**This Keyword**

**--Saksham Khandelwal**

A **function's**this**keyword** behaves a little differently in JavaScript compared to other languages. It also has some differences between strict mode and non-strict mode.

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

Let’s take some examples –

function foo() {

    console.log(this);

    console.log(this === window);

}

foo();

In this example this keyword will be the window element.

var obj = {};

obj.foo = function() {

  console.log("Inside obj foo");

  console.log(this);

}

obj.foo();

In this example this will be the object itself.

function Person(fn, ln) {

    this.first\_name = fn;

    this.last\_name = ln;

    this.display\_name = function () {

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

        console.log(this);

    }

}

let person = new Person("Saksham", "Khandelwal");

person.display\_name();

let person2 = new Person("Saket", "Singh");

person2.display\_name();

In this example this will be the newly created instance.

Let’s see some methods like call, apply and bind.

**Call :-**

let obj = {num: 3};

let addNumbers = function(a,b,c) {

  console.log(this);

  return this.num + a + b + c;

}

// addNumbers(obj, 1, 10, 11);

addNumbers.call(obj, 1, 10, 11);

In this example suppose if we want to return the sum of all numbers but if we run the function directly how it will take the value this.num that’s why we use call method to get the value of the object.

**Apply :-**

let obj = {num: 3};

let addNumbers = function(a,b,c) {

  console.log(this);

  return this.num + a + b + c;

}

let arr = [1,4,6];

addNumbers.apply(obj, arr);

Suppose we want to pass the array so we can use apply method here.

**Bind :-**

let obj = {num: 3};

let addNumbers = function(a,b,c) {

  console.log(this);

  return this.num + a + b + c;

}

let bindFunc = addNumbers.bind(obj, 1, 10, 11);

bindFunc();

It returns the function.