

sketch.js

```
1  /* Project part 4 – Midterms
2  Student: Timna Aversa
3  Class: Introduction to Programming */
4
5
6  var floorPos_y = 432;
7  var max_x = 2000;
8  var cameraPosX;
9
10 var gameChar_x;
11 var gameChar_y;
12
13 var trees_x;
14
15 var canyon;
16 var collectable;
17 var mountain;
18 var cloud;
19
20 var isLeft;
21 var isRight;
22 var isFalling;
23 var isPlummeting;
24
25 // I know the video commented that having things separated by functions was one of
26 // the give
27 // ways for plagiarism but since my code was already organized in this set up 3
28 // hours didn't
29 // seem like enough time to rewrite it all and make sure it worked. I did wrote
30 // the whole thing!
31
32 //character set up
33 //Character hair set up
34 function hair(gameChar_x,gameChar_y,size)
35 {
36     fill(210, 83, 128);
37     ellipse(gameChar_x - 0.18 * size,gameChar_y- 4.6 * size,0.45 * size,0.45 *
size);
38     ellipse(gameChar_x - 0.4 * size,gameChar_y- 4.3 * size,0.4 * size,0.4 * size);
39     ellipse(gameChar_x,gameChar_y- 4.3 * size,0.5 * size,0.4 * size);
40     ellipse(gameChar_x + 0.18 * size,gameChar_y- 4.6 * size,0.45 * size,0.45 *
size);
41     ellipse(gameChar_x + 0.4 * size,gameChar_y- 4.3 * size,0.4 * size,0.4 * size);
42 }
43 //Character eye set up
44 function eye(gameChar_x,gameChar_y,size,direction)
45 {
46     dirNum = 0;
47     if (direction === "left")
48     {
49         dirNum = -3;
50     }
51     if (direction === "right")
52     {
53         dirNum = 3;
54     }
55 }
```

```

53     fill(255);
54     ellipse(gameChar_x + dirNum ,gameChar_y -3.1 * size,2.3 * size,2.3 * size);
55     fill(0);
56     ellipse(gameChar_x + dirNum * 2.4,gameChar_y - 3.1 * size+ dirNum * 0.2,1.4 *
size,1.4 * size);
57     fill(255);
58     ellipse(gameChar_x + dirNum * 3.2,gameChar_y - 3.25 * size,0.7 * size,0.7 *
size);
59     ellipse(gameChar_x - 2 + dirNum,gameChar_y - 2.7 * size,0.3 * size,0.3 *
size);
60 }
61 //Character mouth set up
62 function mouth(gameChar_x,gameChar_y,size,direction)
63 {
64     dirNum = 0;
65     if (direction === "left")
66     {
67         dirNum = -4.2;
68     }
69     if (direction === "right")
70     {
71         dirNum = 4.2;
72     }
73     triangle(gameChar_x - 0.8 * size + dirNum,
74         gameChar_y - 1.3 * size,
75         gameChar_x - 0.6 * size +dirNum,
76         gameChar_y - 1.6 * size,
77         gameChar_x - 0.4 * size+dirNum,
78         gameChar_y - 1.3 * size);
79     triangle(gameChar_x + 0.8 * size + dirNum,
80         gameChar_y - 1.3 * size,
81         gameChar_x + 0.6 * size + dirNum,
82         gameChar_y - 1.6 * size,
83         gameChar_x + 0.4 * size + dirNum,
84         gameChar_y - 1.3 * size);
85     stroke(147, 118, 224);
86     line(gameChar_x - 0.8 * size + dirNum,
87         gameChar_y - 1.3 * size,
88         gameChar_x + 0.8 * size + dirNum,
89         gameChar_y - 1.3 * size);
90 }
91 //Character head set up
92 function headBackground(gameChar_x,gameChar_y,size)
93 {
94     rightHorn(gameChar_x,gameChar_y,size);
95     leftHorn(gameChar_x,gameChar_y,size);
96     fill(255,116,177);
97     ellipse(gameChar_x,gameChar_y-2.4 * size,4 * size,4 * size);
98     noStroke();
99     fill(128, 70, 116, 60);
100     arc(gameChar_x,gameChar_y-2.4 * size,4 * size,4 * size,4.6,1.2,24);
101     fill(255,116,177);
102     ellipse(gameChar_x,gameChar_y-2.4 * size,3.4 * size,4 * size);
103 }
104 //Character position of the leg for standing still set up
105 function stand(gameChar_x,gameChar_y,size,side)
106 {
107     beginShape();
108     vertex(gameChar_x + 0.3 * size * side, gameChar_y);
109     vertex(gameChar_x + 0.35 * size * side, gameChar_y - 0.5 * size);
110     vertex(gameChar_x + 0.8 * size * side, gameChar_y - 0.7 * size);

```

```

111     vertex(gameChar_x + 1 * size * side, gameChar_y);
112     endShape(CLOSE);
113 }
114 // Character making the call for each function according to the leg movement, call
115 // made for both legs.
116 function legs(gameChar_x, gameChar_y, size, action, direction)
117 {
118     fill(255, 116, 177);
119     noStroke();
120     dirNum = 0;
121     if (action === "stand")
122     {
123         stand(gameChar_x, gameChar_y, size, -1);
124         stand(gameChar_x, gameChar_y, size, 1);
125     }
126     if (action === "jump")
127     {
128         jump(gameChar_x, gameChar_y, size, -1, 0);
129         jump(gameChar_x, gameChar_y, size, 1, 0);
130     }
131     if (action === "walk" & direction === "left")
132     {
133         walk(gameChar_x, gameChar_y, size, 1, -12);
134         walk(gameChar_x, gameChar_y, size, 1, 0);
135     }
136     if (action === "walk" & direction === "right")
137     {
138         walk(gameChar_x, gameChar_y, size, -1, +12);
139         walk(gameChar_x, gameChar_y, size, -1, 0);
140     }
141 }
142 //Character position of the leg for walking set upx
143 function walk(gameChar_x, gameChar_y, size, side, direction)
144 {
145     beginShape();
146     vertex(gameChar_x + 0.6 * size * side + direction, gameChar_y);
147     vertex(gameChar_x + 0.35 * size * side + direction, gameChar_y - 0.2 * size);
148     vertex(gameChar_x + 0.4 * size * side + direction, gameChar_y - 0.5 * size);
149     vertex(gameChar_x + 0.9 * size * side + direction, gameChar_y - 0.6 * size);
150     vertex(gameChar_x + 1 * size * side + direction, gameChar_y - 0.4 * size);
151     vertex(gameChar_x + 1.4 * size * side + direction, gameChar_y - 0.2 * size);
152     endShape(CLOSE);
153 }
154 //Character position of the leg for jumping and falling set up
155 function jump(gameChar_x, gameChar_y, size, side, direction)
156 {
157     beginShape();
158     stroke(128, 70, 116, 60);
159     vertex(gameChar_x + 0.9 * size * side + direction, gameChar_y - 0.2 * size +
direction);
160     vertex(gameChar_x + 0.8 * size * side + direction, gameChar_y - 0.55 * size +
direction);
161     vertex(gameChar_x + 1.3 * size * side + direction, gameChar_y - 1 * size +
direction);
162     vertex(gameChar_x + 2 * size * side + direction, gameChar_y - 0.7 * size +
direction);
163     endShape(CLOSE);
164     noStroke();
165 }
166 //Character right horn set up
167 function rightHorn(gameChar_x, gameChar_y, size)

```

```
168 {
169   fill(255);
170   arc(gameChar_x + 0.15 * size, gameChar_y - 3.8 * size, 3 * size, 3 * size, 6, 2, PI);
171 }
172 //Character left horn set up
173 function leftHorn(gameChar_x, gameChar_y, size)
174 {
175   fill(255);
176   arc(gameChar_x - 0.15 * size, gameChar_y - 3.8 * size, 3 * size, 3 * size, 45, 3.5,
PI);
177 }
178 // Call functions to draw all parts of the character
179 function characterMove(gameChar_x, gameChar_y, size, move, direction)
180 {
181   legs(gameChar_x, gameChar_y, size, move, direction);
182   headBackground(gameChar_x, gameChar_y, size);
183   hair(gameChar_x, gameChar_y, size);
184   eye(gameChar_x, gameChar_y, size, direction);
185   mouth(gameChar_x, gameChar_y, size, direction);
186 }
187 //game view
188 //Random generation of numbers function for simplification of code
189 function randNumb(maxNumber)
190 {
191   return Math.floor(Math.random() * maxNumber);
192 }
193 // Draw a tree
194 function tree(x, y, size)
195 {
196   noStroke();
197   y = 432 - size * 6;
198   fill(60, 35, 23);
199   rect(x - size, y - size * 2, size * 2, size * 8);
200   fill(46, 79, 79);
201   stroke(44, 51, 51);
202   triangle(x + size * 10, y, x, y - size * 26, x - size * 10, y);
203   noStroke();
204   fill(44, 51, 51);
205   beginShape();
206   vertex(x + size * 8, y - size * 5);
207   vertex(x, y - size * 13);
208   vertex(x - size * 8, y - size * 5);
209   vertex(x, y - size * 11);
210   endShape();
211 }
212 // Draw a mountain
213 function mountain(x, y, size)
214 {
215   y = 432;
216   fill(212, 173, 252);
217   triangle(x, y, x, y - size * 26, x + size * 10, y);
218   fill(92, 70, 156);
219   triangle(x, y, x, y - size * 26, x - size * 10, y);
220   fill(255);
221   beginShape();
222   vertex(x, y - size * 18);
223   vertex(x, y - size * 26);
224   vertex(x + size * 3, y - size * 18);
225   vertex(x + size / 2, y - size * 20);
226   endShape();
```

```

227     fill(210);
228     beginShape();
229     vertex(x, y-size*18);
230     vertex(x, y-size*26);
231     vertex(x-size*3, y- size*18);
232     vertex(x-size/2,y- size*20);
233     endShape();
234 }
235 // Draw a star
236 function star(x,y,size)
237 {
238     i = randNumb(5)
239     if (i==0){fill(255, 95, 158);}
240     else if (i==1){fill(233, 0, 100);}
241     else if (i==2){fill(249, 217, 73);}
242     else if (i==3){fill(240, 240, 240);}
243     else if (i==4){fill(58, 180, 242);}
244     else{fill(39, 225, 193);}
245     ellipse(x,y,1,1);
246 }
247 // Draw a rock
248 function rock(x,y,size)
249 {
250     // console.log(y)
251     size = size + 2;
252     i = size % 3
253     if (i==0){fill(65, 53, 67,50);}
254     else if (i==1){fill(240, 235, 141,50);}
255     else{fill(143, 67, 238, 50);}
256     ellipse(x,floorPos_y + 25 + y,size,size-3);
257 }
258 // Draw a cloud
259 function cloud(x,y,size)
260 {
261     noStroke();
262     fill(255);
263     rect(x + size * 4, y + size/2 * 10, size * 13, size * 3, size*2);
264     rect(x + size * 10, y + size/2 *5, size * 5, size * 4, size*2);
265     rect(x + size * 7, y + size, size * 4.5, size * 6, size*2);
266 }
267 // Draw a token
268 function token(x,y,size)
269 {
270     noStroke();
271     fill(255, 211, 163);
272     triangle(x - size,y,x,y + 2.5 * size,x + size,y);
273     fill(225, 18, 153);
274     arc(x, y-1, 2.3 * size, 3.3 * size, PI, 0 , CHORD);
275 }
276 // Draw a canyon
277 function canyonDraw(x,size)
278 {
279     noStroke();
280     fill(169, 113, 85);
281     rect(x,floorPos_y,size,200);
282 }
283 //initiating classes of random objects positioning
284 class positionClass
285 {
286     constructor(maxX,maxY,maxSize) {

```

```

287     this.x = randNumb(maxX);
288     this.y = randNumb(maxY);
289     this.size = randNumb(maxSize);
290 }
291 }
292 // object to control the random generated objects
293 let sceneryObjs = {rocks: {'rand':850, 'x':max_x, 'y':120, 'size':8, 'obj':[]},
294                     stars: {'rand':1050, 'x':max_x, 'y':425, 'size':8, 'obj':[]},
295                     mountains: {'rand':7, 'x':max_x, 'y':300, 'size':8, 'obj':[]},
296                     trees: {'rand':8, 'x':max_x, 'y':0, 'size':6, 'obj':[]},
297                     clouds: {'rand':7, 'x':max_x, 'y':100, 'size':8, 'obj':[]}}
298 };
299
300 // Because I wanted something that would look prettier I had created a code in
301 // which
302 // the positions for the items are randomly generated and assigned to an array,
303 // since
304 // the request was for generate an array of objects for the clouds and mountains I
305 // believe
306 // this code still fulfill the purpose. So I only recreated an array for trees.
307 function drawObjectsInArray(array_obj, obj_type)
308 {
309     for (let i = 0; i < array_obj.length; i++)
310     {
311         switch (obj_type) {
312             // case 'trees':
313             //     tree(array_obj[i].x, array_obj[i].y, array_obj[i].size);
314             //     break;
315             case 'rocks':
316                 rock(array_obj[i].x, array_obj[i].y, array_obj[i].size);
317                 break;
318             case 'stars':
319                 star(array_obj[i].x, array_obj[i].y, array_obj[i].size);
320                 break;
321             case 'clouds':
322                 cloud(array_obj[i].x, array_obj[i].y, array_obj[i].size);
323                 break;
324             case 'mountains':
325                 mountain(array_obj[i].x, array_obj[i].y, array_obj[i].size);
326                 break;
327             default:
328
329         }
330     }
331 }
332
333 //game main
334 function setup()
335 {
336     createCanvas(1024, 576);
337     cameraPosX = 0;
338     gameChar_x = 200;
339     gameChar_y = floorPos_y;
340
341     isLeft = false;
342     isRight = false;
343     isFalling = false;
344     isPlummeting = false;
345     // making the requested array for the trees
346     trees_x = [50, 300, 500, 800, 1000, 1200];
347
348     //generate objects in the arrays

```

```
345     Object.entries(sceneryObjs).forEach(([key, value]) => {
346         for (let j = 0; j < value.rand; j++)
347             {
348                 sceneryObjs[key].obj.push(new
positionClass(value.x,value.y,value.size));
349             }
350     });
351
352     canyon = {x_pos: width - 400, width: 70}
353     collectable = {
354         x_pos: width - 150,
355         y_pos: 410,
356         size: 7,
357         isFound: false
358     }
359 }
360
361 function draw()
362 {
363     background(6, 0, 71);
364     noStroke();
365     fill(26, 95, 122);
366     rect(0, 432, 1024, 20);
367     fill(5, 45, 72);
368     rect(0,452,1024, 120);
369     push();
370     translate(-cameraPosX, 0);
371     // drawing each object in the arrays
372     Object.entries(sceneryObjs).forEach(([key, value]) => {
373         drawObjectsInArray(sceneryObjs[key].obj,key);
374     });
375     // looping through the array of trees positions
376     for(var i = 0; i < trees_x.length; i++)
377     {
378         tree(trees_x[i],floorPos_y,5);
379     }
380
381     canyonDraw(canyon.x_pos,canyon.width);
382     // controlling canyon and token interaction with character
383     if (dist(cameraPosX + gameChar_x,gameChar_y,collectable.x_pos,
collectable.y_pos) < 25)
384     {
385         collectable.isFound = true;
386     }
387     if (collectable.isFound == false)
388     {
389         token(collectable.x_pos,collectable.y_pos,collectable.size);
390     }
391     if((canyon.x_pos + canyon.width > cameraPosX + gameChar_x) && (cameraPosX +
gameChar_x > canyon.x_pos) && gameChar_y >= floorPos_y)
392     {
393         isPlummeting = true;
394     }
395
396     //character and camera position control
397     if (isPlummeting == true)
398     {
399         gameChar_y += 8;
400     }
401     else if (isLeft == true)
402     {
```

```
403     cameraPosX -= 5;
404 }
405 else if (isRight == true)
406 {
407     cameraPosX += 5;
408 }
409
410 if (gameChar_y < floorPos_y)
411 {
412     gameChar_y += 4;
413     isFalling = true;
414 }
415 else
416 {
417     isFalling = false;
418 }
419
420 pop();
421
422 //character design
423 if (isLeft && isFalling)
424 {
425     characterMove(gameChar_x,gameChar_y,10,'jump','left');
426 }
427 else if (isRight && isFalling)
428 {
429     characterMove(gameChar_x,gameChar_y,10,'jump','right');
430 }
431 else if (isLeft)
432 {
433     characterMove(gameChar_x,gameChar_y,10,'walk','left');
434 }
435 else if (isRight)
436 {
437     characterMove(gameChar_x,gameChar_y,10,'walk','right');
438 }
439 else if (isFalling || isPlummeting)
440 {
441     characterMove(gameChar_x,gameChar_y,10,'jump','');
442 }
443 else
444 {
445     characterMove(gameChar_x,gameChar_y,10,'stand','');
446 }
447
448 }
449 // user actions control
450 function keyPressed()
451 {
452     if (isPlummeting == false)
453     {
454         if (keyCode == 37)
455         {
456             isLeft = true;
457         }
458         else if (keyCode == 39)
459         {
460             isRight = true;
461         }
462         else if ((keyCode == 32) && (isFalling != true))
```



```
463     {
464         gameChar_y -= 100;
465     }
466     else
467     {
468         console.log(key)
469     }
470 }
471
472 }
473 function keyReleased()
474 {
475     if (keyCode == 37)
476     {
477         isLeft = false;
478     }
479     else if (keyCode == 39)
480     {
481         isRight = false;
482     }
483 }
484
```