

Programación con Java Script I

Sesión sincrónica 5

Programación orientada a objetos



Bienvenida y actividad de bienestar

Duración: 10 minutos.

Nombre de la práctica: Centrarse en lo positivo

Descripción de la práctica: Analizarás sucesos que te hayan ocurrido recientemente, buscando orientar el análisis hacia las consecuencias positivas.

Palabras clave: Resiliencia, esperanza

Instrucciones para el participante: ¿Qué es lo primero que piensas cuando recibes una noticia inesperada? O bien, ¿qué te imaginas cuando un acontecimiento complejo se presenta ante ti?

La mayoría de las personas automáticamente se concentra en el peor de los escenarios independientemente del tipo de noticia que reciban. Martin Seligman sugiere hacer un breve ejercicio para fomentar la resiliencia y la esperanza con base en la premisa antes señalada:

1. Piensa en una noticia reciente que hayas recibido y que creas es negativa para ti.
2. Luego de analizarla, haz una tabla con tres columnas, en la primera señala cuál sería el peor de los escenarios posibles que pudieran resultar de esa noticia, en la segunda columna señala cuál sería el mejor de los escenarios posibles, y en la última cuál es el escenario que realmente tiene mayor probabilidad de ocurrir.
3. Reflexiona sobre los tres escenarios, ¿cómo enfrentarías a cada uno de ellos?

Procura repetir este ejercicio cada vez que sientas que te enfrentas a una situación complicada. Hacerlo te dará perspectiva y te ayudará a cultivar tu resiliencia.

Fuente: Seligman, M. (2011). Building Resilience. Recuperado de <https://hbr.org/2011/04/building-resilience>

Actividad guiada

Parte 1

Duración: 75 minutos.

En el siguiente ejercicio vas a practicar los temas aprendidos en esta experiencia educativa, a través de la creación de una aplicación para una librería en la que se venden libros y comics.

Para alcanzar el objetivo, realiza las siguientes actividades:

Se van a utilizar tres nuevos archivos:

1. Crea un archivo js en blanco, se sugiere que le pongas de nombre main.js.
2. Crea un archivo html con la estructura básica y ponle **index.html**.
3. Agrega un link desde el html hacia el js.

El archivo html se debería ver de la siguiente forma:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <script src="main.js"></script>
</body>
</html>
```

Actividad guiada

Parte 1

Realiza lo siguiente en el archivo js:

I. Creando los libros

1. Crea la clase **Libro**.
2. Crea el método constructor de la clase con las propiedades (**precio, titulo y autor**).
3. Crea un objeto propiedades, las **propiedades** son titulo, autor y precio, y asígnales los valores de las propiedades pasadas como parámetros. Utiliza la convención de _ para indicar que esas propiedades son privadas, como se indica en el siguiente paso.
4. Convierte en privadas las propiedades. Para ello, necesitas crear un objeto llamado **_private** de la clase **WeakMap()**. Utiliza el método set y como argumentos de este método utiliza this y el objeto **propiedades**.

```
_private.set(this,{propiedades})
```

5. Para acceder a las propiedades, utiliza los métodos de acceso **get** y **set**, excepto para la propiedad precio. Para esta propiedad, por seguridad, solo crea el getter. Ejemplo:

```
// Obtiene el título de un libro:  
get titulo() {  
    return _privado.get(this).propiedades['_titulo'];  
}  
// Establece/modifica el título de un libro:  
set titulo(newTitulo) {  
    return _privado.get(this).propiedades['_titulo'] = newTitulo;  
}
```

6. Crea un método llamado **obtenerTodosLosDatos** que imprimirá en la consola todos los valores de la siguiente manera:

```
console.log( `Título: ${this.titulo}, Autor: ${this.autor}, Precio: ${this.precio}` );
```

Receso

Duración: 10 minutos.

```
def __init__(self, **kwargs):
    self.name = kwargs.get("name")
    self.damage = kwargs.get("damage")
    self.armor = kwargs.get("armor")
    self.hit_points = kwargs.get("hp")
    self.current_hit_points = kwargs.get("hp")
    self.level = kwargs.get("level")

def attack(self, enemy: 'Unit') -> int:
    """
    Attack enemy unit. Return number of damage
    """
    damage_top_limit = self.damage + random.randint(0, 10)
    damage_bot_limit = self.damage - random.randint(0, 10)
    calculated_damage = random.randint(damage_bot_limit, damage_top_limit)
    if calculated_damage < 0:
        return 0
    enemy.current_hit_points -= calculated_damage
```

Actividad guiada

Parte 2

Duración: 75 minutos.

II. Creando los comics

7. Crea una clase **Comic** que herede desde la clase **Libro** con **extends**.
8. Crea un constructor al que se le van a pasar las propiedades del padre y agrega la propiedad **ilustradores**.
9. Utiliza la función **super** para poder utilizar las propiedades titulo, autor y precio de la clase padre (**Libro**).
10. Define una nueva propiedad de tipo array, que se llamará **ilustradores**, para contener el nombre de los ilustradores de cada Comic y asígnale el valor pasado como argumento en el constructor.
11. Crea un nuevo método que se llame **agregarIlustrador**, el cual aceptará un arreglo como parámetro. Este método agregará ilustradores al arreglo, utiliza el método push para eso.
12. Sobrescribe el método **obtenerTodosLosDatos**. Para lograrlo, accede al método **obtenerTodosLosDatos** de la clase padre. Además, imprime en consola los ilustradores con:

```
console.log( `Ilustradores: ${this.ilustradores}` );
```

III. Creando un carrito de compras

13. Crea una nueva clase **CarritoCompras**, solo necesitarás un constructor y declarar un arreglo llamado **productos**, el cual se estará llenando posteriormente.
14. Crea un método llamado **agregarProducto**, el cual recibirá dos argumentos: la **cantidad** y el **precio** del producto. Crea un array dinámico para que se cree utilizando los datos de entrada. Puedes utilizar la siguiente instrucción:

```
this.productos.push(...Array(cantidad).fill(precio));
```

15. Crea un nuevo método llamado **mostrarProductos** para mostrar los productos ingresados al array, el cual lo único que hará es imprimir en consola el contenido del arreglo productos.

Actividad guiada

Parte 2

16. Crea un nuevo método llamado **calcTotal** para calcular el total del contenido en el carrito de compras. Para ello, es necesario hacer un mapeo de los productos y con una función flecha le indicaremos que calcule el precio utilizando el método **reduce**, el cual recibe como parámetros un **acumulador** y el **precio**. El acumulador inicializará en 0 y en cada iteración le sumará el **precio** con una función flecha. Puedes utilizar un código parecido al siguiente como cuerpo del método:

```
return this.productos
    .map( precio => precio )
    .reduce( (ac, precio) => ac + precio, 0 );
```

17. Crea una función para poder imprimir el total del carrito, nómbrala **imprimirTicket**, lo único que realizará será imprimir en la consola la cadena:
`Total a pagar \${ this.calcTotal() }`

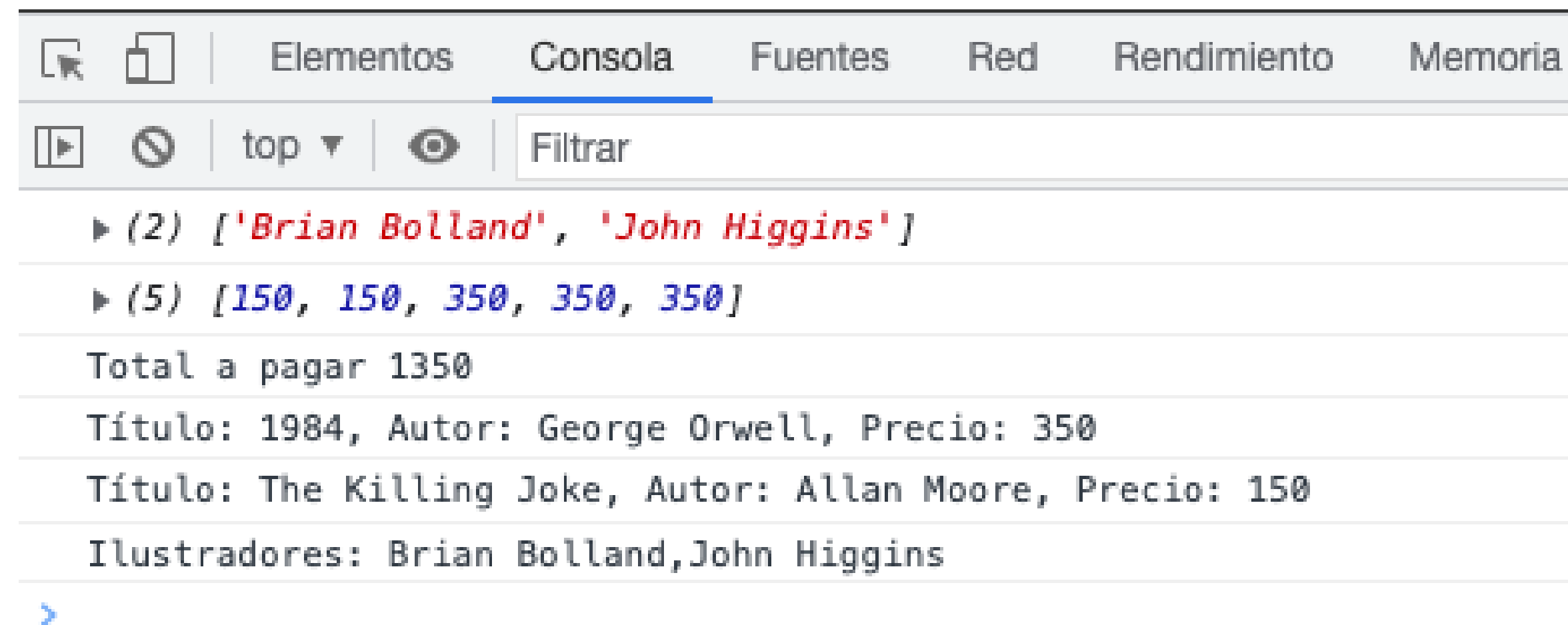
Usando las clases creadas:

18. Crea un nuevo objeto llamado **carrito**, el cual será una instancia de la clase **CarritoCompras**.
19. Utiliza el método **agregarProducto** y pásale como parámetros 2 y el precio del comic1.
20. Utiliza el método **agregarProducto** y pásale como parámetros 3 y el precio del libro1.
21. Utiliza el método **mostrarProductos** para visualizar el contenido del carrito.
22. Utiliza el método **imprimirTicket** para visualizar el total del carrito de compras (en dinero).
23. Utiliza el método **obtenerTodosLosDatos** de **libro1** para mostrar los datos en la consola.
24. Utiliza el método **obtenerTodosLosDatos** de **comic1** para mostrar los datos en la consola.

Actividad guiada

Parte 2

Al final, tu programa tendrá que arrojar una salida parecida a la siguiente:



```
▶ (2) ['Brian Bolland', 'John Higgins']
▶ (5) [150, 150, 350, 350, 350]
Total a pagar 1350
Título: 1984, Autor: George Orwell, Precio: 350
Título: The Killing Joke, Autor: Allan Moore, Precio: 150
Ilustradores: Brian Bolland,John Higgins
>
```

Consideraciones

Instructor, refuerza los conocimientos de aquellos temas que consideres necesarios para que el aprendiz pueda realizar este ejercicio

La obra presentada es propiedad de ENSEÑANZA E INVESTIGACIÓN SUPERIOR A.C. (UNIVERSIDAD TECMILENIO), protegida por la Ley Federal de Derecho de Autor; la alteración o deformación de una obra, así como su reproducción, exhibición o ejecución pública sin el consentimiento de su autor y titular de los derechos correspondientes es constitutivo de un delito tipificado en la Ley Federal de Derechos de Autor, así como en las Leyes Internacionales de Derecho de Autor.

El uso de imágenes, fragmentos de videos, fragmentos de eventos culturales, programas y demás material que sea objeto de protección de los derechos de autor, es exclusivamente para fines educativos e informativos, y cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por UNIVERSIDAD TECMILENIO.

Queda prohibido copiar, reproducir, distribuir, publicar, transmitir, difundir, o en cualquier modo explotar cualquier parte de esta obra sin la autorización previa por escrito de UNIVERSIDAD TECMILENIO. Sin embargo, usted podrá bajar material a su computadora personal para uso exclusivamente personal o educacional y no comercial limitado a una copia por página. No se podrá remover o alterar de la copia ninguna leyenda de Derechos de Autor o la que manifieste la autoría del material.