HTML5/JavaScript Game Tutorial: Fifty Yard Dash

Have you ever seen a flip-book animation? It's a small tablet of paper with images which seem to move as you quickly flip through the pages. Video games work the same way, redrawing the whole screen 30 or even 60 times a second, with small changes each time, giving the illusion of movement.

Knowing that helps us organize our program. There will be some code which gets called on only once: to set up the game, there will be code responsible for clearing and redrawing each 'frame' 30 times a second, and code for keeping track of information, like where the objects should be drawn.

To get started, let's create a simple HTML File. You will need one element in the body, a 'canvas' element, and after it a JavaScript include tag which will pull in our JavaScript game file:

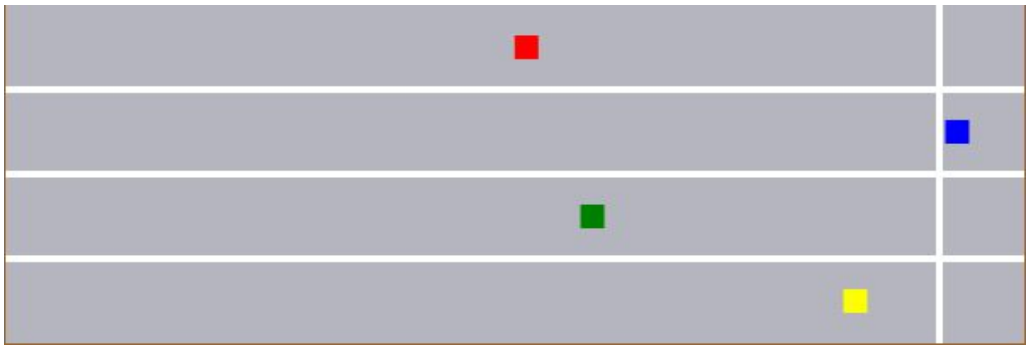
```
<canvas id="canvas"></canvas>
<script type="text/javascript" src="js/game.js"></script>
```

Now, let's create 'game.js' in the same folder, or directory as the HTML page. We are going to start out by setting up some variables and hooking into our canvas element so that our JavaScript has something to draw on.

```
var canvas = document.getElementById('canvas');
canvas.style.backgroundColor = '#b5b5c0'
canvas.width = 600;
canvas.height = 200;

var ctx = canvas.getContext("2d");
var clicks = 0;
var red, green, blue, yellow, gameLoop;
```

At this point, when you open the web page and refresh it, you should have a plain grey rectangle, 600 x 200 pixels. This is a running game, so let's draw some lanes. In Chrome, right-click, select 'inspect' to watch the Javascript Console for errors. The errors will give you hints with line numbers if something goes wrong.



```
function Runner(color, position) {
  this.color = color;
  this.x = 20;
  this.y = position * 50 + 18;
  this.speed = Math.random() * 2 + 2;
}

function drawRunner(runner) {
  red.speed = clicks
  clicks = Math.max(clicks - .05, 0)
  runner.x += runner.speed;
  ctx.fillStyle = runner.color;
  ctx.beginPath();
  ctx.rect(runner.x, runner.y, 14, 14)
  ctx.fill()
}

function drawLanes() {
  ctx.beginPath();
  ctx.strokeStyle = '#ffffff';
  ctx.lineWidth = 4;
  ctx.moveTo(0, 50);
  ctx.lineTo(600, 50);
  ctx.moveTo(0, 100);
  ctx.lineTo(600, 100);
  ctx.moveTo(0, 150);
  ctx.lineTo(600, 150);
  ctx.moveTo(550, 0)
  ctx.lineTo(550, 200)
  ctx.stroke();
}

drawLanes();

function finishLine() {
  if (Math.max(red.x, green.x, blue.x, yellow.x) >=
canvas.width - 48) {
    window.clearInterval(gameLoop)
    window.setTimeout(function() {
      started = false;
    }, 1000)
  }
}
```

Runner is a constructor function. We are using it to set up our 4 runners with default properties of color, position, and speed (which will be a random number between 2 and 4.

The drawRunner function will draw each runner on each frame, based on the information passed to it. The function will have one parameter, a a variable passed to it which it's calling 'runner' (top line).

The numbers are 'x and y coordinates' . They tell the browser: go this many pixels to the right, now this many pixels down and draw a line. First we define the function. We called it drawLanes. All of the function's instructions are inside the two curly braces '{}' The last line 'drawLanes()', calls the function. Nothing happens untill it is called.

This function, when it's called, will check to see if any of the runners (named red, green, blue, or yellow) have passed the finish line. If they have, our game loop gets stopped.

```
function drawFrame() {
  ctx.clearRect(0, 0, canvas.width, canvas.height);
  drawLanes();
  drawRunner(red);
  drawRunner(green);
  drawRunner(blue);
  drawRunner(yellow);
  finishLine();
}

function start() {
  clicks = 0;
  started = true;
  gameLoop = window.setInterval(drawFrame, 33);
  red = new Runner('red', 0);
  blue = new Runner('blue', 1);
  green = new Runner('green', 2);
  yellow = new Runner('yellow', 3);
}

var started = false;
document.body.onkeyup = function(event) {
  if (started === true && event.which === 13) {
    clicks += 1
  } else if (started === false) {
    start()
  }
};
```

Here's the actual game loop. Do you recognize those function calls? It's all the functions we've been defining. Now they each will get called every time 'drawFrame gets called

.

The start function sets everything up and calls the game loop. The setInterval() function is a built in JavaScript function. It will call our 'drawFrame' once every 33 milliseconds; about 30 times a second.

This bit of code listens for a key which has been pressed to be released: 'keyup'. '13' is the keycode for the Enter key. If the game has started, they key will register a click and help the player run, otherwise it will start the game.