"¡Llamaré después!"

Una callback es una función que se pasa como argumento a otra función.
Esta técnica permite que una función llame a otra función.

Secuencia de funciones

Las funciones de JavaScript se ejecutan en la secuencia en que se llaman. No en la secuencia en que están definidos.

Este ejemplo terminará mostrando "Hola" y "Adiós":

Ejemplo

```
function mySecond() {
  console.log("Adiós");
}

function myFirst() {
  console.log("Hola");
}

myFirst();
mySecond();
```

Este otro ejemplo terminará mostrando "Adiós" y "Hola":

Ejemplo

```
function myFirst() {
  console.log("Hola");
}

function mySecond() {
  console.log("Adiós");
}

mySecond();
myFirst();
```

Control de secuencia

A veces nos gustaría tener un mejor control sobre cuándo ejecutar una función.
Supongamos que deseamos hacer un cálculo y luego mostrar el resultado.

Puede llamar a una función de calculadora (myCalculator), guardar el resultado y luego llamar a otra función (myDisplayer) para mostrar el resultado:

Ejemplo

```
const myDisplayer = string => console.log(string);

function myCalculator(num1, num2) {
  let sum = num1 + num2;
  return sum;
}

let result = myCalculator(5, 5);
myDisplayer(result);
```

O bien, puede llamar a una función de calculadora (myCalculator) y dejar que la función de calculadora llame a la función de visualización (myDisplayer):

Ejemplo

```
function myDisplayer(some) {
  console.log("el resultado de la multiplicación es: " +some);
}

function myCalculator(num1, num2) {
  let sum = num1 + num2;
  myDisplayer(sum);
}

myCalculator(5, 5);
```

El problema con el primer ejemplo anterior es que debe llamar a dos funciones para mostrar el resultado.

El problema con el segundo ejemplo es que no puede evitar que la función de calculadora muestre el resultado. Y además myCalculator DEPENDE de myDisplayer

Ahora es el momento de devolver la llamada.

Callback de JavaScript

Un callback es una función que se pasa como argumento a otra función.

Con una callback, puede llamar a la función de calculadora (myCalculator) con una callback y dejar que la función de calculadora ejecute la callback una vez finalizado el cálculo:

Ejemplo

```
function myDisplayer(some) {
  console.log(some);
}

function myCalculator(num1, num2, myCallback) {
  let sum = num1 + num2;
  myCallback(sum);
}

myCalculator(5, 5, myDisplayer);
```

En el ejemplo anterior, myDisplayer es el nombre de una función.

Se pasa a myCalculator()como argumento.

Cuando pase una función como argumento, recuerde no usar paréntesis.

BIEN: `myCalculator (5, 5, myDisplayer);`

MAL: `myCalculator (5, 5, myDisplayer ());`

¿Cuándo usar un callback?

Los ejemplos anteriores no son muy interesantes.

Están simplificados para enseñarle la sintaxis de callback.

Donde las callback realmente brillan son en las funciones asincrónicas, donde una función tiene que esperar a que otra función (como esperar a que se cargue un archivo).

Las funciones asincrónicas se tratan en el siguiente capítulo.

ver: https://javascript.info/callbacks