# 1. 了解this吗 call bind apply具体指的是什么?

**this绑定的四条规则：默认全局，隐式绑定obj.fun, 显式绑定call，apply包括硬绑定，new 绑定 优先级：new和call/apply无法一起使用 new > 硬绑定或显式绑定>隐式绑定>默认**

**箭头函数不使用this的四种标准规则，而是根据外层（函数或全局）作用域来决定this,和之前代码中的self=this一样**

Object.prototype.toString.call()

1. 获取this对象的[[Class]]属性的值.
2. 计算出三个字符串"[object ", 第一步的操作结果Result(1), 以及 "]"连接后的新字符串.
3. 返回第二步的操作结果Result(2).

var bar = new obj1.foo(4) obj1.a // 2 bar.a = 4

- 使用call bind apply 如果要fun.call(null)或者不传递this，非严格模式指向全局，严格模式指向undefined 为了避免修改全局变量 最好指向一个DMZ对象 var empty = Object.create(null) fun.call(empty) https://www.cnblogs.com/linweinb/p/8270780.html
- 隐式绑定 obj.fun.call() this为obj
  function foo() { console.log(this.a); } var obj = { a: 2, foo: foo } obj.foo(); // 2
- 显式绑定 function foo() { console.log(this.a); } var obj = { a: 2 }; foo.call(obj);
- call() apply() bind()的区别 都是用来改变this指向 call()第一个参数是this指向的对象，后面传入的是参数列表，参数可以是任意类型，当第一个参数为null或undefined的时候指向window
- apply第一个参数是this指向的对象，第二个参数是数组,参数解构 apply应用： 求数组中的最大值： var arr = [10,9,1,11,2] var max = Math.max.apply(null,arr) 相当于Math.max(10,9,1,11,2)
- call改变this指向后直接执行函数
- bind改变this指向后不会执行函数，会返回一个永久绑定新this的函数

1. call 用法： var obj = {a:1} function A(param) { console.log(this.a,param) } A.call(obj,333) // A函数调用 this指向obj

A.myCall Function.prototype.myCall = function(context,...args) { context = context || window // 对象的方法调用时 其this指向对象 所以此处this为调用的函数A context.fn = this // 相当于obj.fn = A // context.fn执行 所以此时函数执行的时候this指向变为context let res = context.fn(...args) delete context.fn return res

}

Function.prototype.myApply = function(context, argsArr) { context = context || window context.fn = this // const fnSymbol = Symbol('fn') context[fnSymbol] = this let res; if(argsArr && argsArr.length) { res = context.fn(...argsArr) }else { res = context.fn() } delete context.fn return res }

var obj = {a:2} function A() { console.log(this.a) }

Function.prototype.myBind = function(context,...args) { context = context || window context.fn = this return function(...newArgs) { args = args.concat(newArgs) let res = context.fn(...args) delete context.fn return res } }

# 2. 如果一个构造函数，bind了一个对象，用这个构造函数创建出的实例会继承这个对象的属性吗？为什么

不会，因为this绑定四大原则，new绑定的优先级高于bind显示绑定，new进行构造函数调用时候，会创建一个新对象，这个新对象会代替bind的对象绑定，作为此函数的this，并且在此函数没有返回对象的情况下，返回这个新对象。

var obj = { a:1 } function A() { console.log(this.a) } var a = 3

var bindA = A.bind(obj) bindA() // this是obj 1

var c = new bindA()

# 3.箭头函数和普通函数的区别？箭头函数能当构造函数吗

普通函数this在运行时绑定，取决于调用者和调用位置，四种this绑定规则

箭头函数 根据父级作用域来觉得this

箭头函数常用于回调函数中，如事件处理和定时器

箭头函数没有原型，没有this，没有super，没有arguments，没有new

箭头函数不能new调用

- 箭头函数需要注意的地方 当要求动态上下文的时候，就不能使用箭头函数，也就是this的固定化 在使用箭头函数定义函数的时候，this的指向是定义时所在的对象，而不是使用时所在的对象 不能用作构造函数，不能使用new命令，否则会抛出一个错误 不能使用arguments对象 不能使用yield命令