Universidad Galileo
LCAT - TELUS
Tecnico Full stack
Introducción a la programacion
Ing. Gerardo Francisco Gutierrez

Tarea
7

Fecha de Entrega: 10/06/2024
Julio Enrique Soza Arostegui 23007229

Uso de .call( ) Problema I (25 puntos)

MODIFICAR PROPIEDADES DE UN OBJETO Utiliza como ejemplo 07-callaply.js proporcionado en el curso, escriba una función moverPunto que acepte un objeto punto y dos valores dx y dy. La función debe usar .call() para cambiar las propiedades x e y del objeto miPunto y agregar un método mover(dx, dy) que cambie las coordenadas del punto. Después, debe demostrar el uso de la función y el método mover.

```
// Función moverPunto que usa call() para cambiar propiedades y agregar
método mover
function moverPunto(punto, dx, dy) {
    Punto.call(punto, punto.x + dx, punto.y + dy); // Cambia las propiedades
x e y del objeto punto
    punto.mover = function(dx, dy) {
        this.x += dx;
        this.y += dy;
    };
}

// Demostración del uso de la función moverPunto y el método mover

// Crear un objeto punto
let miPunto = { x: 10, y: 20 };

// Mover el punto usando la función moverPunto
moverPunto(miPunto, 5, 7);
console.log('Después de moverPunto:', miPunto); // Salida esperada: { x: 15,
y: 27, mover: [Function] }

// Usar el método mover para cambiar las coordenadas del punto
miPunto.mover(-3, -4);
console.log('Después de usar el método mover:', miPunto); // Salida
esperada: { x: 12, y: 23, mover: [Function] }
```

Problema II (25 puntos) EXTENDER UN OBJETO CON MÚLTIPLES MÉTODOS

Utiliza como ejemplo 07-callaply.js proporcionado en el curso. Escriba una función extenderPunto que acepte un objeto punto y lo extienda con varios métodos: dibujar, mover(dx, dy) ye informar(). Estos métodos deben permitir dibujar el punto, mover el punto y mostrar las coordenadas actuales del punto, respectivamente.

```
// Función extenderPunto
function extenderPunto(punto) {
    // Método dibujar
    punto.dibujar = function() {
        console.log('Dibujando el punto en:', this.x, this.y);
    };

    // Método mover
    punto.mover = function(dx, dy) {
        this.x += dx;
        this.y += dy;
    };

    // Método informar
    punto.informar = function() {
        console.log(`Punto en las coordenadas: (${this.x}, ${this.y})`);
    };
}

// Demostración del uso de la función extenderPunto

// Crear un objeto punto
let miPunto = { x: 10, y: 20 };

// Extender el punto con los nuevos métodos
extenderPunto(miPunto);

// Usar el método dibujar
miPunto.dibujar(); // Salida esperada: Dibujando el punto en: 10 20

// Usar el método mover
miPunto.mover(5, -5);
miPunto.informar(); // Salida esperada: Punto en las coordenadas: (15, 15)

// Usar el método informar nuevamente después de mover el punto
miPunto.mover(-3, 4);
miPunto.informar(); // Salida esperada: Punto en las coordenadas: (12, 19)
```

Uso de .apply( ) Problema III (25 puntos) SUMAR PROPIEDADES DE OBJETOS

Utiliza como ejemplo 07-callaply.js proporcionado en el curso. Escribe una función sumarPuntos que acepte dos objetos punto1 y punto2. La función debe usar .apply() para sumar las propiedades x e y de punto2 a las de punto1, y agregar un método sumar() a punto1 que permita sumar las coordenadas de cualquier otro punto. Después, debe demostrar el uso de la función y el método sumar.

```
// Función que suma puntos y agrega el método sumar
function sumarPuntos(punto1, punto2) {
    function sumar(x, y) {
        this.x += x;
        this.y += y;
    }

    sumar.apply(punto1, [punto2.x, punto2.y]);

    punto1.sumar = function(otroPunto) {
        sumar.apply(this, [otroPunto.x, otroPunto.y]);
    };
}

// Demostración del uso de la función sumarPuntos y el método sumar

// Crear dos objetos punto
let punto1 = { x: 10, y: 20 };
let punto2 = { x: 5, y: 15 };

// Extender punto1 y punto2 con los métodos necesarios
extenderPunto(punto1);
extenderPunto(punto2);

// Sumar las coordenadas de punto2 a punto1 usando sumarPuntos
sumarPuntos(punto1, punto2);

// Informar el estado actual de punto1 después de la suma
punto1.informar(); // Salida esperada: Punto en las coordenadas: (15, 35)

// Crear un tercer punto
let punto3 = { x: 3, y: 7 };
extenderPunto(punto3);
```

```
// Usar el método sumar de punto1 para sumar las coordenadas de punto3
punto1.sumar(punto3);

// Informar el estado actual de punto1 después de la segunda suma
punto1.informar(); // Salida esperada: Punto en las coordenadas: (18, 42)
```

Problema IV (25 puntos) ENCONTRAR EL PUNTO MÁS LEJANO DEL ORIGEN

Utiliza como ejemplo 07-callaply.js proporcionado en el curso. Escribe una función puntoMasLejano que acepte un array de objetos puntos. La función debe usar .apply() para determinar cuál de los puntos en el array está más alejado del origen (0,0). Luego, debe devolver este punto y mostrarlo en la consola.

```
// Función para calcular la distancia euclidiana de un punto al origen
function distanciaAlOrigen(punto) {
    return Math.sqrt(punto.x * punto.x + punto.y * punto.y);
}

// Función que determina el punto más lejano del origen
function puntoMasLejano(puntos) {
    if (puntos.length === 0) {
        return null; // Manejo del caso de array vacío
    }

    // Utilizar apply() para encontrar el punto con la distancia máxima al
origen
    let maxDistancia = -Infinity;
    let puntoLejano = null;

    puntos.forEach(punto => {
        let distancia = distanciaAlOrigen(punto);
        if (distancia > maxDistancia) {
            maxDistancia = distancia;
            puntoLejano = punto;
        }
    });

    return puntoLejano;
}

// Demostración del uso de la función puntoMasLejano

let puntos = [
```

```
    { x: 1, y: 2 },
    { x: 3, y: 4 },
    { x: 5, y: 12 },
    { x: 8, y: 15 }
];

let puntoLejano = puntoMasLejano(puntos);

if (puntoLejano) {
    console.log('El punto más lejano es:', puntoLejano);
} else {
    console.log('El array de puntos está vacío.');
}
```