whats globalobject in js, window in chrome, and global in ndoejs, globalthis

Here's a clear comparison of the terms you asked about and their usage in various JavaScript environments:

1. The Global Object in JavaScript

- The global object is an object that provides variables and functions available **anywhere** in your code, acting as the top-level scope container.
- The "name" of the global object varies by environment.

2. window (in Browsers / Chrome)

- In web browsers like Chrome, the global object is called `window`.
- All global variables and functions you declare (with `var` and traditional `function` declarations) become properties of `window`.
- For example:

```
window.alert("Hello"); // Equivalent to alert("Hello")
```

3. global (in Node.js)

- In Node.js, the global object is called `global`.
- Declaring a variable with `var`, `let`, or `const` at the top level of a Node.js module **does not** add it to `global` (due to Node's module system).
- Use `global` to set truly global properties/functions across modules, but this practice is generally discouraged for maintainability.

4. globalThis (Universal: Browsers, Node.js, etc.)

- Introduced in **ECMAScript 2020**.
- `globalThis` is a **standardized way to access the global object**, whatever JavaScript environment you're in—browser, Node.js, web worker, etc.
- Replaces the need for environment-specific checks like:

```
// Old way:
var getGlobal = function() {
  if (typeof self !== "undefined") { return self; }
  if (typeof window !== "undefined") { return window; }
  if (typeof global !== "undefined") { return global; }
  throw new Error("unable to locate global object");
};
// New way, universally:
globalThis
```

- Anywhere you would have used `window` (browser) or `global` (Node.js), just use `globalThis`—it points to the correct object everywhere.[1] [2] [3] [4]

5. Practical Summary Table

Environment	Global Object Name	Standardized Alias
Browser (Chrome)	`window`	`globalThis`
Node.js	`global`	`globalThis`
Web Worker	`self`	`globalThis`

6. Key Takeaways

- Use `globalThis` for maximum cross-environment compatibility.
- `window` is browser-specific, and `global` is Node.js-specific.
- With `globalThis`, you no longer need to write environment-detection code to access the global scope.[5] [3] [4] [1]

References:

- [MDN: globalThis][1]
- [CoreUI Blog][2]
- [Educative.io][3]
- [MDN: Glossary Global object][4]
- [Dev.to][5]

: web:1[1]
: web:3[2]
: web:4[5]
: web:5[3]
: web:7[4]

⁂

1. https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/globalThis

2. https://coreui.io/blog/what-is-globalthis-in-javascript/
3. https://www.educative.io/answers/what-is-globalthis-in-javascript
4. https://developer.mozilla.org/en-US/docs/Glossary/Global_object
5. https://dev.to/dipakahirav/javascripts-globalthis-the-universal-object-you-didnt-know-you-needed-26oj