A **function's this keyword** behaves a little differently in JavaScript compared to other languages. It also has some differences between [strict mode](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions_and_function_scope/Strict_mode) and non-strict mode.

In most cases, the value of this is determined by how a function is called. It can't be set by assignment during execution, and it may be different each time the function is called.

**Global Context**

In the global execution context (outside of any function), this refers to the global object whether in strict mode or not.

**Function context**

Inside a function, the value of this depends on how the function is called.

Simple call

Since the following code is not in [strict mode](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Strict_mode), and because the value of this is not set by the call, this will default to the global object, which is [window](https://developer.mozilla.org/en-US/docs/Web/API/Window) in a browser.

**The bind method**

ECMAScript 5 introduced [Function.prototype.bind()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function/bind" \o "The bind() method creates a new function that, when called, has its this keyword set to the provided value, with a given sequence of arguments preceding any provided when the new function is called.). Calling f.bind(someObject)creates a new function with the same body and scope as f, but where this occurs in the original function, in the new function it is permanently bound to the first argument of bind, regardless of how the function is being used.

**Arrow functions**

In [arrow functions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions), this retains the value of the enclosing lexical context's this. In global code, it will be set to the global object.