2024

Primer Semestre

Funciones

Analista Programador

Profesor

Matías Verges

Definiciones Fundamentales

1 Estructurada, números de línea y GoTo

En los viejos lenguajes de programación como BASIC, escribías números al inicio de cada línea de código y usabas `GOTO` para saltar entre líneas.

2 JavaScript

A diferencia de aquel estilo, hoy en JavaScript definimos funciones con nombres y parámetros que organizan el código y lo hacen más reutilizable y fácil de seguir. Una vez declaradas, estas funciones pueden ser llamadas en cualquier parte del código, permitiendo ejecutar sus instrucciones múltiples veces sin necesidad de repetirlas.

3 Definir o declarar una función

Definir o declarar una función en programación significa especificar su nombre, los parámetros que recibe y las instrucciones que debe ejecutar cuando es llamada.

4 Porque necesitamos funciones?

Necesitamos funciones para modularizar y reutilizar código, lo que simplifica la programación y mejora la mantenibilidad de los proyectos.

Definiciones Fundamentales

5

Funciones como tipo de dato

Las funciones son consideradas un tipo de dato, específicamente objetos de primera clase, lo que significa que pueden ser asignadas a variables, pasadas como argumentos a otras funciones, y retornadas desde funciones, igual que otros tipos de datos.

6

Nombrando funciones

En JavaScript, los nombres de funciones y variables deben comenzar con una letra, un guión bajo (_) o un signo de dólar ($), y no pueden incluir espacios ni caracteres especiales, excepto el guión bajo y el dólar.

7

Parametros

Los parámetros de una función son variables que se definen en los paréntesis al declarar una función, y actúan como contenedores para los valores que se pasan a la función cuando es invocada.

8

Valor de retorno

La función realiza operaciones con los datos proporcionados y al finalizar, retorna un valor al contexto desde donde fue llamada.

// Sintaxis: Definir una función

```
// Declarar una función
function saludar() {
 alert("Hola, soy una funcion");
}

// Llamar a una función
saludar();
```

// Sintaxis: Parametros

```
// Declarar una funcion con parametros
function saludarConNombre(nombre) {
 alert("Hola, soy una funcion y mi nombre es  " + nombre);
}

// Llamar a una funcion con parametros
saludarConNombre("Juan");
```

// Sintaxis: Retorno

```
// Declarar una funcion con parametros y retorno
function sumar(a, b) {
 return a + b;
}

// Llamar a una funcion con parametros y retorno
let resultado = sumar(5, 10);
```

// Métodos de String

```
const url = "https://www.Google.com";
// metodo replace():
let urlBing = url.replace("Google", "bing");
console.log(urlBing);
// metodo slice():
let dominio = url.slice(12);
console.log(dominio);
// metodo toLowerCase():
let urlMinuscula = url.toLowerCase();
console.log(urlMinuscula);
// metodo toUpperCase():
let urlMayuscula = url.toUpperCase();
console.log(urlMayuscula);
```

// Métodos de String

```
const url = "https://www.Google.com ";
// metodo startsWith():
let siEmpiezaConHttp = url.startsWith("https");
console.log(siEmpiezaConHttp);
// metodo endsWith():
let siTerminaConCom = url.endsWith(".com");
console.log(siTerminaConCom);
// metodo includes():
let siIncluyeGoogle = url.includes("Google");
console.log(siIncluyeGoogle);
// metodo indexOf():
let posicionDeGoogle = url.indexOf("Google");
console.log(posicionDeGoogle);
// metodo lastIndexOf():
let ultimaPosicionDeO = url.lastIndexOf("o");
console.log(ultimaPosicionDeO);
```

// Variables externas vs locales

```
// Variables externas vs locales
let resultado = 99999;

function sumar(a, b) {
 let resultado = a + b;
 alert(resultado);
}

let num1 = 5;
let num2 = 10;
sumar(num1, num2);
alert(resultado) // 99999;
```

// return similar a break

```
// return corta la ejecución de la función
function sumar(a, b) {
 return a + b;
 alert("Hola");
}
```

Nombrar funciones

1 Nombres específicos y descriptivos

Los nombres deben reflejar claramente lo que hace la función o qué representa el parámetro.

2 Utiliza camelCase

En JavaScript, es convención usar camelCase para nombres de funciones y parámetros (ejemplo: calcularEdad).

3 Evita nombres genéricos o ambiguos

Nombres como data o info no aportan claridad sobre el contenido de la variable o función.

4 Parámetros con nombres significativos

Cada parámetro debe tener un nombre que describa claramente su propósito y contenido (ejemplo: precioInicial).

Materiales de referencia

JavaScript funciones:
https://es.javascript.info/function-basics

JavaScript sintaxis:
https://www.w3schools.com/js/js_syntax.asp

JavaScript arrays:
https://www.w3schools.com/js/js_arrays.asp

JavaScript operadores:
https://www.w3schools.com/js/js_operators.asp

Muchas gracias por su atención!