Practica 03

QuadTree

Mita Yagua Lesly Yaneth
4 de octubre de 2020

1. Resolución de Ejercicios

Ejercicio 01

Se edito el archivo quadtree.js y se implemento la función **query**. La captura de pantalla del **Código 1** se encuentra en la sección Capturas de pantalla como **Figura 4**.

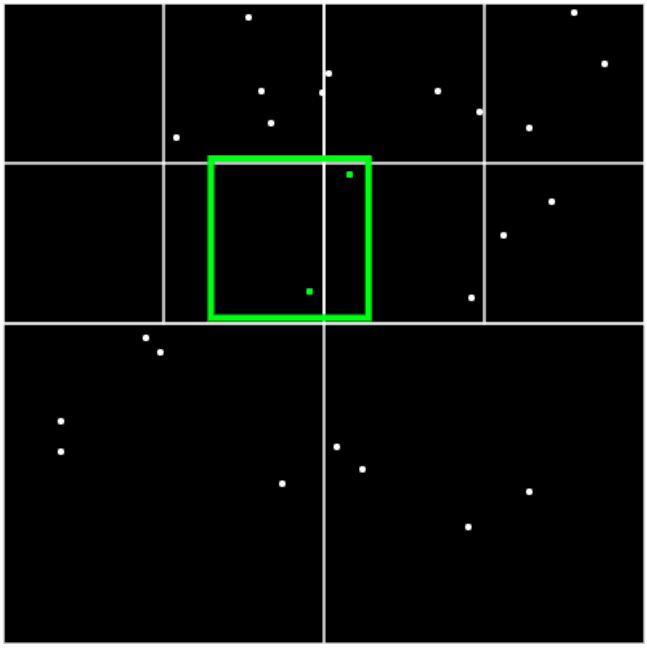
Código 1: Implementación de la función query

```
1   query(range, found) {
2       if (!found) {
3           found = [];
4       }
5
6       if (!range.intersects(this.boundary)) {
7           return found;
8       }
9
10      for (let p of this.points) {
11          if (range.contains(p)) {
12              found.push(p);
13          }
14      }
15
16      if (this.divided) {
17          this.northwest.query(range, found);
18          this.northeast.query(range, found);
19          this.southwest.query(range, found);
20          this.southeast.query(range, found);
21      }
22
23      return found;
24  }
```

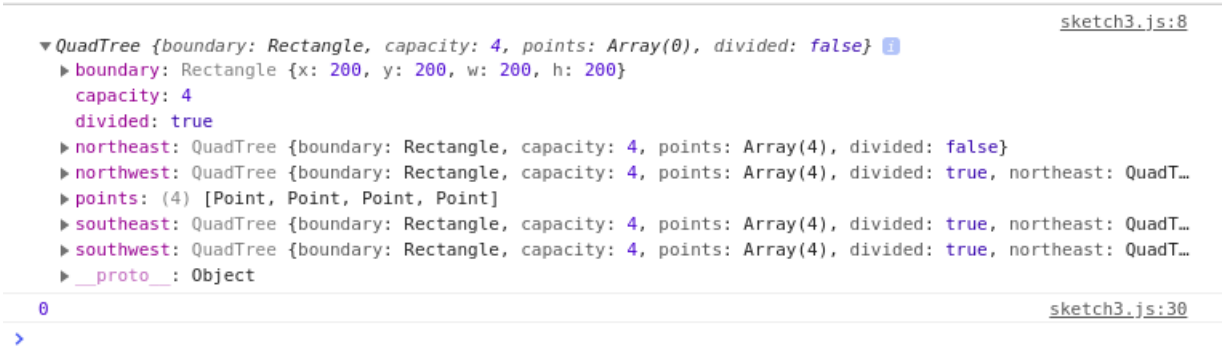
Ejercicio 02

Se creo el archivo sketch3.js, los resultados del código se muestran en la **Figura 1**. La captura de pantalla del **Código 2** se encuentra en la sección Capturas de pantalla como **Figura 7**.

- El código nos genera el mismo setup que anteriormente se vio con 25 puntos aleatorios dentro del área establecida.
- Luego se genera un rectángulo con un rango aleatorio dentro del QuadTree.
- Se pasa los parametros de el rango del rectangulo a la funcion query, por ello se muestra los puntos verdes.
- La variable count se muestra en consola como 0.



(a) Visualización del navegador web



(b) Resultados en consola

Figura 1: Resultados de sketch3.js

Código 2: sketch3.js

```
let qt;
let count = 0;

function setup() {
    createCanvas (400 ,400);
    let boundary = new Rectangle (200 ,200 ,200 ,200);
    qt = new QuadTree ( boundary , 4);
    console.log(qt);

    for (let i =0; i < 25; i ++) {
        let p = new Point ( Math.random() * 400 , Math.random() * 400);
        qt.insert(p);
    }

    background(0);
    qt.show();

    stroke (0 ,255 ,0);
    rectMode(CENTER);

    let range = new Rectangle ( random (200) ,random (200) ,random (50) ,random (50) )
    rect ( range.x, range.y, range.w*2 , range.h*2) ;
    let points = [];
    qt.query(range , points );

    for (let p of points ){
        strokeWeight(4);
        point(p.x, p.y);
    }
    console.log(count);
}
```

Ejercicio 03

Se verifico cuantas veces se consulta un punto dentro de funcion query utilizando la variable count poniendola dentro de if para verificar los puntos, incrementando su valor. También se aumento los puntos en el sketch3.js.

La captura de pantalla del **Código 3** y **Código 4**se encuentra en la sección Capturas de pantalla como **Figura ??** y **Figura ??** respectivamente.

Los resultados se pueden ver en la siguiente figura.

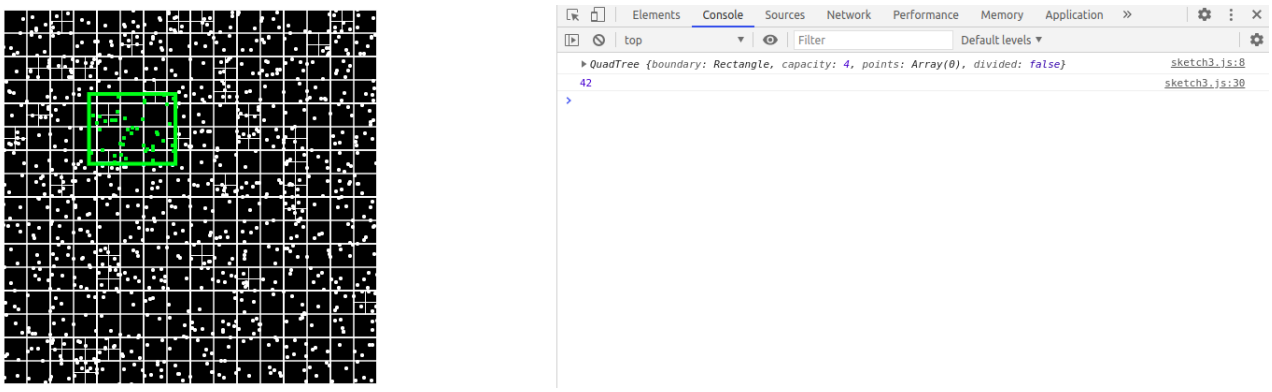


Figura 2: Visualizacion web de QuadTree con consola

Código 3: Modificación de query

```
    query(range, found) {
        if (!found) {
            found = [];
        }

        if (!range.intersects(this.boundary)) {
            return found;
        }

        for (let p of this.points) {
            if (range.contains(p)) {
                count++;
                found.push(p);
            }
        }

        if (this.divided) {
            this.northwest.query(range, found);
            this.northeast.query(range, found);
            this.southwest.query(range, found);
            this.southeast.query(range, found);
        }
        return found;

    }
```

Código 4: Modificación de skecth3.js

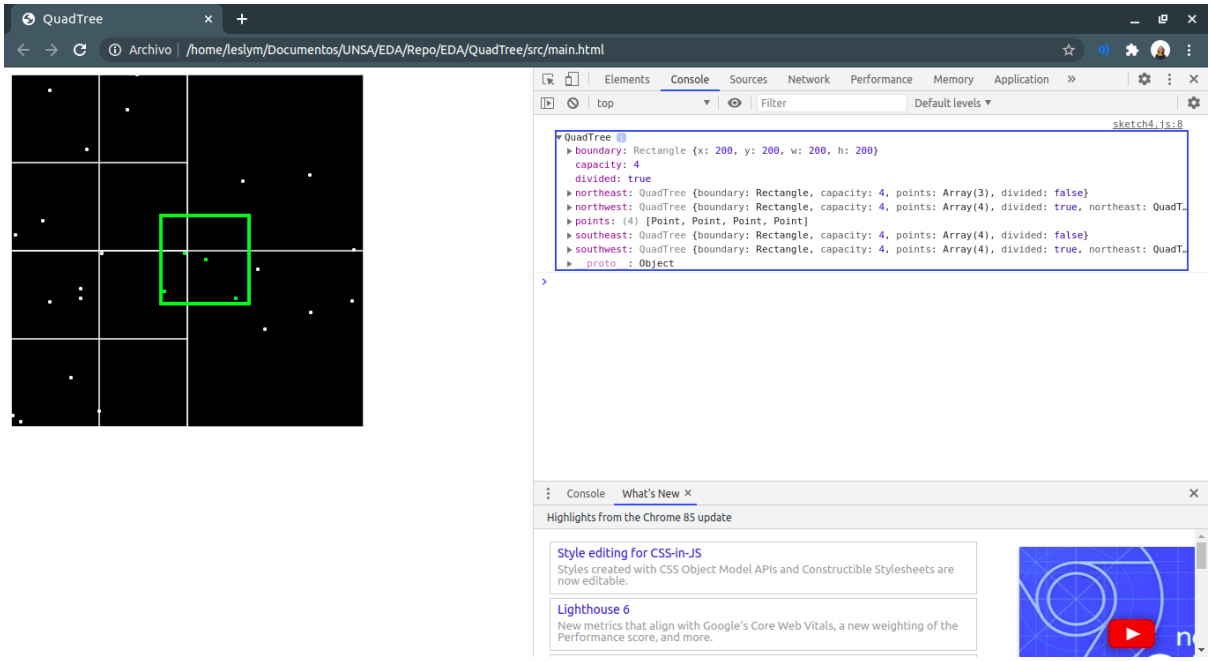
```
let qt;
let count = 0;

function setup() {
    createCanvas(400,400);
    let boundary = new Rectangle(200,200,200,200);
    qt = new QuadTree( boundary , 4);
    console.log(qt);
```

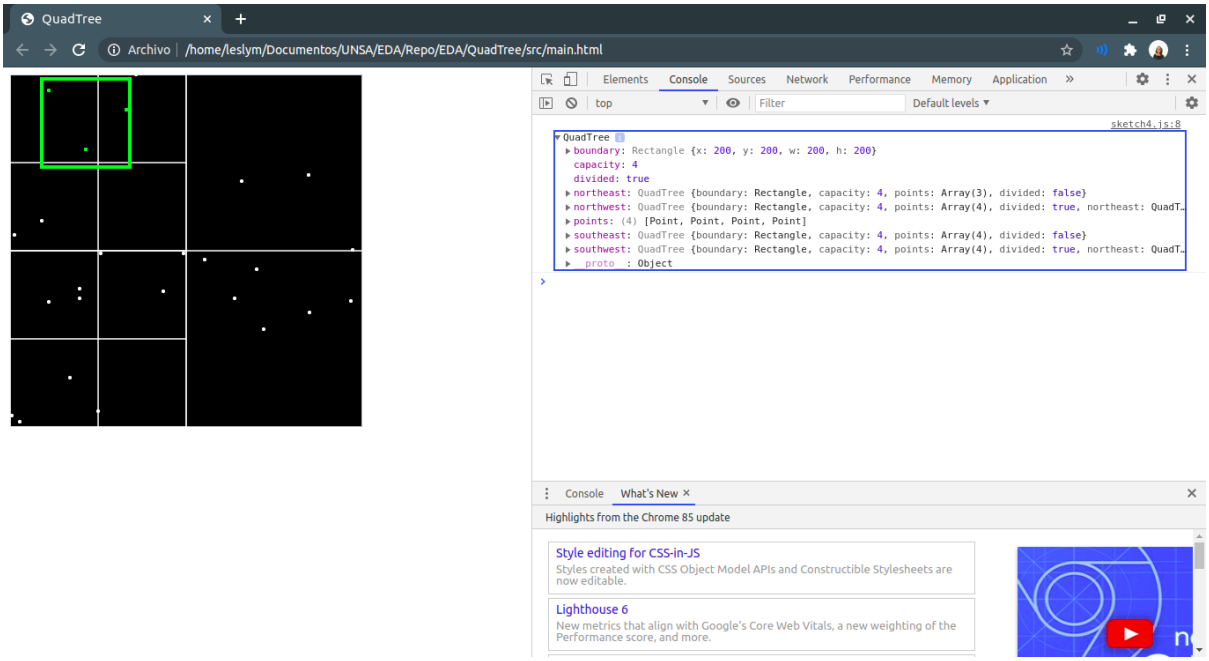
```
9
10      for (let i =0; i < 100; i ++) {
11          let p = new Point ( Math.random() * 400 , Math.random() * 400);
12          qt.insert(p);
13      }
14
15      background(0);
16      qt.show();
17
18      stroke (0 ,255 ,0);
19      rectMode(CENTER);
20
21      let range = new Rectangle ( random (200) ,random (200) ,random (50) ,random (50) )
22      rect ( range.x, range.y, range.w*2 , range.h*2) ;
23      let points = [];
24      qt.query(range , points, count );
25
26      for (let p of points ){
27          strokeWeight(4);
28          point(p.x, p.y);
29      }
30      console.log (count);
31  }
```

Ejercicio 04

Se creo el archivo sketch4.js, donde se realizaron consultas con el mouse, los resultados del código se muestran en la **Figura 3**. La captura de pantalla del **Código 5** se encuentra en la sección Capturas de pantalla como **Figura 8**.



(a) Visualización del navegador web



(b) Visualización del navegador web

Figura 3: Resultados de sketch4.js

Código 5: sketch4.js

```
let qt;
let count = 0;

function setup () {
    createCanvas (400 ,400);
    let boundary = new Rectangle (200 ,200 ,200 ,200);
```

```
7     qt = new QuadTree ( boundary , 4);
8     console .log (qt);
9     for (let i =0; i < 25; i ++) {
10        let p = new Point ( Math.random() * 400 , Math.random() * 400);
11        qt.insert (p);
12    }
13    background(0);
14    qt.show();
15 }
16
17 function draw () {
18    background (0);
19    qt.show();
20    stroke(0 ,255 ,0);
21    rectMode( CENTER );
22    let range = new Rectangle ( mouseX ,mouseY ,50 ,50)
23    rect ( range.x, range.y, range.w*2 , range.h*2);
24    let points = [];
25    qt.query(range , points );
26    for (let p of points ){
27        strokeWeight(4);
28        point(p.x, p.y);
29    }
30    // console . log ( count +" ");
31 }
```

2. Capturas de pantalla

```
query(range, found) {
    if (!found) {
        found = [];
    }

    if (!range.intersects(this.boundary)) {
        return found;
    }

    for (let p of this.points) {
        if (range.contains(p)) {
            found.push(p);
        }
    }

    if (this.divided) {
        this.northwest.query(range, found);
        this.northeast.query(range, found);
        this.southwest.query(range, found);
        this.southeast.query(range, found);
    }

    return found;
}
```

Figura 4: Captura de pantalla de la función query

```
let qt;
let count = 0;

function setup() {
    createCanvas (400 ,400);
    let boundary = new Rectangle (200 ,200 ,200 ,200);
    qt = new QuadTree ( boundary , 4);
    console.log(qt);

    for (let i =0; i < 25; i ++) {
        let p = new Point ( Math.random() * 400 , Math.random() * 400);
        qt.insert(p);
    }

    background(0);
    qt.show();

    stroke (0 ,255 ,0);
    rectMode(CENTER);

    let range = new Rectangle ( random (200) ,random (200) ,random (50) ,random (50) )
    rect ( range.x, range.y, range.w*2 , range.h*2) ;
    let points = [];
    qt.query(range , points );

    for (let p of points ){
        strokeWeight(4);
        point(p.x, p.y);
    }
    console.log(count);
}
```

Figura 5: Captura de pantalla de la sketch3.js

```
query(range, found) {
    if (!found) {
        found = [];
    }

    if (!range.intersects(this.boundary)) {
        return found;
    }

    for (let p of this.points) {
        if (range.contains(p)) {
            count++;
            found.push(p);
        }
    }

    if (this.divided) {
        this.northwest.query(range, found);
        this.northeast.query(range, found);
        this.southwest.query(range, found);
        this.southeast.query(range, found);
    }
    return found;

}
```

Figura 6: Captura de pantalla de modificación de query

```
let qt;
let count = 0;

function setup() {
    createCanvas (400 ,400);
    let boundary = new Rectangle (200 ,200 ,200 ,200);
    qt = new QuadTree ( boundary , 4);
    console.log(qt);

    for (let i =0; i < 100; i ++) {
        let p = new Point ( Math.random() * 400 , Math.random() * 400);
        qt.insert(p);
    }

    background(0);
    qt.show();

    stroke (0 ,255 ,0);
    rectMode(CENTER);

    let range = new Rectangle ( random (200) ,random (200) ,random (50) ,random (50)
    rect ( range.x, range.y, range.w*2 , range.h*2) ;
    let points = [];
    qt.query(range , points, count );

    for (let p of points ){
        strokeWeight(4);
        point(p.x, p.y);
    }
    console.log (count);
}
```

Figura 7: Captura de pantalla de modificación de sketch3.js

```
let qt;
let count = 0;

function setup () {
    createCanvas (400 ,400);
    let boundary = new Rectangle (200 ,200 ,200 ,200);
    qt = new QuadTree ( boundary , 4);
    console .log (qt);
    for (let i =0; i < 25; i ++) {
        let p = new Point ( Math.random() * 400 , Math.random() * 400);
        qt.insert (p);
    }
    background(0);
    qt.show();
}

function draw () {
    background (0);
    qt.show();
    stroke(0 ,255 ,0);
    rectMode( CENTER );
    let range = new Rectangle ( mouseX ,mouseY ,50 ,50)
    rect ( range.x, range.y, range.w*2 , range.h*2);
    let points = [];
    qt.query(range , points );
    for (let p of points ){
        strokeWeight(4);
        point(p.x, p.y);
    }
    // console . log ( count +" ");
}
```

Figura 8: Captura de pantalla de la sketch4.js

3. Repositorio

La practica 03 se encuentra en el siguiente Repositorio https://github.com/Leslym03/EDA/tree/master/QuadTree dentro de el se encuentra la carpeta **src** donde se esta el código fuente de esta practica y la carpeta **doc** donde se encuentra este documento.