

```

1  <html>
2      <head>
3          <title>Sub Program</title>
4
5          <script language="javascript">
6
7              var colorOne = "#0000FF";
8
9              //defines the boundaries of the canvas as varaibles to make it easier to hot
              swap
10             var canvasBoundX = 700;
11             var canvasBoundY = 500;
12
13             var speed = 6;
14
15             //defines initial position of the sub
16             var xSub = 350;
17             var ySub = 200;
18
19             var currentMovement = "none";
20             var previousMovement = "none";
21
22             var isOn = false;
23             var time = 0;
24
25             var health = 2000;
26             //used for timer to determine if it is on or not
27
28             //an array to hold all the fishes
29             var fishes = new Array();
30
31             class Fish{
32
33                 constructor(x, y, image, direction, speedX, speedY){
34                     this.x = x;
35                     this.y = y;
36                     this.image = image;
37                     this.direction = direction;
38                     this.speedX = speedX;
39                     this.speedY = speedY;
40                 }
41
42                 setX(newX){
43                     this.x = newX - 50;
44                 }
45
46                 setY(newY){
47                     this.y = newY - 50;
48                 }
49
50                 getX(){
51                     return this.x + 50;
52                 }
53
54                 getY(){
55                     return this.y + 50;
56                 }
57
58                 setSpeedX(speedX){
59                     this.speedX = speedX;
60
61                     if(speedX > 0){
62                         this.setDirection("Right");
63                     }
64                     else if(speedX < 0){
65                         this.setDirection("Left");
66                     }

```

```

67     }
68
69     getSpeedX(){
70         return this.speedX;
71     }
72
73     setSpeedY(speedY){
74         this.speedY = speedY;
75     }
76
77     getSpeedY(){
78         return this.speedY;
79     }
80
81     setDirection(direction){
82         this.direction = direction;
83     }
84
85     getImage(){
86         return this.image + this.direction;
87     }
88 }
89
90 //for loop to make a bunch of fishes
91 for(var i = 0; i < 6; i++){
92
93     var xCoord = Math.floor(Math.random() * (canvasBoundX - 200)) + 100;
94     var yCoord = Math.floor(Math.random() * (canvasBoundY - 200)) + 100;
95
96     var colorIndex = Math.floor(Math.random() * 9) + 1;
97     var image = "fish" + colorIndex;
98
99     //var direction = "Right";
100    var fishSpeedX = Math.floor(Math.random() * 4) - 2;
101    var fishSpeedY = Math.floor(Math.random() * 4) - 2;
102
103    if(fishSpeedX > 0)
104        direction = "Right";
105    else if(fishSpeedX < 0)
106        direction = "Left";
107    else{
108        fishSpeedX = 2;
109        direction = "Right";
110    }
111
112
113    fishes[i] = new Fish(xCoord, yCoord, image, direction, fishSpeedX,
114        fishSpeedY);
115
116    var smallFish = new Fish(300, 300, "littleFish", "Left", 3, 3)
117
118
119    //adds keyboard listener
120    window.addEventListener("keydown", function(event){
121        //changes keyboard input into diretions
122        switch(event.key){
123            case "a":
124                currentMovement = "left";
125                break;
126            case "d":
127                currentMovement = "right";
128                break;
129            case "s":
130                currentMovement = "down";
131                break;
132            default:

```

```

133         currentMovement = "none";
134         break;
135     }
136
137     if(currentMovement != "none" && currentMovement != "down")
138         previousMovement = currentMovement;
139
140 }, true);
141
142 window.addEventListener("keyup", function(event){
143     currentMovement = "none";
144
145 }, true);
146
147
148 //fill background and turn on timer when the body gets initialized
149 function initialize(){
150     var canvas = document.getElementById("myCanvas");
151     var context = canvas.getContext("2d");
152
153     context.fillStyle="#ADD8E6";
154     context.fillRect(0, 0, canvasBoundX, canvasBoundY);
155
156     turnOn();
157 }
158
159 //periodically called ever 20ms to update the game
160 function update(){
161     time += .02;
162     resetBackground();
163     controlSub();
164     moveFish();
165     decayHealth();
166     writeText(Math.floor(time * 100) / 100, 20, 40);
167     writeText(health, 20, 80);
168
169     if(health < 0){
170         writeText("GAME OVER", 300, 300);
171         writeText("You lasted: " + (Math.floor(time * 100) / 100) + "
172         seconds!", 300, 400);
173         turnOff();
174     }
175 }
176
177 function drawSub(x, y, direction){
178     var canvas = document.getElementById("myCanvas");
179     var context = canvas.getContext("2d");
180
181     var image;
182
183     if(direction == "right")
184         image = document.getElementById("sub right");
185     else
186         image = document.getElementById("sub left");
187
188     context.drawImage(image, x - 50, y - 50, 120, 75);
189 }
190
191 function drawImage(id, x, y, width, height){
192     var canvas = document.getElementById("myCanvas");
193     var context = canvas.getContext("2d");
194
195     var image = document.getElementById(id);
196     context.drawImage(image, x, y, width, height);
197 }
198
199 //writes text on the screen given what to write and the location

```

```

199     function writeText(text, x, y){
200         var canvas = document.getElementById("myCanvas");
201         var context = canvas.getContext("2d");
202
203         context.font = "30px Arial";
204         context.fillStyle = "#FF0000";
205         context.fillText(text, x, y);
206     }
207
208     //moves the sub based on the given direction
209     function controlSub(){
210
211         switch(currentMovement){
212             case "left":
213                 if(xSub > 40)
214                     xSub -= speed;
215                 drawSub(xSub, ySub, previousMovement);
216                 break;
217             case "right":
218                 if(xSub < canvasBoundX - 60)
219                     xSub += speed;
220                 drawSub(xSub, ySub, previousMovement);
221                 break;
222             case "down":
223                 if(ySub <= canvasBoundY - 15)
224                     ySub += speed;
225                 drawSub(xSub, ySub, previousMovement);
226                 break;
227             default:
228
229                 drawSub(xSub, ySub, previousMovement);
230                 break;
231         }
232
233         if(currentMovement != "down" && ySub > 40)
234             ySub -= speed / 5;
235     }
236
237     //used to reset the background to avoid smearing
238     function resetBackground(){
239         var canvas = document.getElementById("myCanvas");
240         var context = canvas.getContext("2d");
241         //paint the background of the canvas
242         context.fillStyle="#ADD8E6";
243         context.fillRect(0, 0, canvasBoundX, canvasBoundY);
244     }
245
246     //places the defender to a given location and moves the turret based on the
    direction
247     function moveSub(x, y, direction){
248         var canvas = document.getElementById("myCanvas");
249         var context = canvas.getContext("2d");
250
251         var directX = 0;
252         var directY = 0;
253
254         context.fillStyle = colorOne;
255         context.beginPath();
256         context.arc(x, y, 15, 0, 2 * Math.PI, true);
257         context.closePath();
258         context.fill();
259
260         if(direction == "left")
261             directX = -1;
262         else if(direction == "right")
263             directX = 1;
264

```

```

265         context.fillStyle = colorOne;
266         context.beginPath();
267         context.arc(x + (15 * directX), y, 7.5, 0, 2 * Math.PI, true);
268         context.closePath();
269         context.fill();
270     }
271
272     function moveFish(){
273
274         for(var i = 0; i < fishes.length; i++){
275             var currentFish = fishes[i];
276
277             if(time % 2 < 0.1){
278                 currentFish.setSpeedX((Math.random() - 0.5) * 4);
279                 currentFish.setSpeedY((Math.random() - 0.5) * 4);
280             }
281
282             if((currentFish.getX() < 25) || (currentFish.getX() > canvasBoundX -
283                 25))
284                 currentFish.setSpeedX(currentFish.getSpeedX() * -1);
285
286             if((currentFish.getY() < 10 && currentFish.getSpeedY() < 0) || (
287                 currentFish.getY() > canvasBoundY - 10 && currentFish.getSpeedY() > 0
288                 ))
289                 currentFish.setSpeedY(currentFish.getSpeedY() * -1);
290
291             if(Math.abs(currentFish.getX() - xSub) < 25 && previousMovement ==
292                 "right")
293                 currentFish.setSpeedX(6);
294             else if(Math.abs(currentFish.getX() - xSub) < 25 && previousMovement
295                 == "left")
296                 currentFish.setSpeedX(-6);
297
298             if(Math.abs(currentFish.getY() - ySub) < 25 && currentMovement ==
299                 "down")
300                 currentFish.setSpeedY(6);
301             else if(Math.abs(currentFish.getY() - ySub) < 25 && currentMovement
302                 != "down")
303                 currentFish.setSpeedY(-6);
304
305             currentFish.setX(currentFish.getX() + currentFish.getSpeedX());
306             currentFish.setY(currentFish.getY() + currentFish.getSpeedY());
307         }
308
309         moveSmallFish();
310         drawFish();
311     }
312
313     function moveSmallFish(){
314         var currentFish = smallFish;
315
316         if(time % 2 < 0.1){
317             currentFish.setSpeedX((Math.random() - 0.5) * 4);
318             currentFish.setSpeedY((Math.random() - 0.5) * 4);
319         }
320
321         if((currentFish.getX() < 25) || (currentFish.getX() > canvasBoundX -
322             25))
323             currentFish.setSpeedX(currentFish.getSpeedX() * -1);
324
325         if((currentFish.getY() < 10 && currentFish.getSpeedY() < 0) || (
326             currentFish.getY() > canvasBoundY - 10 && currentFish.getSpeedY() > 0
327             ))
328             currentFish.setSpeedY(currentFish.getSpeedY() * -1);

```

```

322         currentFish.setX(currentFish.getX() + currentFish.getSpeedX());
323         currentFish.setY(currentFish.getY() + currentFish.getSpeedY());
324     }
325
326     function drawFish(){
327         var canvas = document.getElementById("myCanvas");
328         var context = canvas.getContext("2d");
329
330         for(var i = 0; i < fishes.length; i++){
331             if(fishes[i])
332
333                 var image = document.getElementById(fishes[i].getImage());
334
335                 context.drawImage(image, fishes[i].x, fishes[i].y, 100, 100);
336         }
337
338         var image = document.getElementById(smallFish.getImage());
339         context.drawImage(image, smallFish.x, smallFish.y, 75, 75);
340     }
341
342     function decayHealth(){
343         for(var i = 0; i < fishes.length; i++){
344             var currentFish = fishes[i];
345
346             if(Math.abs(currentFish.getX() - smallFish.getX()) < 50 && Math.abs(
347                 currentFish.getY() - smallFish.getY()) < 50)
348                 health -= 20;
349         }
350     }
351
352     //toggles the timer
353     function toggleTimer(){
354         //if the timer is on the turn it off, if its not then turn it on
355         isOn ? turnOff() : turnOn();
356     }
357
358     //manually turns on the timer
359     function turnOn(){
360         //turns the timer on
361         timer = setInterval("update()", 20);
362         isOn = true;
363     }
364
365     //manually turns off the timer
366     function turnOff(){
367         //turns the timer off
368         clearInterval(timer);
369         isOn = false;
370     }
371 </script>
372 </head>
373
374 <body onload="initialize()">
375     <center>
376         <h1>Sub Program</h1>
377
378     </br>
379     </br>
380
381     <canvas id="myCanvas" width="700" height="500"
382         style="border:2px solid rgb(195, 195, 195);">
383         Your browser does not support the canvas element
384     </canvas>
385
386     

```

```
387     
389     
391     
393     
395     
397     
399     
401     
403     
405     
407     
409
410     
412     
414     
416     
418     
420     
422     
424     
426     
428     
430
431     
433     
435
436 </br>
437 </br>
438 </center>
439 </body>
440 </html>
```