



# UNIVERSITI MALAYA

## WIF3005 - Software Maintenance Evolution Semester 1, Session 2024/2025

### ALTERNATIVE ASSESSMENT

#### Question 1

By:  
Adeline Kong Earn Ning  
22004762

<b>QUESTION 1: Reusable component.....</b>	<b>2</b>
<a href="https://github.com/jakesgordon/javascript-racer">https://github.com/jakesgordon/javascript-racer</a> .....	2
1. BACKGROUND (javascript-racer/images/background.js).....	2
a. Description:.....	2
b. Snapshots:.....	2
c. Usage examples in other projects or contexts.....	3
2. Dom (javascript-racer/common.js).....	4
a. Description:.....	4
b. Snapshots:.....	5
c. Usage examples in other projects or contexts.....	6

# QUESTION 1: Reusable component

<https://github.com/jakesgordon/javascript-racer>

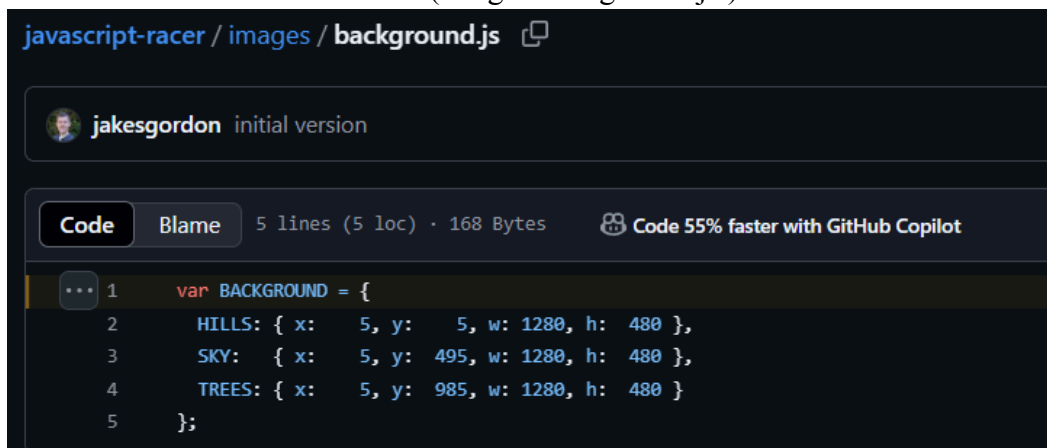
## 1. BACKGROUND (javascript-racer/images/background.js)

### a. Description:

The BACKGROUND object defines a collection of reusable properties for the coordinates for the various background layers in a graphical context. It defines the coordinates for the hills, sky, and trees when rendering the background of the game. This is so that the background layers look uniform in all game modes. (straight, hills, curves, final)

### b. Snapshots:

#### i. BACKGROUND (images/background.js )



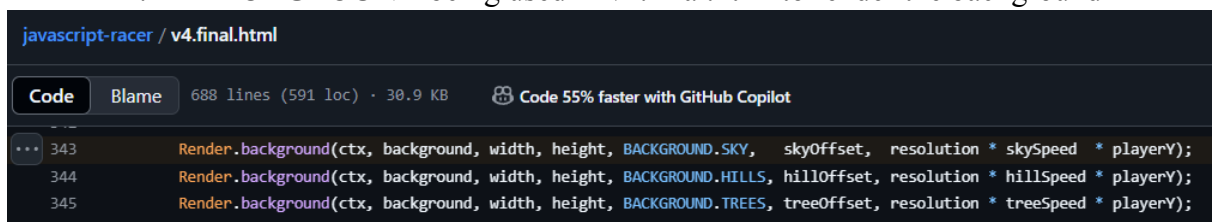
```
javascript-racer / images / background.js

jakesgordon initial version

Code Blame 5 lines (5 loc) · 168 Bytes Code 55% faster with GitHub Copilot

1 var BACKGROUND = {
2   HILLS: { x: 5, y: 5, w: 1280, h: 480 },
3   SKY: { x: 5, y: 495, w: 1280, h: 480 },
4   TREES: { x: 5, y: 985, w: 1280, h: 480 }
5 };
```

#### ii. BACKGROUND being used in v4.final.html to render the background

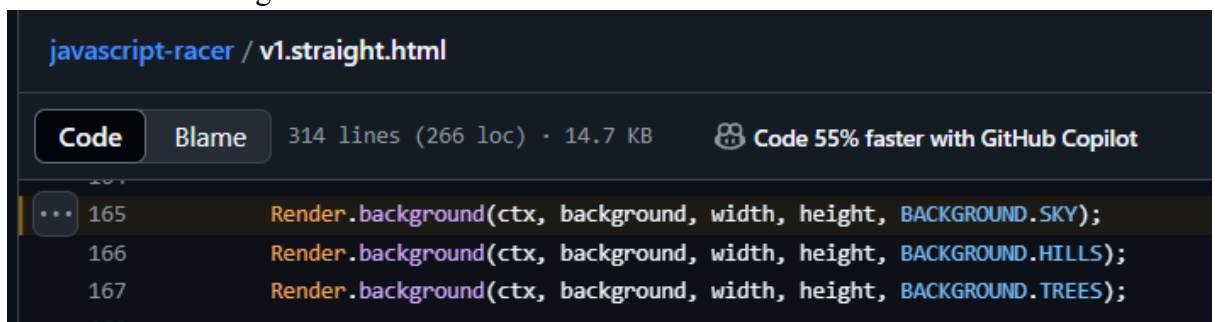


```
javascript-racer / v4.final.html

Code Blame 688 lines (591 loc) · 30.9 KB Code 55% faster with GitHub Copilot

343 Render.background(ctx, background, width, height, BACKGROUND.SKY, skyOffset, resolution * skySpeed * playerV);
344 Render.background(ctx, background, width, height, BACKGROUND.HILLS, hillOffset, resolution * hillSpeed * playerV);
345 Render.background(ctx, background, width, height, BACKGROUND.TREES, treeOffset, resolution * treeSpeed * playerV);
```

#### iii. BACKGROUND being used in v1.straight.html to render the background



```
javascript-racer / v1.straight.html

Code Blame 314 lines (266 loc) · 14.7 KB Code 55% faster with GitHub Copilot

165 Render.background(ctx, background, width, height, BACKGROUND.SKY);
166 Render.background(ctx, background, width, height, BACKGROUND.HILLS);
167 Render.background(ctx, background, width, height, BACKGROUND.TREES);
```

- iv. BACKGROUND being used in v2.curves.html to render the background

```
javascript-racer / v2.curves.html

Code Blame 387 lines (328 loc) · 17.9 KB Code 55% faster with GitHub Copilot

...
185 Render.background(ctx, background, width, height, BACKGROUND.SKY, skyOffset);
186 Render.background(ctx, background, width, height, BACKGROUND.HILLS, hillOffset);
187 Render.background(ctx, background, width, height, BACKGROUND.TREES, treeOffset);
...
```

- v. BACKGROUND being used in v3.hills.html to render the background

```
javascript-racer / v3.hills.html

Code Blame 419 lines (357 loc) · 20 KB Code 55% faster with GitHub Copilot

188 Render.background(ctx, background, width, height, BACKGROUND.SKY, skyOffset, resolution * skySpeed * playerY);
189 Render.background(ctx, background, width, height, BACKGROUND.HILLS, hillOffset, resolution * hillSpeed * playerY);
190 Render.background(ctx, background, width, height, BACKGROUND.TREES, treeOffset, resolution * treeSpeed * playerY);
191
```

### c. Usage examples in other projects or contexts

A 2D side-scrolling platformer game:

- A 2D side-scrolling platformer game such as Super Mario and Terraria. In this example, it uses the BACKGROUND component that defines the background layer's properties such as the coordinates and the speed for the sky, mountains and ground.
- Inside render(), it defines the layers with the BACKGROUND component.

Background.js

```
const BACKGROUND = {
  SKY: { x: 0, y: 0, w: 1280, h: 480, speed: 0.2 },
  MOUNTAINS: { x: 0, y: 480, w: 1280, h: 240, speed: 0.5 },
  GROUND: { x: 0, y: 720, w: 1280, h: 240, speed: 1.0 },
};
```

render()

```
function render(
  ctx,
  spriteSheet,
  cameraX,
  canvasWidth,
  canvasHeight
) {
  const layers = [BACKGROUND.SKY, BACKGROUND.MOUNTAINS, BACKGROUND.GROUND];

  layers.forEach((Layer, index) => {
    const offset = (cameraX / (index + 2)) % Layer.w;
    ctx.drawImage(
```

```

        spriteSheet,
        layer.x,
        layer.y,
        layer.w,
        layer.h,
        -offset,
        0,
        canvasWidth,
        canvasHeight / layers.length
    );
    ctx.drawImage(
        spriteSheet,
        layer.x,
        layer.y,
        layer.w,
        layer.h,
        layer.w - offset,
        0,
        canvasWidth,
        canvasHeight / layers.length
    );
});
}

```

## 2. Dom (javascript-racer/common.js)

### a. Description:

The Dom object is a reusable utility module designed to simplify and streamline interactions with the Document Object Model (DOM) in the game.

Function	Description
get(id)	Get function retrieves an element by id
set(id, html)	Set function updates the innerHTML of an element
on(ele, type, fn, capture)	On function adds an event listener to an element
un(ele, type, fn, capture)	Un function removes an event listener from an element

show(ele, type)	Show function displays an element
blur(env)	Blur function removes focus from an element
addClassName(ele, name)	addClassName adds a class to an element
removeClassName(ele, name)	removeClassName removes a class from an element
toggleClassName(ele, name)	toggleClassName adds or removes a class based on the toggle
storage	Storage function provides access to localStorage

## b. Snapshots:

### i. The Dom object in common.js

```

1 //=====
2 // minimalist DOM helpers
3 //=====
4
5 var Dom = {
6
7   get: function(id)           { return ((id instanceof HTMLElement) || (id === document)) ? id : document.getElementById(id); },
8   set: function(id, html)     { Dom.get(id).innerHTML = html; },
9   on: function(ele, type, fn, capture) { Dom.get(ele).addEventListener(type, fn, capture); },
10  un: function(ele, type, fn, capture) { Dom.get(ele).removeEventListener(type, fn, capture); },
11  show: function(ele, type)     { Dom.get(ele).style.display = (type || 'block'); },
12  blur: function(ev)           { ev.target.blur(); },
13
14  addClassName: function(ele, name) { Dom.toggleClassName(ele, name, true); },
15  removeClassName: function(ele, name) { Dom.toggleClassName(ele, name, false); },
16  toggleClassName: function(ele, name, on) {
17    ele = Dom.get(ele);
18    var classes = ele.className.split(' ');
19    var n = classes.indexOf(name);
20    on = (typeof on == 'undefined') ? (n < 0) : on;
21    if (on && (n < 0))
22      classes.push(name);
23    else if (!on && (n >= 0))
24      classes.splice(n, 1);
25    ele.className = classes.join(' ');
26  },
27
28  storage: window.localStorage || {}
29
30 }
31

```

### ii. Dom object being used in v1.straight.html's refreshTweakUI function

```

300 function refreshTweakUI() {
301   Dom.get('lanes').selectedIndex = lanes-1;
302   Dom.get('currentRoadWidth').innerHTML = Dom.get('roadWidth').value = roadWidth;
303   Dom.get('currentCameraHeight').innerHTML = Dom.get('cameraHeight').value = cameraHeight;
304   Dom.get('currentDrawDistance').innerHTML = Dom.get('drawDistance').value = drawDistance;
305   Dom.get('currentFieldOfView').innerHTML = Dom.get('fieldOfView').value = fieldOfView;
306   Dom.get('currentFogDensity').innerHTML = Dom.get('fogDensity').value = fogDensity;
307 }

```

### iii. Dom object being used in v3.hills.html's refreshTweakUI function

```

406     function refreshTweakUI() {
407         Dom.get('lanes').selectedIndex = lanes-1;
408         Dom.get('currentRoadWidth').innerHTML = Dom.get('roadWidth').value = roadWidth;
409         Dom.get('currentCameraHeight').innerHTML = Dom.get('cameraHeight').value = cameraHeight;
410         Dom.get('currentDrawDistance').innerHTML = Dom.get('drawDistance').value = drawDistance;
411         Dom.get('currentFieldOfView').innerHTML = Dom.get('fieldOfView').value = fieldOfView;
412         Dom.get('currentFogDensity').innerHTML = Dom.get('fogDensity').value = fogDensity;
413     }
414

```

iv. Dom object being used in v2.curves.html's refreshTweakUI function

```

373     function refreshTweakUI() {
374         Dom.get('lanes').selectedIndex = lanes-1;
375         Dom.get('currentRoadWidth').innerHTML = Dom.get('roadWidth').value = roadWidth;
376         Dom.get('currentCameraHeight').innerHTML = Dom.get('cameraHeight').value = cameraHeight;
377         Dom.get('currentDrawDistance').innerHTML = Dom.get('drawDistance').value = drawDistance;
378         Dom.get('currentFieldOfView').innerHTML = Dom.get('fieldOfView').value = fieldOfView;
379         Dom.get('currentFogDensity').innerHTML = Dom.get('fogDensity').value = fogDensity;
380     }

```

v. Dom object being used in v4.final.html's update function

```

216     if (position > playerZ) {
217         if (currentLapTime && (startPosition < playerZ)) {
218             lastLapTime = currentLapTime;
219             currentLapTime = 0;
220             if (lastLapTime <= Util.toFloat(Dom.storage.fast_lap_time)) {
221                 Dom.storage.fast_lap_time = lastLapTime;
222                 updateHud('fast_lap_time', formatTime(lastLapTime));
223                 Dom.addClassName('fast_lap_time', 'fastest');
224                 Dom.addClassName('last_lap_time', 'fastest');
225             }
226             else {
227                 Dom.removeClassName('fast_lap_time', 'fastest');
228                 Dom.removeClassName('last_lap_time', 'fastest');
229             }
230             updateHud('last_lap_time', formatTime(lastLapTime));
231             Dom.show('last_lap_time');
232         }
233         else {
234             currentLapTime += dt;
235         }
236     }
237

```

c. Usage examples in other projects or contexts

To-do list app:

- This to-do list app can add tasks and clear tasks. It uses Dom component to simply DOM manipulation. In this example, it uses Dom.get to get the element with id 'tasklist', Dom.set to set taskList to empty, and Dom.on to add 'click' event listener to addTaskButton and to clearTaskButton.

```
// Add a new task
function addTask(task) {
  const taskList = Dom.get('taskList'); // Get element with id 'taskList'
  const taskItem = document.createElement('li'); // Create element with id 'li'
  taskItem.textContent = task;
  taskList.appendChild(taskItem);
}

// Clear all tasks
function clearTasks() {
  Dom.set('taskList', ''); // Set taskList to empty
}

// Add 'click' event listener to addTaskButton that calls addTask()
Dom.on('addTaskButton', 'click', () => {
  const taskInput = Dom.get('taskInput');
  addTask(taskInput.value);
  taskInput.value = '';
});

// Add 'click' event listener to clearTaskButton that calls clearTasks()
Dom.on('clearTasksButton', 'click', clearTasks);
```