**底层原理**

#### 闭包

什么是闭包？闭包的特性？

1、函数嵌套函数

2、函数内部可以引用外部的参数和变量

3、参数和变量不会被垃圾回收机制回收

闭包的缺点：变量长期存在占用内存，在不使用时要做到变量及时释放

**Demo1**

function couter() { var num = 0;

return { // 增 up: function() { num++ },

down: function() { num--; }// 获取

getNum: function() {console.log(num);}

}

}

var c1 = couter();

c1.up();

c1.up();

c1.getNum(); // 2

**Demo2**

for (var i = 0; i < 5; ++i) {

func[i] = function() {

console.log(i);

}

(function(i) {

func[i] = function() {

console.log(i);

}

})(i);

}

func[3](); // 3

#### 高阶函数

什么是高阶函数？高阶函数的特点？

1、函数可以作为参数被传递

2、函数可以作为返回值输出

常见的高阶函数Debounce和Throttle

**Debounce（防抖）**

function debounce(fn,delay){

delay = delay || 1000

var timer = null

return function() {

var arg = arguments

clearTimeout(timer)

timer = null

timer = setTimeout(function(){

fn.apply(this,arg)

},delay)

}

}

Var num=0

window.onscroll = debounce(function(e){

console.log(e,++num)

},2000)

**Throttle(节流)**

function(fn,delay){

var timer = null

var lastTime = 0

return function(){

var arg = arguments

clearTimeout(timer)

timer = null

var now = new Date().getTime()

if(lastTime === 0) {

lastTime = now

}

if(now – lastTime >= delay) {

fn.apply(this, arg)

lastTime = now

} else {

timer = setTimeout(function(){

fn.apply(this, arg)

lastTime = 0

},delay)

}

}

}

Var num=0

window.onscroll = debounce(function(e){

console.log(e,++num)

},2000)

#### 柯里化（currying）

什么是柯里化？柯里化的特点

又称为部分求值，是把接受多个参数的函数变换成接受一个单一参数（最初函数的第一个参数）的函数，并且返回一个新的函数的技术，新函数接受余下参数并返回运算结果

**Demo**

function verify(reg){

return function(text){

return reg.test(text)

}

}

const isNum = verify(/\d/g)

const isString = verify(/[a-z]/g)

console.log(isNum(1))

console.log(isString(1))