
MDN Web Docs
moz://a

Scope

The current context of execution. The context in which values and **expressions** are "visible" or can be referenced. If a **variable** or other expression is not "in the current scope," then it is unavailable for use. Scopes can also be layered in a hierarchy, so that child scopes have access to parent scopes, but not vice versa.

A **function** serves as a **closure** in JavaScript, and thus creates a scope, so that (for example) a variable defined exclusively within the function cannot be accessed from outside the function or within other functions. For instance, the following is invalid:

```
function exampleFunction() {
    var x = "declared inside function";  // x can only be used in exa
    console.log("Inside function");
    console.log(x);
}

console.log(x);  // Causes error
```

However, the following code is valid due to the variable being declared outside the function, making it global:

```
var x = "declared outside function";

exampleFunction();

function exampleFunction() {
    console.log("Inside function");
    console.log(x);
}

console.log("Outside function");
console.log(x);
```

Learn more

General knowledge

- Scope (computer science) on Wikipedia