

Conditions (If's) in JavaScript

Introduction

In this week's lecture you were told about conditions in JavaScript. In this tutorial you will explore these concepts by creating some simple programs.

Task 1: A simple IF statement

Create a new HTML file and between two `<script> ... </script>` tags, type the following code:

```
// Calculate mark classification
var mark = 55;

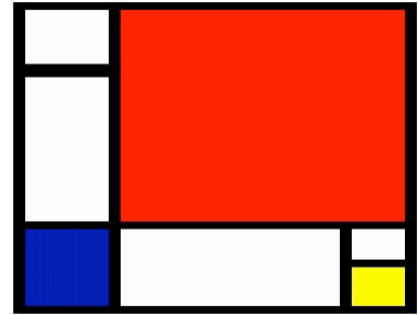
if (mark >= 70){
    console.log("This is a First");
}
else if (mark >= 60 && mark <= 69){
    console.log("This is a 2:1");
}
else if (mark >= 50 && mark <= 59){
    console.log("This is a 2:2");
}
else if (mark >= 40 && mark <= 49){
    console.log("This is a Third");
}
```

Run your code by loading the webpage in Google Chrome with the JavaScript console visible. This program using a set of conditions to test what classification a mark is and outputs this using the `console.log()` method.

Change the value for the mark variable to test whether each condition works as expected.

Task 2: Mondrian Generator

“Piet Mondrian was a Dutch painter and an important contributor to the De Stijl art movement, which was founded by Theo van Doesburg. Despite being well-known, often-parodied, and even trivialized, Mondrian's paintings exhibit a complexity that belie their apparent simplicity. The non-representational paintings for which he is best known, consisting of rectangular forms of red, yellow, blue, or black, separated by thick, black, rectilinear lines, are actually the result of a stylistic evolution that occurred over the course of nearly thirty years, and which continued beyond that point to the end of his life.”



https://en.wikipedia.org/wiki/Piet_Mondrian

Until now, you have been outputting the results of your programs to the JavaScript console. However, the console is restricted to showing unformatted text so the next step is to start to output the results of our code in the HTML pages itself. One method of doing this is to use the HTML5 `canvas` element, which “is used to draw graphics, on the fly, on a web page”:

http://www.w3schools.com/html/html5_canvas.asp

The `canvas` element is simply an area on a webpage where you can draw graphics using JavaScript. In a new HTML page type the following “skeleton” code in the `body` of the page that will generate a very simple version of a Mondrian painting using a combination of functions and conditions. You were shown the completed code during the lecture.

Copy this code into a new HTML document taking care to think about what each line is doing.

```
<> mondrian_scaffold.html > html > body > script > draw
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <title>Document</title>
6  </head>
7  <body>
8    <canvas id="sCanvas" width="550" height="400"></canvas>
9
10  <script>
11    // Variables
12    // Colours
13    const blue = "#0000FF";
14    const green = "#006600";
15    const brown = "#543725";
16    const red = "#FF0000";
17    const yellow = "#FFFF00";
18    const grey = "#666666";
19    const white = "#FFFFFF";
20    const black = "#000000";
21    let colour; // global variable for use in functions
```

```

22
23 // Get the Canvas element with ID sCanvas that will be drawn on
24 const c = document.getElementById("sCanvas");
25 const ctx = c.getContext("2d");
26
27 // Draw a square with a specified size, position and colour
28 function drawFilledShape(sWidth,sHeight, x,y,colour){
29     ctx.beginPath();
30     ctx.rect(x,y,sWidth,sHeight);
31     ctx.fillStyle = colour;
32     ctx.fill();
33     ctx.strokeStyle = black;
34     ctx.lineWidth = 5;
35     ctx.stroke();
36 }
37
38 function turnfNameIntoColour(fName){
39     if(fName.length > 6){
40         colour = white;
41     }
42     else{
43         colour = grey;
44     }
45     return colour;
46 }
47
48 function turnAgeIntoColour(age){
49
50     return colour;
51 }
52
53 function turnEyeColourIntoColour(eyeColour){
54
55     return colour;
56 }
57
58 function draw(){
59     let age = 53; //using let as will be taking user input later
60     let eyeColour = "blue"; // choose from: blue, green, brown
61     let fName = "Matt";
62
63     // Turn details into colours
64     const sColourTopLeft = turnAgeIntoColour(age);
65     const sColourBottomLeft = turnfNameIntoColour(fName);
66     const sColourRight = turnEyeColourIntoColour(eyeColour);
67
68     // Draw the coloured shapes
69     drawFilledShape(150, 150, 0, 0, sColourTopLeft);
70     drawFilledShape(150, 250, 0, 150, sColourBottomLeft);
71     drawFilledShape(400, 400, 150, 0, sColourRight);
72 }
73
74 draw(); //call the main draw function
75 </script>
76 </body>
77 </html>

```

At the moment this code will run but you will notice that two of the squares (`sColourBottomLeft` and `sColourRight`) are grey and one is black (`sColourTopLeft`).

Your task is to create a set of conditions in the functions `turnEyeColourIntoColour()` and `turnAgeIntoColour()` to change the colour of those squares depending on a value that is sent to the function.

To help you, you have been given the code (line 38) for the `turnfNameIntoColour()` function which has a String parameter, `fName`, and then checks to see whether the length of `fName` i.e. how many letters, is greater than 6 (`fName.length > 6`). If it is, then the function returns the colour as `white`, otherwise it returns the colour as `black`.

Once you have typed in the code, try and understand what each section does.

turnEyeColourIntoColour

For the `turnEyeColourIntoColour` function create a set of conditions that simply takes the `eyeColour` parameter and sets the colour variable to be the same as the value of the `eyeColour` parameter e.g.

```
if(eyeColour == "green"){
    colour = green;
}
```

do this for the colours green, blue and brown.

turnAgeIntoColour

For the `turnAgeIntoColour` function, create a set of conditions that takes the `age` parameter and if `age` is less than 30, the colour is `yellow`, otherwise the colour is `red`.

Once you have created these sets of conditions, change the values of the `age`, `eyeColour` and `fName` variables to your own details and see what your personalised Mondrian looks like.

Task 3: Mondrian Generator with user input

In the lecture you were shown an improved version of the Mondrian generator that allowed the user to enter their age, name and select eye colour from a drop-down menu.

Re-save your code from task 2 *as* `task3.html`.

For task 4 you are to create a text box for the user to enter their name into and a button that when clicked will take their name and use it as a value for the variable `fName` in the code.

Steps you will need to carry out.

1. Comment out the function call `draw()` on line 74, we want to use a button to call the function, not have it run automatically.

```
72 }  
73  
74 //draw(); //call the main draw function  
75 </script>
```

2. In the html create a text box with id 'fName' and a button that calls function `draw()` when clicked (you were given the code for this in the interactive area calculator in the previous lab session). These elements should be within the body of your HTML doc but before the `<script>` tags.
3. Remove the hard coded value assigned to variable `fName` (see line 61 below) and add the code necessary to get the value from the text box you created above in step 1. (This technique was explained again in today's lecture and given to you in the previous lab session).

```
58 function draw(){  
59   let age = 53; //using let as will be taking user input later  
60   let eyeColour = "blue"; // choose from: blue, green, brown  
61   let fName =
```

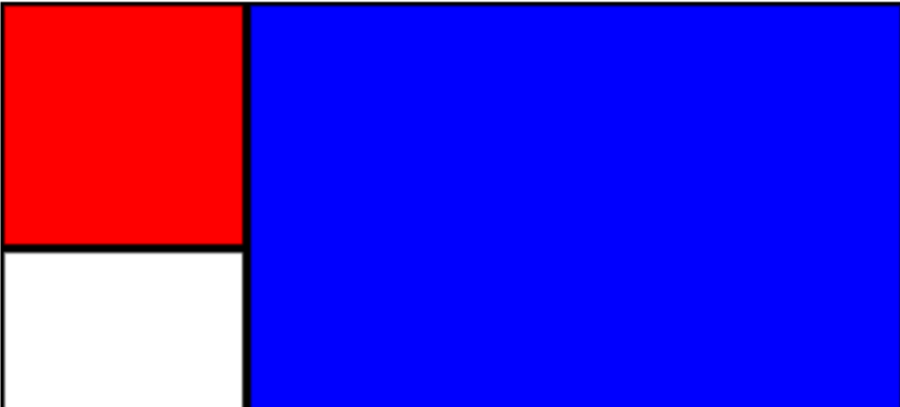
When you load the page you should see something like this:

Name:

Nothing is being drawn yet because the button click will call the draw function.

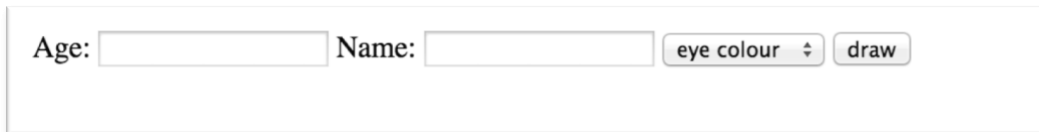
Enter your name and click the button, you should see something like this:

Name:



The example I showed in the lecture had a text box for age and a drop down menu for eye colour.

If you have got the code for the user's name and the button working see if you can implement age and eye colour too.



Age: Name: eye colour

Questions to consider

1. Why have I used a drop-down list for eye colour and not just provided another text box?
2. In the example I showed in the lecture I had alert boxes popping up if I did not enter anything into the text boxes before clicking the button. This technique is known as *form validation* or *data input validation*. In the code I am asking questions, using if statements, such as “is this text box empty” or “is this data a number or not”. If not display an alert box message and require the user to go back and try again.

Loops in JavaScript

Introduction

In this week's lecture you were told about loops in JavaScript. In this tutorial you will explore these concepts by creating some simple programs that use the HTML Canvas

Task 1: "Art Balling" Generator

"Former England cricket captain Michael Vaughan has found a new use for that pull shot: whacking paint-daubed balls at canvas to produce highly collectable works of art. Watch out, abstract expressionism - here comes 'artballing'" –

The following code is the "skeleton" for a program that will generate a very simple version of a Michael Vaughan painting using a combination of functions and loops. You were shown the completed code during the lecture.



```
10 <canvas id="sCanvas" width="550" height="400"></canvas>
11 <script>
12 // Art Balling Generator 550px x 400px // Using Loops
13 // Variables
14 var randomColour;
15 var randomSize;
16 var xPos;
17 var yPos;
18 var i; // counter
19 var j; // counter
20 // Get the Canvas element with ID sCanvas that will be drawn on
21 c = document.getElementById("sCanvas");
22 ctx = c.getContext("2d");
23
24 function drawFilledCircle(size,xPos,yPos,colour){
25     ctx.beginPath();
26     ctx.arc(xPos,yPos,size,0,2*Math.PI);
27     ctx.fillStyle = colour;
28     ctx.fill();
29 }
30
31 function drawSplatter(size,xPos,yPos,colour){
32     for(j=0;j<10;j++){
33         var splatSize = size / Math.round(Math.random()*30);
34         drawFilledCircle(splatSize,xPos + Math.round(Math.random()*50),yPos + Math.round(Math.random()*50),colour);
35     }
36 }
37
38 </script>
39
```

This will not currently run as it requires code that utilises the given functions. Spend time first of all trying to understand what the 2 functions will do. You will notice that the `drawFilledCircle` function is very similar to the function you used in the last tutorial to create coloured squares. The `drawSplatter` function utilises a loop to draw 10 circles of random shapes and positions dependent on 4 parameters (`size,xPos,yPos,colour`) passed to the function.

Calling the functions in a loop

The following code can be used to create one “normal circle” and one “circle splatter”. Insert it around line 37 before the closing script tag.

```
randomSize = Math.round(Math.random()*50);
xPos = Math.round(Math.random()*550);
yPos = Math.round(Math.random()*400);
randomColour = '#' + Math.random().toString(16).substring(2, 8);
drawFilledCircle(randomSize, xPos, yPos, randomColour);
drawSplatter(randomSize, xPos, yPos, randomColour);
```

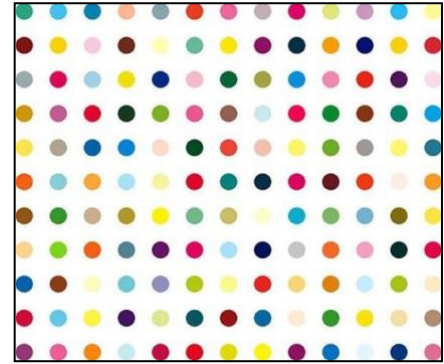
Your task is to take the above code and using either a for loop or a while loop, draw 10 “normal circles” and “circle splatters”. Once you have done this modify your code so that it produces a random number of “normal circles” and “circle splatters” between 1 and 20.

Random colour optional method

```
randomColour = "hsl(" + 360*Math.random() + ",50%,50%)";
```


Task 2: “Spot” Generator

“Damien Steven Hirst is an English artist and the most prominent member of the group known as “Young British Artists” (or YBAs), who dominated the art scene in Britain during the 1990s” ... “Hirst's best known works are his paintings, medicine cabinet sculptures, and glass tank installations. For the most part, his paintings have taken on two styles. One is an arrangement of color spots with titles that refer to pharmaceutical chemicals, known as Spot paintings.”



```
9      <body>
10      <canvas id="sCanvas" width="550" height="550"></canvas>
11      <script type="text/javascript">
12
13      // Spot Painting Generator 550px x 550px
14      // Variables
15      var randomColour;
16      var xPos;
17      var yPos;
18      var i; // counter
19      var j; // counter
20      var c; // canvas element
21      var ctx;
22
23      // draw a circle with a specified size, position and colour
24      function drawCircle(size,x,y,colour){
25          ctx.beginPath();
26          ctx.arc(x,y,size/2,0,Math.PI*2,true);
27          ctx.fillStyle = colour;
28          ctx.fill();
29      }
30
31      // Get the Canvas element with ID sCanvas that will be drawn on
32      c = document.getElementById("sCanvas");
33      ctx = c.getContext("2d");
34
35      // Starting y position
36      yPos = 30;
37
38      // Starting x position
39      xPos = 30;
40
41      // Inner loop for 9 columns
42      for(j=0;j<9;j++){
43          // Generate random colour
44          randomColour = '#' + Math.random().toString(16).substring(2, 8);
45          drawCircle(30, xPos, yPos, randomColour);
46          xPos += 50;
47      }
48      yPos += 50;
49
50      </script>
51
52
```

The above code is the “skeleton” for a program that will generate a very simple version of a Damien Hirst Spot painting using a combination of functions and loops. You were shown the completed code during the lecture.

When you run this code you will notice that only one line of 9 horizontal “spots” appears. Your task is to use a nested loop to produce the full painting, which is comprised of 9 horizontal “spots” and 9 vertical “spots”.

You can do this by creating another for loop, which surrounds the current for loop (and xPos = 30 assignment) and uses a counter called i.

Loops in JavaScript

Introduction

In this week's lecture you were told about loops in JavaScript. In this tutorial you will explore these concepts by creating some simple programs that use the HTML Canvas

Task 1: "Art Balling" Generator

"Former England cricket captain Michael Vaughan has found a new use for that pull shot: whacking paint-daubed balls at canvas to produce highly collectable works of art. Watch out, abstract expressionism - here comes 'artballing'" –

The following code is the "skeleton" for a program that will generate a very simple version of a Michael Vaughan painting using a combination of functions and loops. You were shown the completed code during the lecture.



```
10 <canvas id="sCanvas" width="550" height="400"></canvas>
11 <script>
12 // Art Balling Generator 550px x 400px // Using Loops
13 // Variables
14 var randomColour;
15 var randomSize;
16 var xPos;
17 var yPos;
18 var i; // counter
19 var j; // counter
20 // Get the Canvas element with ID sCanvas that will be drawn on
21 c = document.getElementById("sCanvas");
22 ctx = c.getContext("2d");
23
24 function drawFilledCircle(size,xPos,yPos,colour){
25     ctx.beginPath();
26     ctx.arc(xPos,yPos,size,0,2*Math.PI);
27     ctx.fillStyle = colour;
28     ctx.fill();
29 }
30
31 function drawSplatter(size,xPos,yPos,colour){
32     for(j=0;j<10;j++){
33         var splatSize = size / Math.round(Math.random()*30);
34         drawFilledCircle(splatSize,xPos + Math.round(Math.random()*50),yPos + Math.round(Math.random()*50),colour);
35     }
36 }
37
38 </script>
39
```

This will not currently run as it requires code that utilises the given functions. Spend time first of all trying to understand what the 2 functions will do. You will notice that the `drawFilledCircle` function is very similar to the function you used in the last tutorial to create coloured squares. The `drawSplatter` function utilises a loop to draw 10 circles of random shapes and positions dependent on 4 parameters (`size,xPos,yPos,colour`) passed to the function.

Calling the functions in a loop

The following code can be used to create one “normal circle” and one “circle splatter”. Insert it around line 37 before the closing script tag.

```
randomSize = Math.round(Math.random()*50);
xPos = Math.round(Math.random()*550);
yPos = Math.round(Math.random()*400);
randomColour = '#' + Math.random().toString(16).substring(2, 8);
drawFilledCircle(randomSize, xPos, yPos, randomColour);
drawSplatter(randomSize, xPos, yPos, randomColour);
```

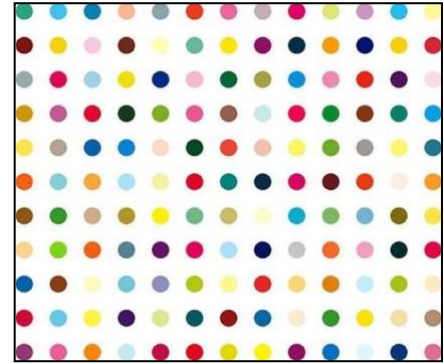
Your task is to take the above code and using either a for loop or a while loop, draw 10 “normal circles” and “circle splatters”. Once you have done this modify your code so that it produces a random number of “normal circles” and “circle splatters” between 1 and 20.

Random colour optional method

```
randomColour = "hsl(" + 360*Math.random() + ",50%,50%)";
```

Task 2: “Spot” Generator

“Damien Steven Hirst is an English artist and the most prominent member of the group known as “Young British Artists” (or YBAs), who dominated the art scene in Britain during the 1990s” ... “Hirst's best known works are his paintings, medicine cabinet sculptures, and glass tank installations. For the most part, his paintings have taken on two styles. One is an arrangement of color spots with titles that refer to pharmaceutical chemicals, known as Spot paintings.”



```
9      <body>
10     <canvas id="sCanvas" width="550" height="550"></canvas>
11     <script type="text/javascript">
12
13     // Spot Painting Generator 550px x 550px
14     // Variables
15     var randomColour;
16     var xPos;
17     var yPos;
18     var i; // counter
19     var j; // counter
20     var c; // canvas element
21     var ctx;
22
23     // draw a circle with a specified size, position and colour
24     function drawCircle(size,x,y,colour){
25         ctx.beginPath();
26         ctx.arc(x,y,size/2,0,Math.PI*2,true);
27         ctx.fillStyle = colour;
28         ctx.fill();
29     }
30
31     // Get the Canvas element with ID sCanvas that will be drawn on
32     c = document.getElementById("sCanvas");
33     ctx = c.getContext("2d");
34
35     // Starting y position
36     yPos = 30;
37
38     // Starting x position
39     xPos = 30;
40
41     // Inner loop for 9 columns
42     for(j=0;j<9;j++){
43         // Generate random colour
44         randomColour = '#' + Math.random().toString(16).substring(2, 8);
45         drawCircle(30, xPos, yPos, randomColour);
46         xPos += 50;
47     }
48     yPos += 50;
49
50     </script>
51
52
```

The above code is the “skeleton” for a program that will generate a very simple version of a Damien Hirst Spot painting using a combination of functions and loops. You were shown the completed code during the lecture.

When you run this code you will notice that only one line of 9 horizontal “spots” appears. Your task is to use a nested loop to produce the full painting, which is comprised of 9 horizontal “spots” and 9 vertical “spots”.

You can do this by creating another for loop, which surrounds the current for loop (and xPos = 30 assignment) and uses a counter called i.

Loops in JavaScript

Introduction

In this week's lecture you were told about loops in JavaScript. In this tutorial you will explore these concepts by creating some simple programs that use the HTML Canvas

Task 1: "Art Balling" Generator

"Former England cricket captain Michael Vaughan has found a new use for that pull shot: whacking paint-daubed balls at canvas to produce highly collectable works of art. Watch out, abstract expressionism - here comes 'artballing'" –

The following code is the "skeleton" for a program that will generate a very simple version of a Michael Vaughan painting using a combination of functions and loops. You were shown the completed code during the lecture.



```
10 <canvas id="sCanvas" width="550" height="400"></canvas>
11 <script>
12 // Art Balling Generator 550px x 400px // Using Loops
13 // Variables
14 var randomColour;
15 var randomSize;
16 var xPos;
17 var yPos;
18 var i; // counter
19 var j; // counter
20 // Get the Canvas element with ID sCanvas that will be drawn on
21 c = document.getElementById("sCanvas");
22 ctx = c.getContext("2d");
23
24 function drawFilledCircle(size,xPos,yPos,colour){
25     ctx.beginPath();
26     ctx.arc(xPos,yPos,size,0,2*Math.PI);
27     ctx.fillStyle = colour;
28     ctx.fill();
29 }
30
31 function drawSplatter(size,xPos,yPos,colour){
32     for(j=0;j<10;j++){
33         var splatSize = size / Math.round(Math.random()*30);
34         drawFilledCircle(splatSize,xPos + Math.round(Math.random()*50),yPos + Math.round(Math.random()*50),colour);
35     }
36 }
37
38 </script>
39
```

This will not currently run as it requires code that utilises the given functions. Spend time first of all trying to understand what the 2 functions will do. You will notice that the `drawFilledCircle` function is very similar to the function you used in the last tutorial to create coloured squares. The `drawSplatter` function utilises a loop to draw 10 circles of random shapes and positions dependent on 4 parameters (`size,xPos,yPos,colour`) passed to the function.

Calling the functions in a loop

The following code can be used to create one “normal circle” and one “circle splatter”. Insert it around line 37 before the closing script tag.

```
randomSize = Math.round(Math.random()*50);
xPos = Math.round(Math.random()*550);
yPos = Math.round(Math.random()*400);
randomColour = '#' + Math.random().toString(16).substring(2, 8);
drawFilledCircle(randomSize, xPos, yPos, randomColour);
drawSplatter(randomSize, xPos, yPos, randomColour);
```

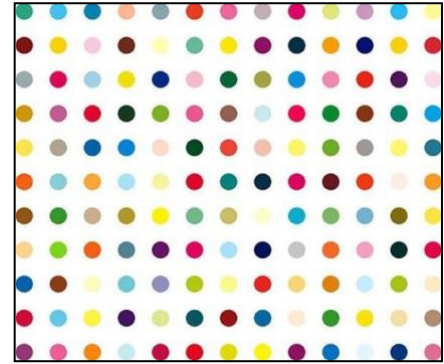
Your task is to take the above code and using either a for loop or a while loop, draw 10 “normal circles” and “circle splatters”. Once you have done this modify your code so that it produces a random number of “normal circles” and “circle splatters” between 1 and 20.

Random colour optional method

```
randomColour = "hsl(" + 360*Math.random() + ",50%,50%)";
```


Task 2: “Spot” Generator

“Damien Steven Hirst is an English artist and the most prominent member of the group known as “Young British Artists” (or YBAs), who dominated the art scene in Britain during the 1990s” ... “Hirst's best known works are his paintings, medicine cabinet sculptures, and glass tank installations. For the most part, his paintings have taken on two styles. One is an arrangement of color spots with titles that refer to pharmaceutical chemicals, known as Spot paintings.”



```
9      <body>
10      <canvas id="sCanvas" width="550" height="550"></canvas>
11      <script type="text/javascript">
12
13      // Spot Painting Generator 550px x 550px
14      // Variables
15      var randomColour;
16      var xPos;
17      var yPos;
18      var i; // counter
19      var j; // counter
20      var c; // canvas element
21      var ctx;
22
23      // draw a circle with a specified size, position and colour
24      function drawCircle(size,x,y,colour){
25          ctx.beginPath();
26          ctx.arc(x,y,size/2,0,Math.PI*2,true);
27          ctx.fillStyle = colour;
28          ctx.fill();
29      }
30
31      // Get the Canvas element with ID sCanvas that will be drawn on
32      c = document.getElementById("sCanvas");
33      ctx = c.getContext("2d");
34
35      // Starting y position
36      yPos = 30;
37
38      // Starting x position
39      xPos = 30;
40
41      // Inner loop for 9 columns
42      for(j=0;j<9;j++){
43          // Generate random colour
44          randomColour = '#' + Math.random().toString(16).substring(2, 8);
45          drawCircle(30, xPos, yPos, randomColour);
46          xPos += 50;
47      }
48      yPos += 50;
49
50      </script>
51
52
```

The above code is the “skeleton” for a program that will generate a very simple version of a Damien Hirst Spot painting using a combination of functions and loops. You were shown the completed code during the lecture.

When you run this code you will notice that only one line of 9 horizontal “spots” appears. Your task is to use a nested loop to produce the full painting, which is comprised of 9 horizontal “spots” and 9 vertical “spots”.

You can do this by creating another for loop, which surrounds the current for loop (and xPos = 30 assignment) and uses a counter called i.