JavaScript Glossary

Here you have a list over the terms that are used in JavaScript.

AJAX (Asynchronous JavaScript and XML)

A technique for updating parts of a webpage without reloading the whole page using asynchronous HTTP requests.

Example: fetch('https://api.example.com/data').then(response => response.json()).then(data => console.log(data));

Array

A data structure that can hold multiple values under a single name, and these values can be accessed by an index.

Example: let numbers = [1, 2, 3, 4, 5];

Asynchronous

A way of programming where operations can proceed without blocking the main thread, often used with AJAX, Promises, and async/await.

Example: async function fetchData() { let response = await fetch(url); }

Boolean

A data type that can have a value of true or false.
Example: let isAvailable = true;

Callback

A function given as an argument to another function and executed after an event or task is completed.

Example: function fetchData(callback) { setTimeout(() => { callback('data'); }, 1000); }

Class

A syntactic sugar for prototype-based inheritance in JavaScript, allowing the creation of objects and managing inheritance.

Example: class Person { constructor(name) { this.name = name; } }

Closure

A function along with references to its surrounding lexical environment, allowing a function to access variables from an outer function even after the outer function has returned.

Example: function outer() { let x = 10; return function inner() { console.log(x); }; }

DOM (Document Object Model)

A programming interface for HTML and XML documents, representing the document as a tree structure where each node is a part of the document.

Example: document.getElementById('example').innerText = 'Hello, World!';

Event

An action or occurrence detected by the system that JavaScript can react to, such as a user click or a loaded page.

Example: button.addEventListener('click', () => { alert('Button clicked!'); });

Function

A block of code that can be defined once and executed many times.

Example: function greet(name) { return 'Hello, ' + name; }

Hoisting

JavaScript's default behavior of moving variable and function declarations to the top of their scope before code execution.

Example: console.log(x); var x = 5; // undefined

JSON (JavaScript Object Notation)

A lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate.

Example: let obj = JSON.parse('{"name":"John", "age":30}');

Loop

A control structure that repeats a sequence of instructions until a specific condition is met.

Example: for(let i = 0; i < 5; i++) { console.log(i); }

Null

A special keyword that represents a deliberate non-value, meaning "no value".

Example: let value = null;

Object

A collection of key-value pairs (properties and methods) used to store and manipulate data.

Example: let person = { name: 'Alice', age: 25 };

Promise

An object representing the eventual completion (or failure) of an asynchronous operation and its resulting value.

Example: let promise = new Promise((resolve, reject) => { resolve('Success!'); });

Prototype

A mechanism by which JavaScript objects can inherit properties and methods from one another.

Example: function Person(name) { this.name = name; } Person.prototype.greet = function() { console.log('Hello, ' + this.name); };

Recursion

A technique where a function calls itself directly or indirectly to solve a problem.

Example: function factorial(n) { if (n === 0) { return 1; } return n * factorial(n - 1); }

Scope

The current context of code, which determines the visibility and accessibility of variables.

Example: let x = 10; function scopeTest() { let y = 20; console.log(x, y); }

String

A sequence of characters used to represent text.

Example: let greeting = 'Hello, World!';

This

A keyword that refers to the object it belongs to in the current code context.

Example: let person = { name: 'Alice', greet() { console.log('Hello, ' + this.name); } };

Undefined

A variable that has been declared but not assigned a value.

Example: let x; console.log(x); // undefined

Variable

A container for storing data that can be changed during program execution.

Example: let age = 25;

XMLHttpRequest

An API for transferring data between a web browser and a server, often used for AJAX requests.

Example: let xhr = new XMLHttpRequest(); xhr.open('GET', 'https://api.example.com/data'); xhr.send();