cts

This chapter documents all of JavaScript's standard, built-in objects, including their methods and properties.

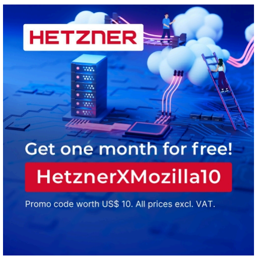
The term "global objects" (or standard built-in objects) here is not to be confused with **the global object**. Here, "global objects" refer to **objects in the global scope**.

The **global object** itself can be accessed using the `this` operator in the global scope. In fact, the global scope **consists of** the properties of the global object, including inherited properties, if any.

Other objects in the global scope are either created by the user script or provided by the host application. The host objects available in browser contexts are documented in the API reference.

For more information about the distinction between the DOM and core JavaScript, see JavaScript technologies overview.

In this article



Cloud Server in Singapore!Starting at US$ 7.59 per month.Try now!
Ad

Standard objects by category

Value properties

These global properties return a simple value. They have no properties or methods.

- `globalThis`
- `Infinity`
- `NaN`
- `undefined`

Function properties

These global functions—functions which are called globally, rather than on an object—directly return their results to the caller.

- `eval()`
- `isFinite()`
- `isNaN()`
- `parseFloat()`
- `parseInt()`
- `decodeURI()`
- `decodeURIComponent()`
- `encodeURI()`
- `encodeURIComponent()`
- `escape()` Deprecated
- `unescape()` Deprecated

Fundamental objects

These objects represent fundamental language constructs.

- `Object`
- `Function`
- `Boolean`
- `Symbol`

Error objects

Error objects are a special type of fundamental object. They include the basic `Error` type, as well as several specialized error types.

- `Error`
- `AggregateError`
- `EvalError`
- `RangeError`
- `ReferenceError`
- `SuppressedError`
- `SyntaxError`
- `TypeError`
- `URIError`
- `InternalError` Non-standard

Numbers and dates

These are the base objects representing numbers, dates, and mathematical calculations.

- `Number`
- `BigInt`
- `Math`
- `Date`
- `Temporal`

Text processing

These objects represent strings and support manipulating them.

- `String`
- `RegExp`

Indexed collections

These objects represent collections of data which are ordered by an index value. This includes (typed) arrays and array-like constructs.

- Array
- TypedArray
- Int8Array
- Uint8Array
- Uint8ClampedArray
- Int16Array
- Uint16Array
- Int32Array
- Uint32Array
- BigInt64Array
- BigUint64Array
- Float16Array
- Float32Array
- Float64Array

Keyed collections

These objects represent collections which use keys. The iterable collections (Map and Set) contain elements which are easily iterated in the order of insertion.

- Map
- Set
- WeakMap
- WeakSet

Structured data

These objects represent and interact with structured data buffers and data coded using JavaScript Object Notation (JSON).

- ArrayBuffer
- SharedArrayBuffer
- DataView
- Atomics

- `JSON`

Managing memory

These objects interact with the garbage collection mechanism.

- `WeakRef`
- `FinalizationRegistry`

Control abstraction objects

Control abstractions can help to structure code, especially async code (without using deeply nested callbacks, for example).

- `Iterator`
- `AsyncIterator`
- `Promise`
- `GeneratorFunction`
- `AsyncGeneratorFunction`
- `Generator`
- `AsyncGenerator`
- `AsyncFunction`
- `DisposableStack`
- `AsyncDisposableStack`

Reflection

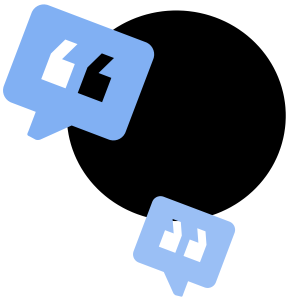
- `Reflect`
- `Proxy`

Internationalization

Additions to the ECMAScript core for language-sensitive functionalities.

- `Intl`
- `Intl.Collator`
- `Intl.DateTimeFormat`

- `Intl.DisplayNames`
- `Intl.DurationFormat`
- `Intl.ListFormat`
- `Intl.Locale`
- `Intl.NumberFormat`
- `Intl.PluralRules`
- `Intl.RelativeTimeFormat`
- `Intl.Segmenter`



Help improve MDN

Was this page helpful to you?

👍Yes 👎No

Learn how to contribute

This page was last modified on Jul 29, 2025 by MDN contributors.

View this page on GitHub • Report a problem with this content