



BOOTCAMP

# Generación T

 streambe

# Introducción a funciones

# ¿Qué son las funciones en JS?



## Bloques de código

Agrupar instrucciones que se pueden invocar repetidamente.



## Reutilización

Permiten ejecutar el mismo código múltiples veces desde diferentes partes.



## Modularidad

Dividen problemas complejos en piezas más pequeñas y manejables.

# Importancia de las funciones



# Formas de crear funciones



## Declaración

```
function nombre(p1, p2...) { }
```

Disponible en todo el ámbito debido al hoisting.



## Expresión

```
let nombre = function(p1, p2...) { }
```

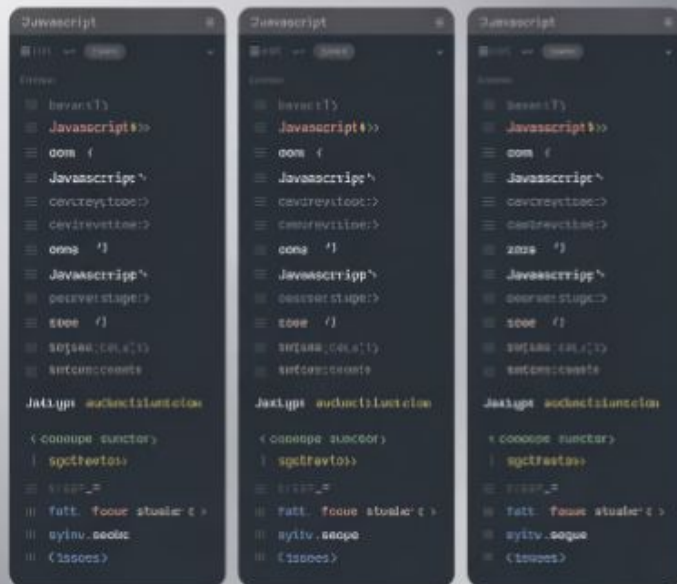
Solo disponible después de su definición.



## Constructor

```
new Function(p1, p2..., 'code')
```

No recomendado por razones de seguridad y rendimiento.

[Home](#)[Docs](#)[Examples](#)[Sign in](#)

## Compare function syntaxes

[Try it yourself](#)

# Ejemplos y sintaxis práctica

```
// Función declarativa
```

```
function sumar(a, b) {  
  return a + b;  
}
```

```
// Función expresión
```

```
const multiplicar = function(a, b) {  
  return a * b;  
};
```

```
// Función flecha (ES6)
```

```
const dividir = (a, b) => a / b;
```

**Escribir**  
Define la función con su lógica interna.



## Invocar

Llama a la función: `sumar(5, 3)`

## Resultado

Obtén el valor retornado: 8

# Diferencia Entre Declarar Y Ejecutar Una Función

## Declaración

Es la definición del funcionamiento interno.

- Crea el "plano" de la función
- Define parámetros y lógica
- No ejecuta el código interno

Ejemplo:

```
function saludar(nombre) {  
  return "Hola " + nombre;  
}
```

## Ejecución

Es la invocación que activa el código.

- Llama a la función con paréntesis
- Envía argumentos si es necesario
- Procesa el código y retorna resultados

Ejemplo:

```
// Invocación de la función  
saludar("María"); // Retorna: "Hola María"
```



# Funciones Con Parámetros

Los bloques de código se vuelven flexibles al recibir datos externos.

AB

## Entrada

Los parámetros funcionan como variables locales que reciben valores externos.



## Proceso

La función procesa estos datos mediante su lógica interna.



## Salida

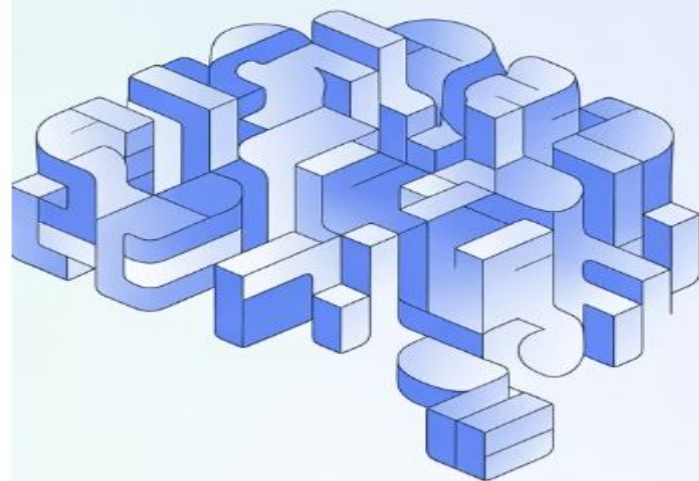
El valor de retorno entrega el resultado final al código que la invocó.

```
function calcularArea(base, altura) {  
  return base * altura;  
}
```

```
// Resultado: 50  
let area = calcularArea(10, 5);
```

Functionflow  
Optimize your codebase  
with intelligent parameter  
management,

Try it free



# Funciones Con Múltiples Parámetros

Las funciones alcanzan su máximo potencial al manejar múltiples entradas.



## Orden Importante

Los parámetros se asignan por posición en la invocación.



## Parámetros Opcionales

Usa valores predeterminados con el operador = para parámetros no enviados.



## Operador Rest

Captura argumentos ilimitados con ...args para funciones flexibles.

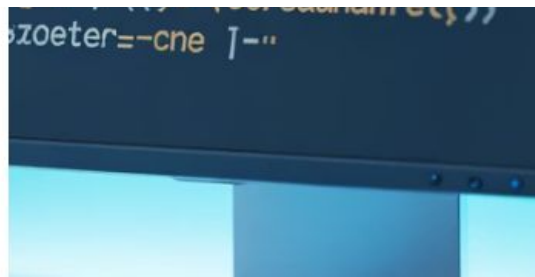


## Destructuración

Extrae propiedades de objetos directamente en los parámetros.

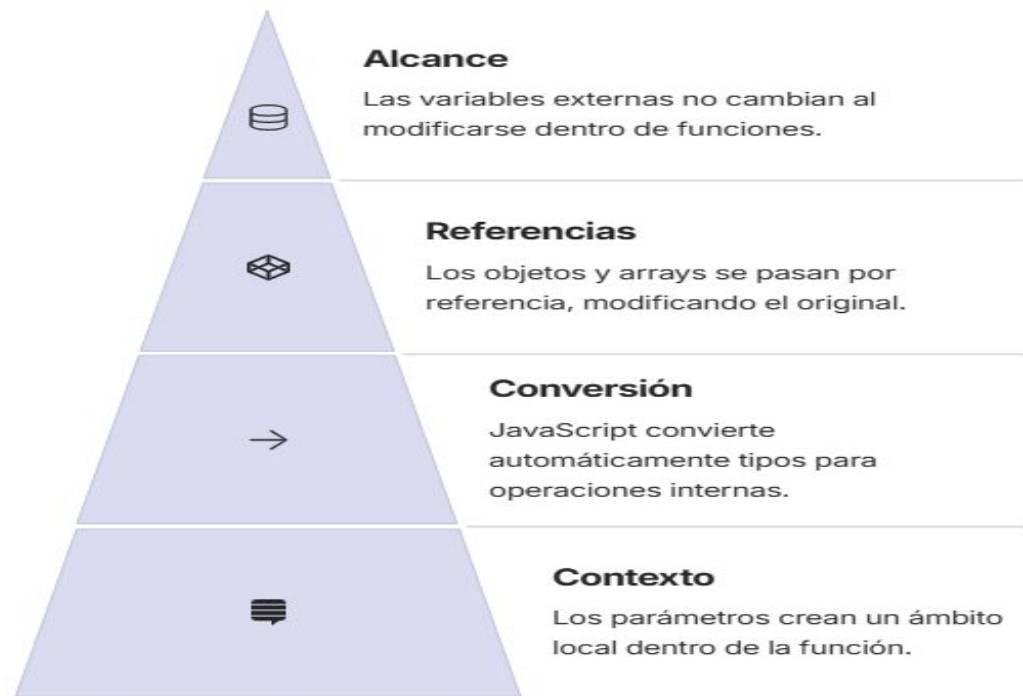
```
function calcularPrecio(base, impuesto = 0.21, descuento = 0) {  
    return base + (base * impuesto) - descuento;  
}
```

```
// Resultado: 121  
calcularPrecio(100);
```



# Funciones Con Variables Por Parámetro

Las funciones pueden recibir y transformar cualquier tipo de variable.



Una función puede recibir y manipular strings, números, booleanos, objetos o incluso otras funciones.

```
eds- c3, TApales and 1decustdTL
ascript (ede curttion
dved-]fvucnorT)l++
vppe-lcd-cod () l-dieeloconion(
> oveapclue,leleceenlce,+f(
z vevpes vaset T loecvcento
vesanped-Tyureeueeu > oamill
vosde (ll*,29,(l(= =)
veisarvdetion~*,c3,l(*+,curll
veurecss venlereee(
vele vaset -letoideoch,+
yecusonioccel,-love},-edvcnl ;)
oecrio.gsrom"i ,];,01,
```

ly manage  
e variables  
obust  
: function

Run

# Parámetros Por Default

Las funciones modernas permiten establecer valores predeterminados que simplifican el código y manejan casos donde faltan argumentos.

## Sintaxis Simple

Asigna valores directamente en la definición de parámetros usando el operador `=`.

## Código Más Limpio

Elimina verificaciones condicionales dentro del cuerpo de la función.

## Valores Flexibles

Los valores default pueden ser cualquier expresión válida, incluso llamadas a otras funciones.


```
function crearUsuario(nombre, edad = 18, rol = "invitado") {  
  return { nombre, edad, rol };  
}
```

```
// Resultado: {nombre: "Ana", edad: 18, rol: "invitado"}  
const usuario = crearUsuario("Ana");
```

# La Keyword Return

El comando que finaliza la ejecución de una función y determina qué valor se enviará de vuelta al código que la invocó.





```
function multiplicar(a, b) {  
  return a * b; // Devuelve el producto  
}  
  
const resultado = multiplicar(5, 3); // resultado = 15
```

Si no se especifica un return, la función devolverá undefined por defecto.





# Hoisting En JavaScript

Un comportamiento único que afecta cómo se procesan las declaraciones de variables y funciones antes de ejecutar el código.



## Elevación

Las declaraciones se "elevan" al inicio del ámbito, pero no sus inicializaciones.

$f(x)$

## Funciones Completas

Las declaraciones de funciones se elevan completamente, permitiendo llamarlas antes de declararlas.



## Variables Parciales

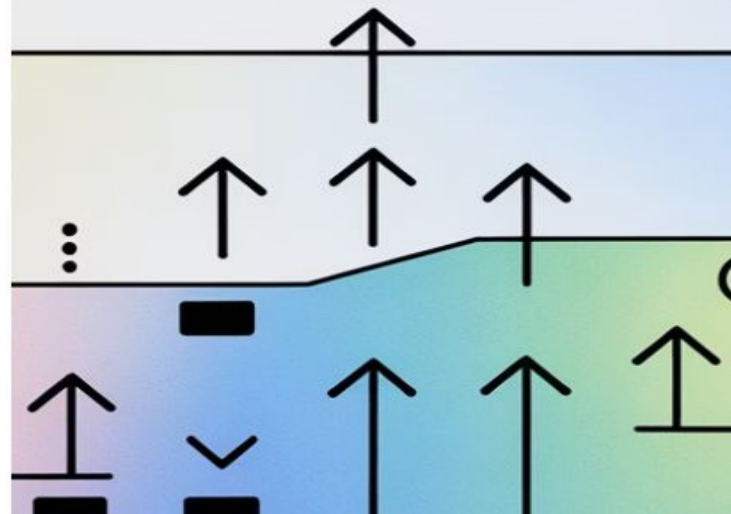
Con var, solo la declaración se eleva, no su valor. Causa undefined inicialmente.




## Let y Const

Se elevan pero quedan en "zona muerta temporal" hasta su declaración real.

# Javascript Hoisting







```
// Funciona gracias al hoisting
saludar();
```

```
function saludar() {
  console.log("¡Hola mundo!");
}
```

```
// Variables con var
console.log(x); // undefined
var x = 5;
```

# Hands on!





# Muchas gracias.

# Nuestras Redes

[www.generaciont.org](http://www.generaciont.org)

[www.streambe.com](http://www.streambe.com)

[www.instagram.com/generaciont\\_ar](https://www.instagram.com/generaciont_ar)

[www.tiktok.com/@generaciont](https://www.tiktok.com/@generaciont)

[generaciont@generaciont.org](mailto:generaciont@generaciont.org)

Cel: [11 61331747](tel:1161331747)



[11 6133-1747](tel:1161331747)



GENERACIÓN T

[generaciont@generaciont.org](mailto:generaciont@generaciont.org)



GENERACIÓN T