

CLOSURES, ALTO ORDEN Y CIUDADANÍA DE PRIMERA CLASE

Conceptos de Programación Funcional y
JavaScript con matices de TypeScript

A hand in a white shirt cuff points to a specific location on a complex, multi-colored subway map. The map is spread out, showing various lines in blue, yellow, red, and green. The background is dark and out of focus.

Al terminar la charla se compartirá el acceso:

- ▶ A la grabación
- ▶ A las diapositivas
- ▶ A los recursos de la sesión

¡¡AVISO A NAVEGANTES!!

PRESENTACIÓN

Soy Pepe, actualmente desarrollador de front-end, trabajando con React y TypeScript. Y casi siempre trasteando con distintas tecnologías.





ANTES DE EMPEZAR

Esta charla es una continuación directa de otras:

- ▶ <https://github.com/jofaval/talks-about/tree/master/concepts-of-js/pureness-side-effects-and-idempotence>
- ▶ <https://github.com/jofaval/talks-about/tree/master/concepts-of-js/memoization-serialization-and-value-comparison>

Pero repasemos algunos conceptos...

CONTINUACIÓN DE OTRA CHARLA

- ▶ **Pureza**
 - ▶ Funciones determinísticas que mitigan la cantidad de efectos secundarios
- ▶ **Efectos Secundarios**
 - ▶ Acciones más allá del cuerpo de la función
- ▶ **Idempotencia**
 - ▶ Término matemático para la pureza
- ▶ **Memoización**
 - ▶ Cache para funciones idempotentes
- ▶ **Serialización (en JavaScript)**
 - ▶ Conversión de valores primitivos y no primitivos a string
- ▶ **Comparación de valores**
 - ▶ Problemática, pero parcheable

CONTEXTUALIZANDO...

- ▶ Closures
 - ▶ Encapsulación privada
- ▶ Ciudadanía de primera clase
- ▶ Alto orden
 - ▶ Funciones de alto orden
 - ▶ Componentes de alto orden (React)

¿QUÉ VEREMOS?



CLOSURE



Un closure en JavaScript es una función creada dentro de otra

Pero también es:

- ▶ Un contexto
- ▶ Encapsulación privada

¿QUÉ ES UN CLOSURE?

A veces, a lo largo del desarrollo, hay partes que queríamos extraer, pero que no podemos, porque necesitan de un **contexto**.

¿Y si te dijese que eso se puede hacer?

Solo necesitamos algo que nos ayude a **proveer** ese contexto y listo.

Veamos más beneficios de los closures.

CONTEXTO

La encapsulación privada es un concepto común en lenguajes orientados a objetos (Java, C#, etc.)

Se dice que JavaScript no tiene propiedades privadas

- ▶ Y esto es cierto... a medias

ENCAPSULACIÓN PRIVADA

Ocultar la implementación, controlar el acceso a propiedades, en definitiva, proteger la implementación.

Esto es algo que podemos conseguir con los closures, no tienen el mismo comportamiento que las propiedades privadas, pero casi

¿QUÉ ENTENDEMOS POR
ENCAPSULACIÓN PRIVADA?

```
type SetState<T> = [T | undefined, (value: T) => void];

const useState = function <TValue>(defaultValue?: TValue): SetState<TValue> {
  let value = defaultValue;
  const setValue = (newValue: TValue) => (value = newValue);
  return [value, setValue];
};
```

```
const [count, setCount] = useState(1);
// count es "inaccesible" desde fuera
console.log(count); // 1
setCount(3);
console.log(count); // 3
```

EJEMPLO DE CLOSURE

En runtime, es decir, podemos “enriquecer” el ecosistema, si vamos con control

```
const generadorDeMultiplicaciones =  
  (multiplicador: number) => {  
    return (a: number) => a * multiplicador;  
  };
```

```
const multiplyPorCero = generadorDeMultiplicaciones(0);  
multiplyPorCero(1); // 0  
multiplyPorCero(2); // 0  
multiplyPorCero(4); // 0
```

```
const duplica = generadorDeMultiplicaciones(2);  
duplica(2); // 4
```

GENERACIÓN DE FUNCIONES Y EVENTOS



CIUDADANÍA DE PRIMERA CLASE



Una entidad que permite la mayoría de operaciones disponibles a otras entidades:

- ▶ Pasarse como argumento de una función (input)
- ▶ Devolverse de una función (output)
- ▶ Asignarse una variable

¿QUÉ ES LA CIUDADANÍA DE PRIMERA CLASE?

Un término un poco absurdo, que tiene ciertas restricciones en comparación:

- ▶ No puede ser asignado a una variable
- ▶ O pasado como parámetro
- ▶ O devuelto de una función

Una clase sería el ejemplo idóneo de esta casuística

CIUDADANÍA DE SEGUNDA CLASE

ANTES DE NADA



A 3D rendering of a puzzle with many grey pieces and one prominent red piece in the center. The text is overlaid on the red piece. Several white diagonal lines are on the right side of the image.

¿SE DIFERENCIA EN ALGO
DE UN CLOSURE?

La ciudadanía de primera clase y un closure...

- Son diseño e implementación en JavaScript

Esto significa que el concepto se implementa con closures, en JavaScript, diferentes lenguajes proveen de diferentes herramientas.

NO DEBERÍA...

A close-up, shallow depth-of-field shot of a camera lens. The lens is dark and metallic, with its front element reflecting a blurred cityscape with tall buildings. The background is a soft, out-of-focus bokeh of purple and blue light spots. The text is overlaid on the left side of the lens.

¿CÓMO IMPLEMENTARÍAS
EL CONCEPTO?

Pero entonces, si la ciudadanía es de primera clase, ¿cómo llamamos a esos ciudadanos?

Es decir, si son funciones ¿cómo distinguimos a una función de primera clase con otra de segunda clase?

ADENTRÁNDONOS EN EL CONCEPTO





ALTO ORDEN



No, no tiene que ver con la orden 66

- ▶ Es un elemento de primer nivel en la programación
- ▶ Un estilo de programación que puede usar primitivos y no primitivos como valores
- ▶ Para este contexto, es similar a la ciudadanía de primera clase

La entidad más conocida de este grupo son las Funciones de Alto Orden, que veremos a continuación

¿QUÉ ES ALGO DE ALTO ORDEN?

Existen distintos elementos de alto orden, veremos los siguientes:

- ▶ Funciones de alto orden
- ▶ Componentes de alto orden (orientado a React)

ELEMENTOS DE ALTO ORDEN

High-Order Functions o HOFs

Su definición oficial dice que cumplen una o más de estas condiciones:

- ▶ Reciben como input una o más funciones
- ▶ Devuelven como output una o más funciones

Es decir, closures, un closure puede recibir una función o devolver otra.

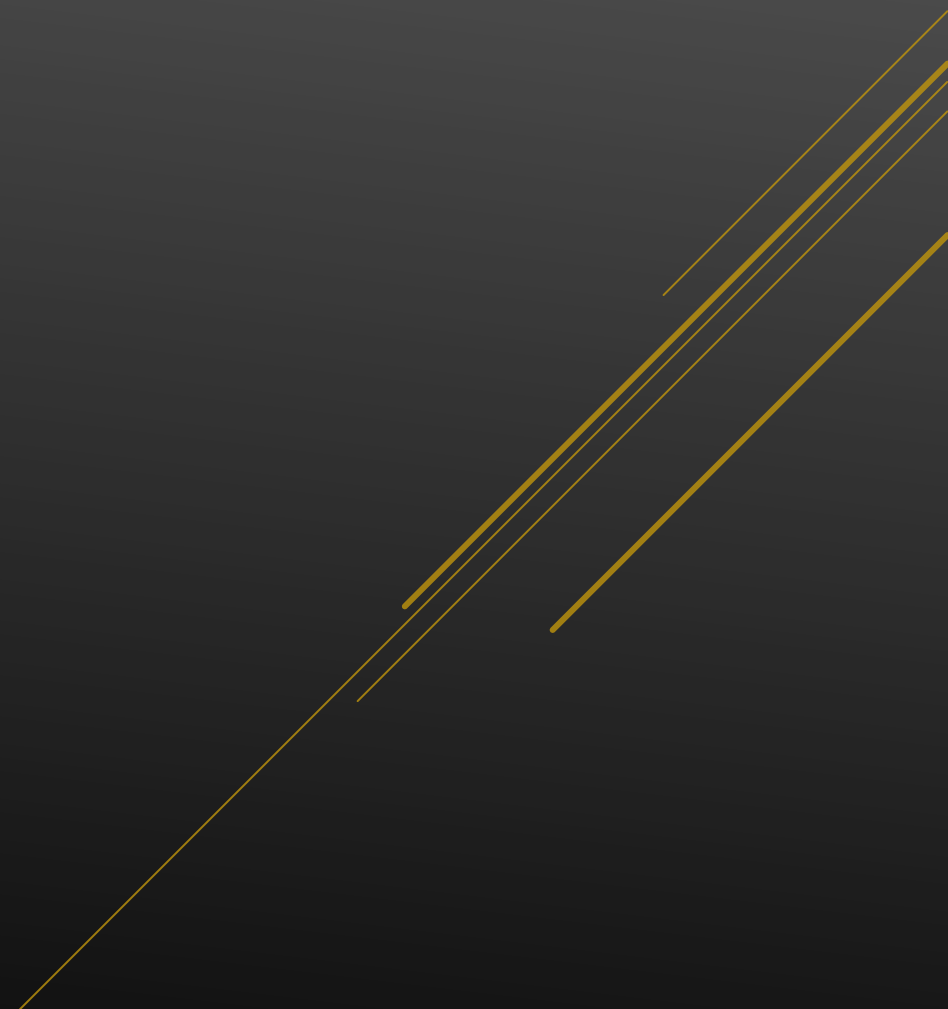
FUNCIÓN DE ALTO ORDEN

High-Order Components o **HOCs**, proveen de contexto al children

```
<Context.Provider>  
  <ChildrenComponentWithoutContext />  
</Context.Provider>
```

COMPONENTE DE ALTO ORDEN

RECAPITULEMOS...



Hemos visto ciudadanías, alto orden y closures.

La ciudadanía de primera clase permite algunas operaciones comunes (actuar como un valor) que la ciudadanía “de segunda” no permite (una clase no puede ser devuelta de una función, una instancia sí).

El alto orden, para pasar entidades y devolverlas, y como un closure en JavaScript es una función de alto orden

CONCEPTOS REVISITADOS

Son conceptos complejos y algo teóricos, ahora podría ser un muy buen momento para comentar las dudas.

DUDAS

Several thin, parallel white lines of varying lengths and orientations are positioned in the bottom right corner of the slide, creating a modern, abstract graphic element.

Ahora podemos entender que trabajar con contextos puede ser más sencillo, y que las funciones se pueden adaptar según los valores del runtime.

El contexto de un closure puede dar ideas sobre cómo implementar algunos conceptos, o incluso crear algunos nuevos.

CAMINOS ABIERTOS

Several thin, white, parallel diagonal lines are positioned in the bottom right corner of the slide, extending from the right edge towards the center.

Al menos... por ahora.

FIN DE LA SERIE



- ▶ Localidad
- ▶ Inmutabilidad (programación funcional y estructuras de datos)
- ▶ Libros y charlas
- ▶ Reevalúa código escrito recientemente

¿POR DÓNDE PODRÍA INVESTIGAR
MÁS?

Puedes encontrar el artículo más detallado y con ejemplos en:

► <https://medium.com/@jofaval/d26fc09e149>

Hay algunas implementaciones a mano a modo de ejemplo.

ARTÍCULO

Artículos

<https://medium.com/@jofaval/a60130f073ef>

<https://medium.com/@jofaval/d26fc09e149>

Libros

<https://amzn.eu/d/0CVqCi6>

Cursos

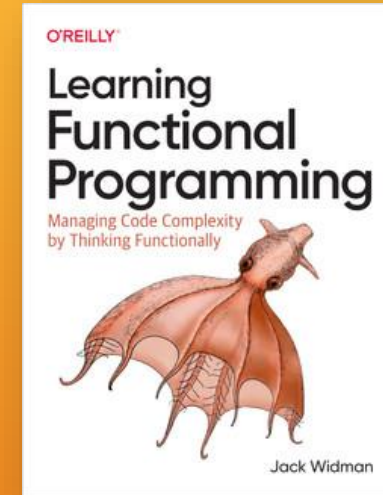


<https://justjavascript.com/>



Programando...

Charlas



<https://a.co/d/dBo8L3m>



Salida a mano izquierda están las camisetas.

GRACIAS POR TU ATENCIÓN

