

Funciones

DEV.F
DESARROLLAMOS(PERSONAS);

dev

FUNCIONES

Recibe **elementos de entrada** por ejemplo x , y

Entrega un **valor de salida**

Podemos pasar **fórmulas matemáticas** para obtener cierto valor

$$f(x, y) = x^2 + y^2$$



RECETA PARA TRABAJAR CON FUNCIONES

1.- DECLARAR MI FUNCIÓN.

2.- PODEMOS ESTABLECER LOS PARÁMETROS O BIEN PEDIRLE AL USUARIO QUE INGRESE ESOS PARÁMETROS.

3.- INGRESAR LAS INSTRUCCIONES A MI FUNCIÓN, ES DECIR, EL CÓDIGO QUE VA A EJECUTAR.

4.- LLAMAR LA FUNCIÓN QUE DECLARAMOS PREVIAMENTE.



DECLARAR NUESTRA FUNCIÓN

Nombre de mi función

```
function MyFunction ( ) {  
  
}
```

PARÁMETROS

```
function MyFunction( ) {
```

```
}
```

```
function MyFunction( params ) {
```

```
}
```

```
function MyFunction ( num1, nombre, edad ){
```

```
}
```

INSTRUCCIONES

```
function Suma ( num1, num2 ) {  
    Instrucciones | bloque de código  
}
```

```
function Suma ( num1, num2 ) {  
    var total = num1 + num2;  
    return "LA SUMA ES: " + total;  
}
```

LLAMAR MI FUNCIÓN

```
function Suma ( num1, num2 ) {  
    var total = num1 + num2;  
    return "LA SUMA ES: " + total;  
}  
Suma( 2 , 2)
```

" LA SUMA ES: 4 "

FUNCIONES ANIDADAS

EJEMPLO DE FUNCIÓN ANIDADA

```
function ObtenerMacador() {  
    var partido1 = 3;  
    var partido2 = 3;  
  
    function Agregar( ) {  
        var nombre = "juanito";  
        return nombre + " anoto "  
        + (partido1 + partido2) + " goles";  
    }  
    return Agregar()  
}  
  
ObtenerMacador()
```



Console.log vs return

`console.log("some text here")`

Imprime lo que sigue dentro de los paréntesis

`return "some text here" + variable`

devuelve el valor de una operación. Esto sirve para luego asignar ese valor a una variable.

El procesador realiza las operaciones de la función, pero sólo va a guardar su resultado en memoria si le indicamos **return**.

Y una vez guardado en memoria podemos rescatar ese valor para asignarlo a una variable y poder seguir haciendo operaciones con ese valor.

FUNCIÓN PURA

```
function sumaUnoAlNumero(numero) {  
  return numero + 1;  
}
```

- No tiene efectos de estado que la afecten, lo que quiere decir que siempre que le pasemos un número A este dará como resultado un número B, como ejemplo si le damos como número un 3 este nos arrojará como número un 4.

FUNCIÓN IMPURA

```
function sumaNumeroRandom(numero) {  
  return numero + Math.random()  
}
```

- Tiene efectos de estados porque un agente externo como es `Math.random()` está afectando la fiabilidad con la que nos dará los resultados la función.

Scope

DEV.F
DESARROLLAMOS(PERSONAS);

ALCANCE (SCOPE)

ámbito global

```
let edad = 19;
let nombre = 'JUANITO BANANAS';

function MayorEdad( ) {
  if ( edad >= 18) {
    return nombre + " ERES MAYOR DE EDAD"
  }else if (edad < 18) {
    return nombre + " ERES MENOR DE EDAD"
  }
}
MayorEdad( )
```

ámbito local

```
function MayorEdad( ) {
  let edad = 16;
  if ( edad >= 18) {
    return "ERES MAYOR DE EDAD"
  }else if (edad < 18) {
    return "ERES MENOR DE EDAD"
  }
}
MayorEdad( )
```

SCOPE GLOBAL

SCOPE LOCAL

CORRECTO

Resultado:

Un poco de texto mismo ejemplo anterior pero con funciones anidadas

