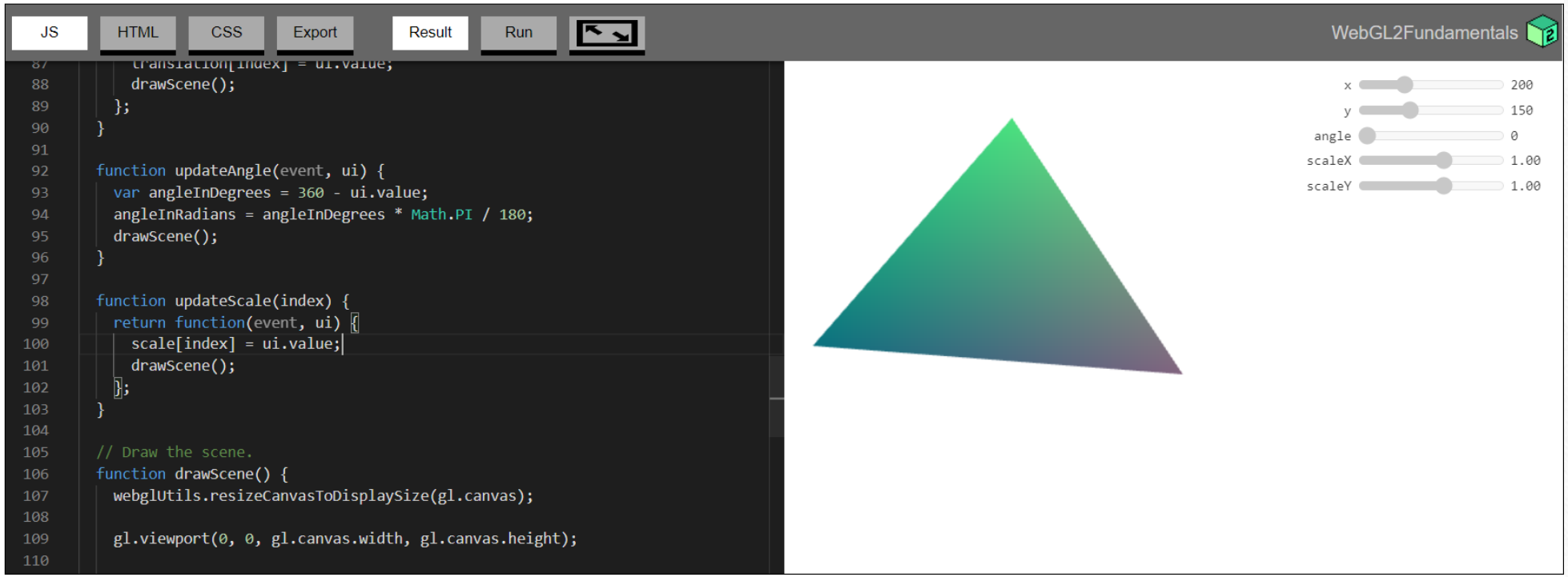
Sierpinski Triangle

我是使用How it works中已經有UI控制的範例下去改的



Step1: 設定UI

```
<canvas id="canvas" width="1000" height="800"></canvas>
<div id="uiContainer">
  <div id="ui">
    <div id="x"></div>
    <div id="y"></div>
    <div id="angle"></div>
    <div id="scaleX"></div>
    <div id="scaleY"></div>
    <div id="Layer"></div>
    <div id="Colorful"></div>
```

```
var layer = 0;//層數從0開始 也就是一個三角形
var colorcheck = 0;//控制顏色 0表黑白 1表隨機彩色

drawScene(layer, colorcheck);//畫出一開始layer=0時的三角形

// Setup a ui.
webglLessonsUI.setupSlider("#x",      {value: translation[0], slide: updatePosition(0), max: gl.canvas.width });
webglLessonsUI.setupSlider("#y",      {value: translation[1], slide: updatePosition(1), max: gl.canvas.height});
webglLessonsUI.setupSlider("#angle",  {slide: updateAngle, max: 360});
webglLessonsUI.setupSlider("#scaleX", {value: scale[0], slide: updateScale(0), min: -5, max: 5, step: 0.01, precision: 2});
webglLessonsUI.setupSlider("#scaleY", {value: scale[1], slide: updateScale(1), min: -5, max: 5, step: 0.01, precision: 2});
webglLessonsUI.setupSlider("#Layer", {value: layer, slide: updateLayer(1), min: 0, max: 9, step: 1, precision: 1});//新增控制層數的滑軌
webglLessonsUI.setupSlider("#Colorful", {value: colorcheck, slide: updateColor(1), min: 0, max: 1, step: 1, precision: 1});//控制顏色的滑軌
```

Step1: 設定UI

- 由前頁所示，保留原本的UI，新增兩個滑軌用於控制層數(layer)與控制顏色(colorful)，均從0開始，並給予function作值的更新。
- layer範圍: 0~9。
- Colorful : 0表示黑色，1表示隨機顏色。

Step1: 設定UI

- 控制值的function。

```
153
154   function updateLayer(index) {//控制層數的function
155     return function(event, ui) {
156       layer = ui.value;
157       drawScene(layer, colorcheck);
158     };
159   }
160
161   function updateColor(index) {//控制顏色
162     return function(event, ui) {
163       colorcheck = ui.value;
164       drawScene(layer , colorcheck);
165     };
166   }
167
```

Step2: 初始點座標

- 在drawScene( )給予第一個三角形初始的三個點座標後，呼叫drawww( )畫圖。

```
193     //第一個三角形三點的初始位置
194       var x1 = 0;
195       var y1 = 400;
196       var x2 = 600;
197       var y2 = 400;
198       var x3 = 300;
199       var y3 = 0;
200
201       drawww(x1 ,y1 ,x2 ,y2 ,x3 ,y3 ,index, colorcheck);//呼叫drawww作畫
202   }
```

Step3: 畫圖

s1093319-hw1.html

D: > download > s1093319-hw1.html > html > script > main > updateAngle

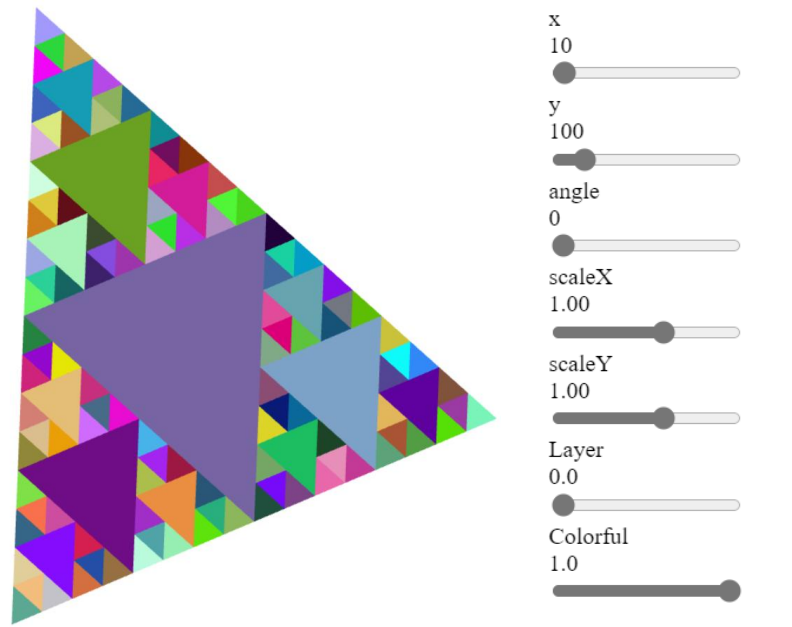
```
203
204     function draww(a1 ,b1 ,a2 ,b2,a3 ,b3, layer, colorcheck){//畫三角形的recursive
205       var x1 = a1;
206       var y1 = b1;
207       var x2 = a2;
208       var y2 = b2;
209       var x3 = a3;
210       var y3 = b3;
211
212       //colorcheck=0 為黑色
213       if(colorcheck == 0){
214           if(layer!=0)gl.uniform4f(colorLocation, 0, 0, 0, 0 );
215           else gl.uniform4f(colorLocation, 0, 0, 0, 1 );
216       }
217       else gl.uniform4f(colorLocation, Math.random(), Math.random(), Math.random(), 1);//否則隨機顏色
218
219       //畫出三角形
220       setGeometry_3(gl ,x1 ,y1 , x2 , y2 ,x3 ,y3);
221       var offset = 0;
222       var count = 3;
223       gl.drawArrays(gl.TRIANGLES, offset, count);
224
225       if(layer == 0) return;
226           else{
227             //找到三角形三邊的中點，與各頂點形成小三角形再次呼叫直到layer=0
228             var dx = getmidpoint(x1 ,x2);
229             var dy = getmidpoint(y1 ,y2);
230             var ex = getmidpoint(x2 ,x3);
231             var ey = getmidpoint(y2 ,y3);
232             var fx = getmidpoint(x3 ,x1);
233             var fy = getmidpoint(y3 ,y1);
234             draww(x1 ,y1 ,dx ,dy ,fx ,fy ,layer-1, colorcheck);
235             draww(dx ,dy ,x2 ,y2 ,ex ,ey ,layer-1, colorcheck);
236             draww(fx ,fy ,ex ,ey ,x3 ,y3 ,layer-1, colorcheck);
237           }
238     }
```

1) 由傳入的點座標畫出三角形。
2) 取三角形各邊中點，兩兩搭配一個原本的頂點形成三個小三角形。
3) 對於這三個小三角形再個別呼叫drawww( )做遞迴，直到layer等於0。

Step3: 畫圖

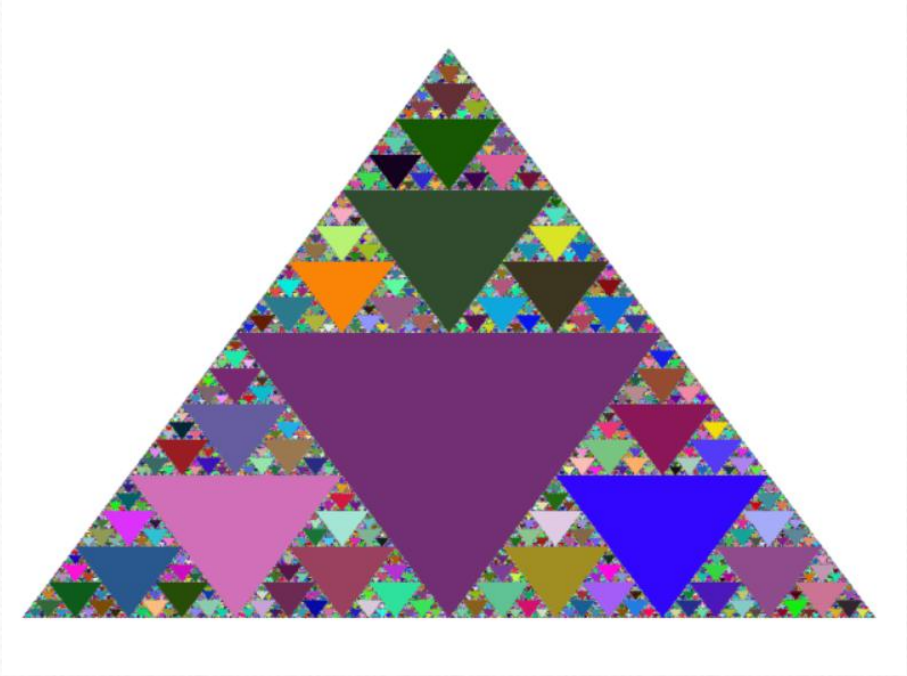
- 其實就相當於一直畫這個圖形。對每個切出來的小三角形做這件事，根據層數多寡重複。

成果:

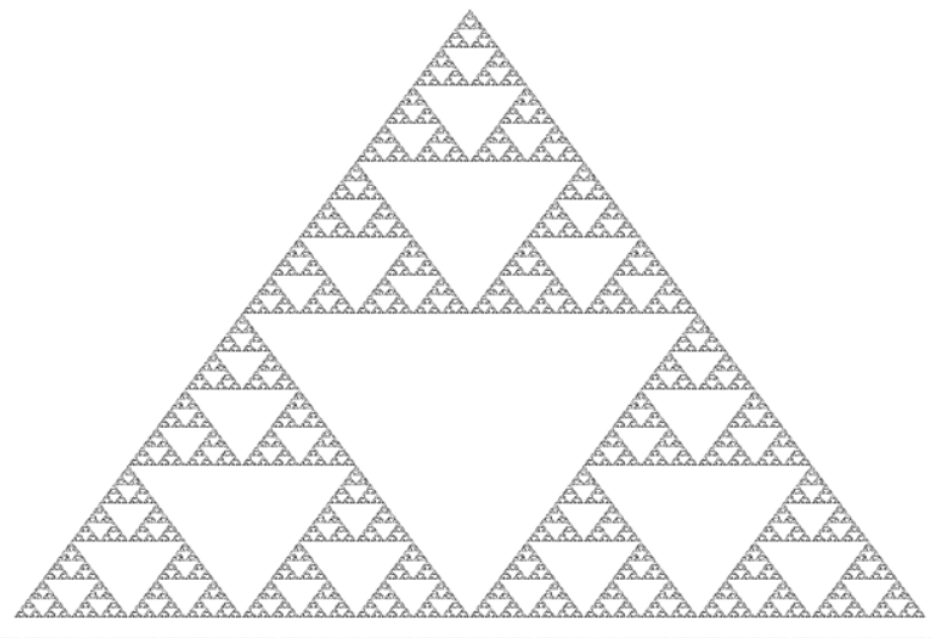


- 保留了原本範例中所有的UI功能。
- 新增:
  ① 調整層數。
  ② 選擇黑白或彩色。

成果:



- 彩色



- 黑白