正课:

1. ES5:

2. ES6:

1. ES5:

call, apply, bind

共同的作用: 替换函数内的错误的this

何时: 只要函数中的this不是想要的，就用这三个函数替换

不同:

call, apply:

什么是: 在调用函数时，临时替换this

如何: 要调用的函数.call(替换this的对象, 参数值, ...)

如果给定的参数值，是放在数组中的。就需要用apply代替call，先将数组打散，再传入函数。

bind:

什么是: 创建一个很原函数一模一样的新函数，但永久绑定this和部分参数。

何时: 只要希望给一个函数永久绑定this时

如何: var fun=函数.bind(替换this的对象, 参数值, ... )

固定用法: 今后只要替换回调函数中的this时，都用bind，而不用call

数组API:

1. 判断:

every(): 判断数组中是否所有元素都符合条件

some(): 判断数组中是否包含符合条件的元素

如何: var bool=arr.every(function(elem,i,arr){

//elem: 自动获得当前元素值

//i: 自动获得当前位置

//arr: 自动获得当前数组对象

return 判断条件

})

原理: every会用回调函数在每个元素上执行一次，

每次执行，会自动传入当前元素值，当前位置，当前数组对象本身。

2. 遍历:

forEach(): 对原数组中每个元素执行相同的操作

map(): 依次取出原数组中的每个元素，执行相同操作后，再放入新元素

相同: 都遍历所有元素

不同: forEach()直接修改原数组

map() 不修改原数组，创建新数组

如何:

arr.forEach(function(elem,i,arr){

arr[i]=新值

})

arr.map(function(elem,i,arr){

return 新值;

})

总结: forEach，算普通for循环的简化

问题: forEach无法：控制遍历的顺序和步调，以及无法随时退出

3. 过滤和汇总:

过滤: 复制出原数组中符合条件的元素，组成新数组返回

var sub=arr.filter(function(elem,i,arr){

return 判断条件

})

汇总: 将数组中所有元素值，汇总出一个最终结果

//仅以累加举例子:

var sum=arr.reduce(function(prev,elem,i,arr){

//截止到当前位置之前，所有元素的临时汇总值

return prev+elem;

}, base)

2. ES6:

let: 专门代替var 用于声明变量

为什么: js中的var会被声明提前，破坏执行顺序

js中没有块级作用域，导致块内的变量影响外部

块: 程序中的{}结构

比如: if...else... while for switch

原理: let不允许提前使用未声明的变量

let会自动生成匿名函数自调

for 和 let:

for(let i=0;i<xxx;i++)

=>

for(var i=0;i<xxx;i++)

(function(i){//i受闭包保护的变量

})(i)

参数增强:

默认值default: 给函数参数添加默认值，如果调用时，未指定参数值，自动使用默认值代替

原理: fromi=fromi||0;

剩余参数rest: 代替arguments

arguments的问题:

1. 不是数组, 而是类数组对象

2. arguments必须获得全部参数

rest: 1. 是数组

2. 可选择获取部分剩余参数

强调: 必须放在参数列表结尾

如何: function fun(参数1, 参数2, ...数组名){

//数组会接住除参数1,参数2外的所有剩余参数

}

原理: 将任意类数组对象转为数组:

var nums=Array.prototype.slice.call(arguments);

其实可以简写: var nums= [].slice.call(arguments)

选取部分参数:

var nums= [].slice.call(arguments, fromi)

散播/打散spread: 代替apply

如何: 在调用函数时: fun(参数, ...数组)

...可打散数组为单个参数值再传入

箭头函数: 简化所有回调函数和匿名函数自调

如何:

去掉function，改箭头

如果只有一个参数可省略()

如果函数体只有一句话，可省略{}

如果唯一一句话还是return则省略return

问题: 箭头函数会让内外this通用

解决: 事件处理函数，尽量不要用箭头函数简化

模板字符串: 代替+执行字符串拼接

如何: 整个字符串模板，必须用` `包裹

优点: 1. 支持换行

2. 支持动态执行表达式: 必须放在${表达式}

解构:

什么是: 从一个大的对象/数组中仅选取自己需要的保存在变量中使用。

如何: 3种情况:

1. 数组解构: 下标对下标

var date=[2018, 4, 25];

var [y,m,d]=date;

console.log(y, m, d);

2. 对象解构: 属性对属性:

3. 参数结构: 其实就是对象解构:

何时: 今后不确定传入参数的个数和顺序时

如何: 2步:

1. 定义函数时:

function fun({参数1, 参数2, ... ... })

2. 调用函数时:

fun({参数1:值1, 参数2: 值2, ... })

后果: 方法定义中的参数，获得调用时对象中的属性值

优点: 与传入参数的个数和顺序无关