This keyword behaves differently based on the place where you are using

**“this” refers to global object**

|  |
| --- |
| function foo () { |
|  | console.log("Simple function call"); |
|  | console.log(this === window); |
|  | } |
|  |  |
|  | foo(); //prints true on console |
|  | console.log(this === window) //Prints true on console. |

**Immediately Invoked Function Expression (IIFE)**

|  |
| --- |
| (function(){ |
|  | console.log("Anonymous function invocation"); |
|  | console.log(this === window); |
|  | })(); |
|  | // Prints true on console |

*If****strict mode****is enabled for any function then the value of “this” will be “undefined” as in strict mode, global object refers to undefined in place of windows object.*

function foo () {

'use strict';

console.log("Simple function call")

console.log(this === window);

}

foo(); //prints false on console as in “strict mode” value of “this” in global execution context is undefined.

function Person(fn, ln) {

this.first\_name = fn;

this.last\_name = ln;

this.displayName = function() {

console.log(`Name: ${this.first\_name} ${this.last\_name}`);

}

}

let person = new Person("John", "Reed");

person.displayName(); // Prints Name: John Reed

let person2 = new Person("Paul", "Adams");

person2.displayName(); // Prints Name: Paul Adams