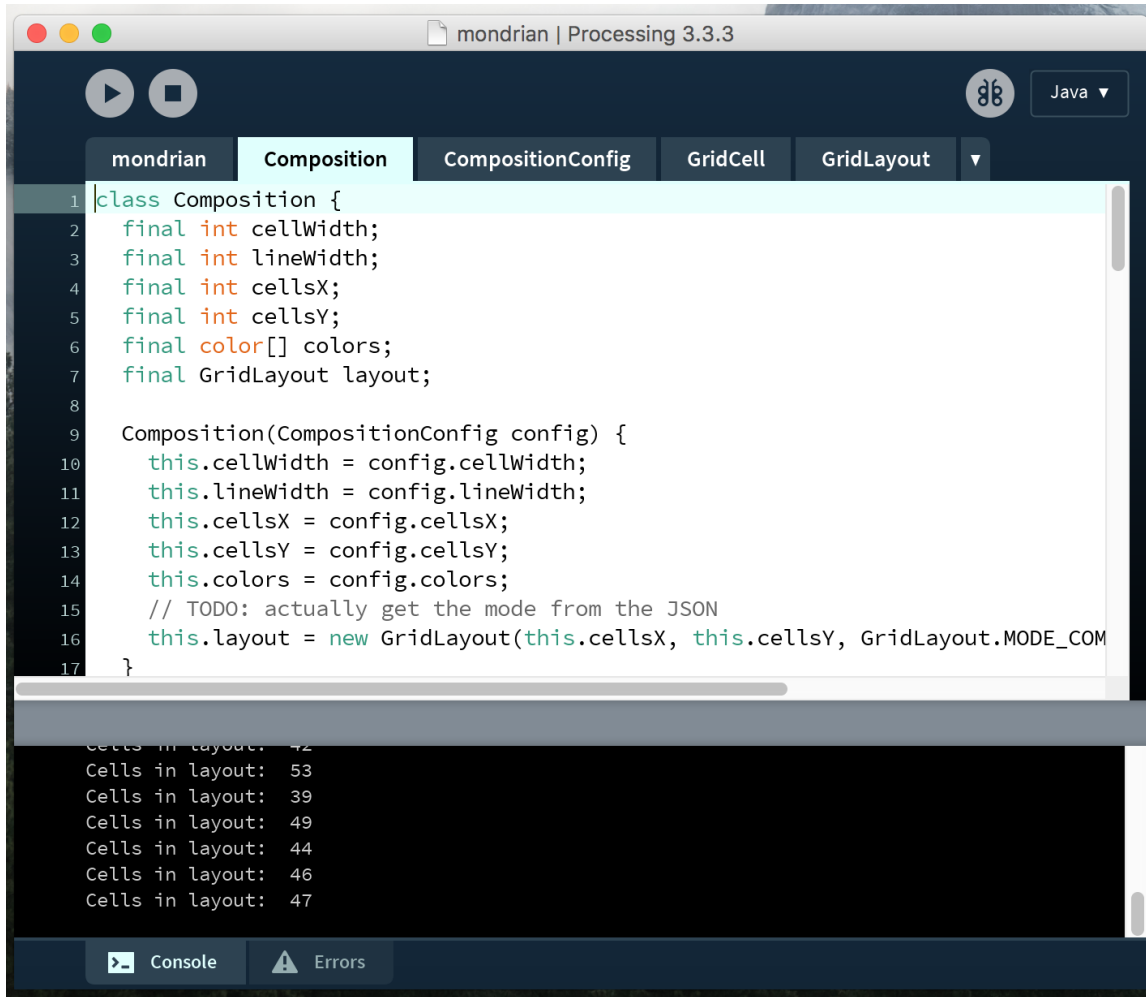
```
Cells in layout: 42
Cells in layout: 53
Cells in layout: 39
Cells in layout: 49
Cells in layout: 44
Cells in layout: 46
Cells in layout: 47
```

At the bottom of the IDE, there are tabs for 'Console' and 'Errors'.

Example: Modern art



Write code...



The screenshot shows the Processing IDE interface. The title bar indicates the file is 'mondrian' and the version is 'Processing 3.3.3'. The interface includes a toolbar with play and stop buttons, a language dropdown set to 'Java', and a tabbed editor with tabs for 'mondrian', 'Composition', 'CompositionConfig', 'GridCell', and 'GridLayout'. The 'Composition' tab is active, displaying the following Java code:

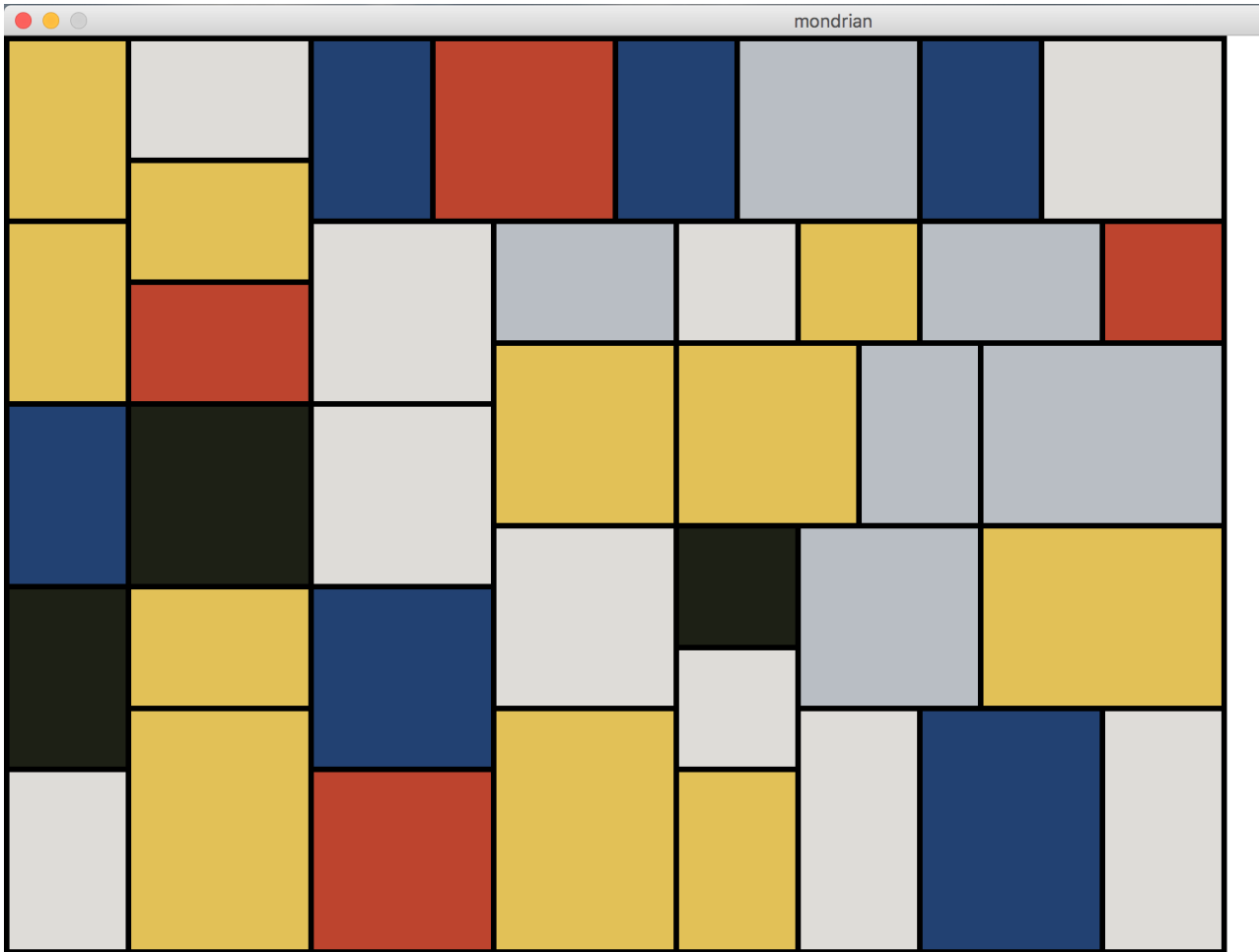
```
1 class Composition {  
2   final int cellWidth;  
3   final int lineWidth;  
4   final int cellsX;  
5   final int cellsY;  
6   final color[] colors;  
7   final GridLayout layout;  
8  
9   Composition(CompositionConfig config) {  
10    this.cellWidth = config.cellWidth;  
11    this.lineWidth = config.lineWidth;  
12    this.cellsX = config.cellsX;  
13    this.cellsY = config.cellsY;  
14    this.colors = config.colors;  
15    // TODO: actually get the mode from the JSON  
16    this.layout = new GridLayout(this.cellsX, this.cellsY, GridLayout.MODE_COM  
17  }
```

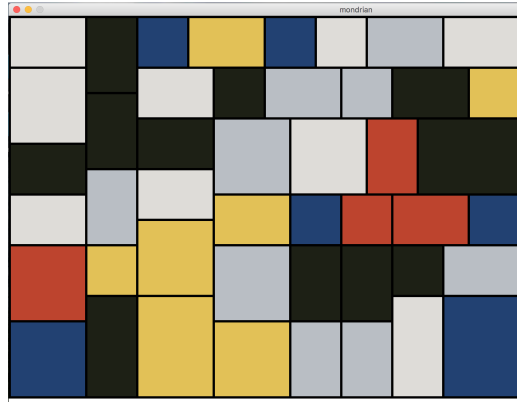
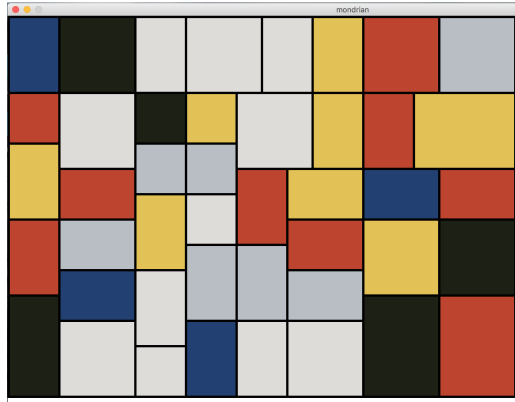
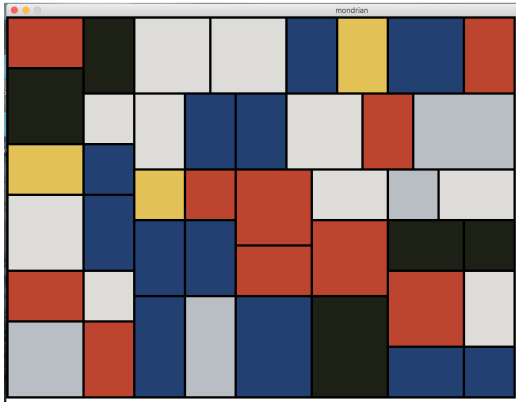
Below the code editor is a console window showing the output of the program:

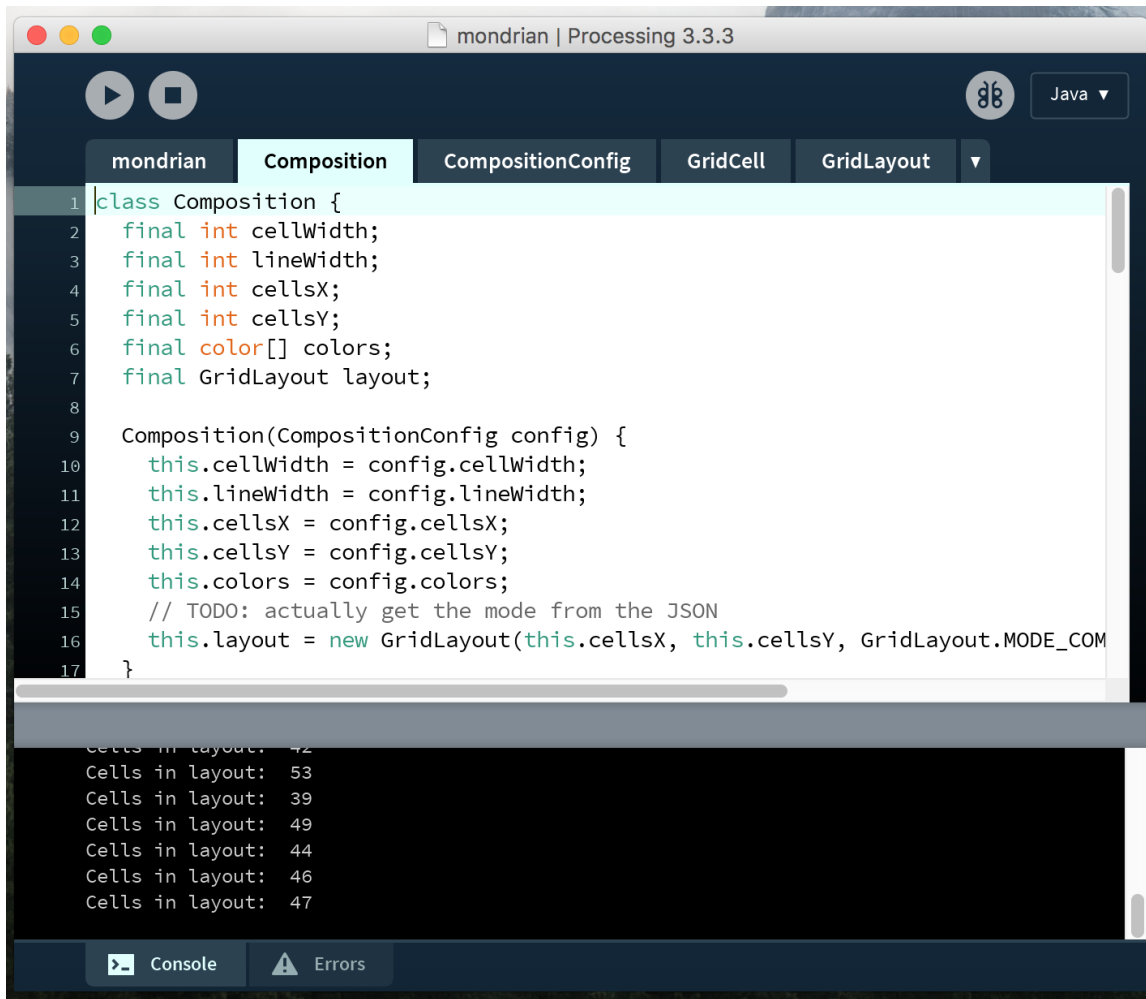
```
Cells in layout: 42  
Cells in layout: 53  
Cells in layout: 39  
Cells in layout: 49  
Cells in layout: 44  
Cells in layout: 46  
Cells in layout: 47
```

At the bottom of the IDE, there are tabs for 'Console' and 'Errors'.

The code turns into pictures!



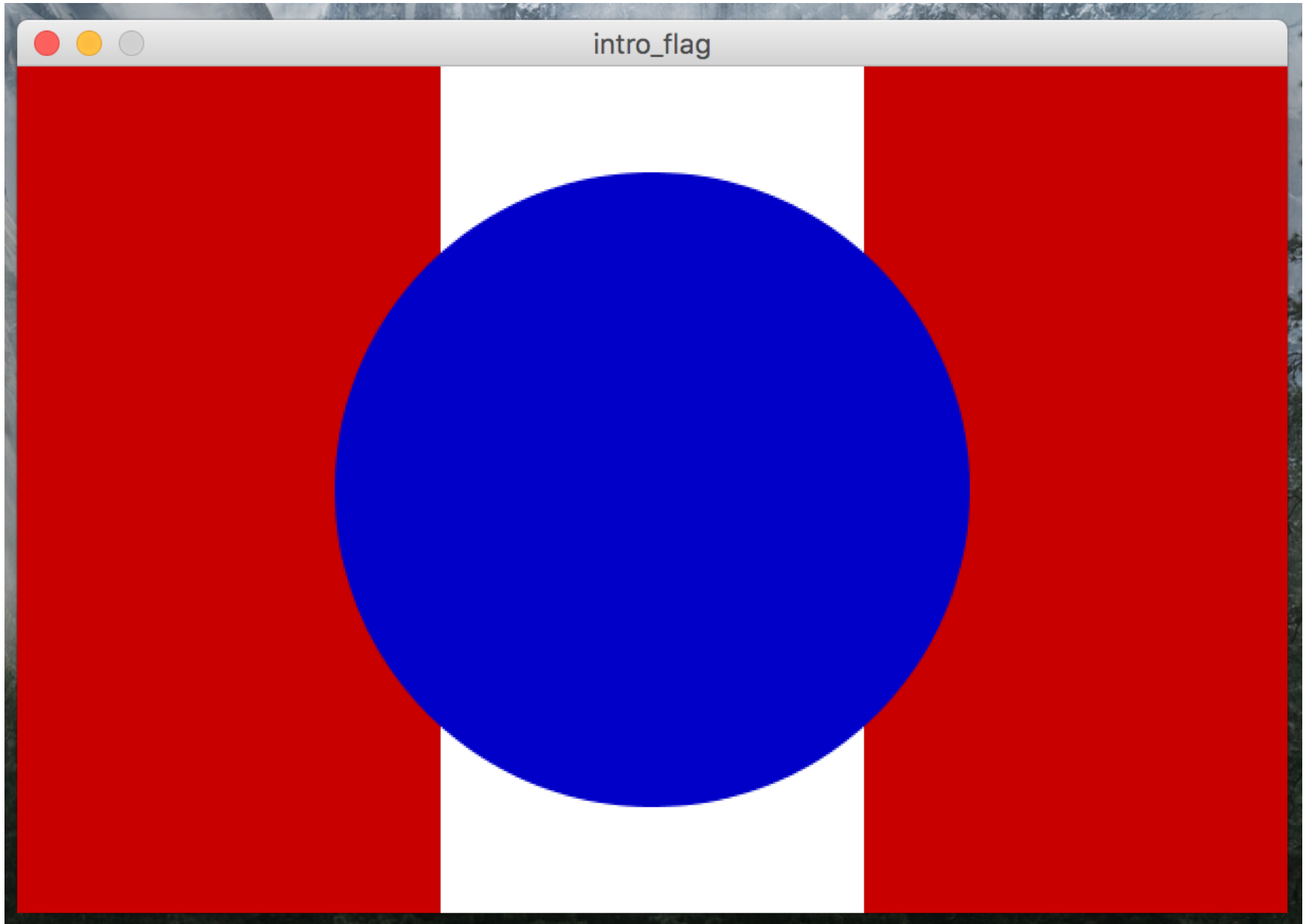




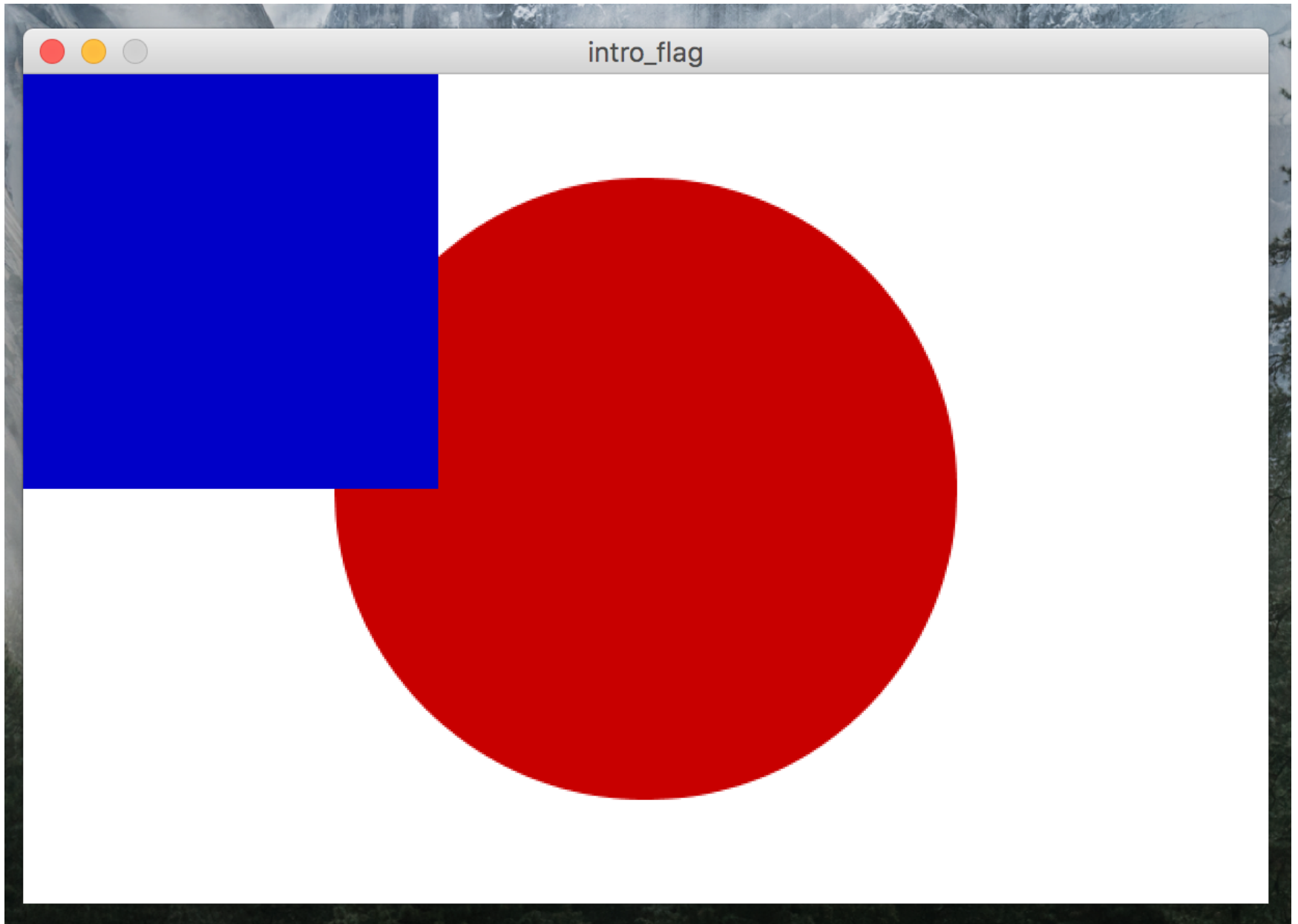
By the end of the course, you will know how to create this!

Simpler example...

Flag for an imaginary country



We have a headstart!



Download the starting code

1. Go to https://github.com/UtahRETC/intro_flag
2. Click "Clone or download"
3. Click "Download ZIP"

The Processing Reference

<https://processing.org/reference/>

Processing

p5.js

Processing.py

Processing for Android

Processing Foundation

Processing

Cover

Download

Donate

Exhibition

Reference

Libraries

Tools

Environment

Tutorials

Examples

Books

Handbook

Overview

People

Shop

» Forum

» GitHub

» Issues

» Wiki

» FAQ

» Twitter

» Facebook

» Medium

Reference. Processing was designed to be a flexible software sketchbook.

Structure

Shape

Color

() (parentheses)

, (comma)

. (dot)

/* */ (multiline comment)

/** */ (doc comment)

// (comment)

;(semicolon)

= (assign)

[] (array access)

{ } (curly braces)

catch

class

draw()

exit()

extends

false

final

implements

import

loop()

new

noLoop()

null

popStyle()

private

public

pushStyle()

redraw()

return

createShape()

loadShape()

PShape

2D Primitives

arc()

ellipse()

line()

point()

quad()

rect()

triangle()

Curves

bezier()

bezierDetail()

bezierPoint()

bezierTangent()

curve()

curveDetail()

curvePoint()

curveTangent()

curveTightness()

3D Primitives

box()

sphere()

sphereDetail()

Setting

background()

clear()

colorMode()

fill()

noFill()

noStroke()

stroke()

Creating & Reading

alpha()

blue()

brightness()

color()

green()

hue()

lerpColor()

red()

saturation()

Image

createImage()

PImage

Loading & Displaying

image()

imageMode()