1. 红宝书22章
2. 作用域安全的构造函数

function Person(name,age,job){

this.name=name;

this.age=age;

this.job=job;

}

var person=Person("name1",20,"worker");

console.log(window.name);//name1

如果没有用new操作符调用构造函数，由于this对象是在运行时绑定是，this会映射到全局对象window上。

修改：

function Person(name,age,job){

if(this instanceof Person){

this.name=name;

this.age=age;

this.job=job;

}

else{

return new Person(name,age,job);

}

}

var person=Person("name1",20,"worker");

console.log(window.name);//""

console.log(person.name);//"name1"

如果使用构造函数窃取模式继承且不使用原型链，这个继承可能会被破坏。

使用构造函数窃取结合原型链或寄生组合可以解决这个问题。

function Polygon(sides){

if(this instanceof Polygon){

this.sides=sides;

this.getArea=function(){

return 0;

}

}

else{

return new Polygon(sides);

}

}

function Rectangle(width,height){

Polygon.call(this,2);

this.width=width;

this.height=height;

this.getArea=function(){

return this.width\*this.height;

}

}

Rectangle.prototype=new Polygon();

var rect=new Rectangle(5,10);

console.log(rect.sides);//2

1. 惰性载入函数

惰性载入函数表示函数的分支仅会发生一次。

实现方法：在函数被调用时再处理函数，在第一次调用的过程中，该函数会被覆盖为另一个按合适方式执行的函数。在分支里：fn=function(){}在声明时就指定适当的函数。在分支里：return function(){}

1. 函数绑定

只要是将某个函数指针以值的形式进行传递，同时该函数必须在特定环境中执行，被绑定函数的效用就显现出来了。

常常和回调函数和事件处理程序一起使用，以便在将函数作为变量传递的同时保留代码执行环境。

var handler={

message:"Event handled",

handleClick:function(event){

console.log(2);

console.log(this.message+":"+event.type);

}

}

var btn=document.getElementById("btn");

EventUtil.addHandler(btn,"click",handler.handleClick.bind(handler));

1. 函数柯里化

把接受多个参数的函数变换成接收单一参数的函数，并且返回接受余下的参数而且返回结果的新函数。

与函数绑定的共同点：使用闭包返回一个函数。区别：当函数被调用时，返回的函数还需要设置一些传入的参数。

1. 放篡改对象

一旦把对象定义为防篡改，就无法撤销了

* 1. 不可扩展对象

不能添加属性，可以修改和删除已经存在的属性

var person={name:"my name"};

Object.preventExtensions(person);

person.age=20;

console.log(person.age);//undefined

person.name="new name";

console.log(person.name);//new name

console.log(Object.isExtensible(person));//false

* 1. 密封的对象

不能添加和删除属性，可以修改属性值

被密封说明也不能扩展

var person={name:"my name"};

Object.seal(person);

person.age=20;

console.log(person.age);//undefined

person.name="new name";

console.log(person.name);//new name

console.log(Object.isSealed(person));//true

* 1. 冻结对象

不能增加和删除属性，也不能修改属性值

var person={name:"my name"};

Object.freeze(person);

person.age=20;

console.log(person.age);//undefined

person.name="new name";

console.log(person.name);//my name

console.log(Object.isFrozen(person));//true

被冻结说明不能扩展而且密封了

1. 定时器

在js中没有代码是立即执行的，但一旦进入空闲则尽快执行。

指定的时间间隔表示何时将定时器的代码添加到队列，而不是何时实际执行代码。

setTimeout(function(){

//

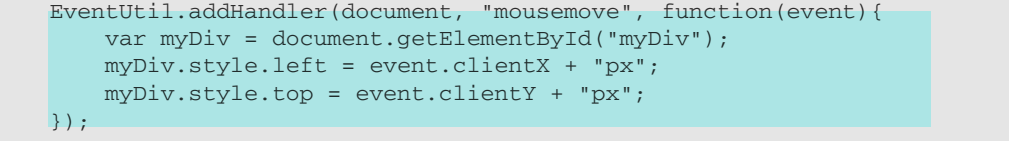
setTimeout(arguments.callee,interval);

},interval)

链式调用setTimeout 模仿setInterval 推荐这种

1. 拖放

鼠标拖尾：



拖放：p622

1. 博客
2. 检测类型

function isArray(obj){

return Object.prototype.toString.call(obj)=="[object Array]";

}

console.log(isArray([1,2,3]));//true

function isFunction(obj){

return Object.prototype.toString.call(obj)=="[object Function]";

}

var fn=function(){};

console.log(isFunction(fn));//true

function isRegExp(obj){

return Object.prototype.toString.call(obj)=="[object RegExp]";

}

console.log(isRegExp(/a/));//true

return window.JSON&&Object.prototype.toString.call(JSON)=="[object JSON]";//是否为原生json

1. css计数器

body,h2,h3,p{margin: 0;}

.show{counter-reset: c2;}

.show h2{counter-reset: c3 cp;counter-increment: c2;}

.show h3{counter-reset: cp;counter-increment: c3;text-indent: 2em;}

.show p{counter-increment: cp;text-indent: 4em;}

.show h2::before{content: counter(c2);}

.show h3::before{content: counter(c2) '.' counter(c3);}

.show p::before{content: counter(c2) '.' counter(c3) '.' counter(cp)}

counter-reset:计数器标识符 重置的数字 默认为none

counter-reset:c1 4 表示将c1的计数器重置为4

counter-reset c1 表示将c1的计数器重置为0

counter-increment 计数器标识符 递增的数字

counter-increment: c1 4 将c1的递增设为4

counter-increment: c1 将c1的递增设为1

还需要结合伪元素、cotent和counter()来使用

div::before{

content:counter(c1,upper-roman)

}

第一个参数是计数器

第二个参数是计数器样式 默认为数字 none disc circle square decimal lower-roman upper-roman lower-alpha upper-alpha lower-greek decimal-leading-zero

1. 伪元素

:first-letter

元素第一个字母

只能与块级元素相连

选择器与左大括号有空格时，ie6才支持

:first-line

元素第一行

只能与块级元素相连

选择器与左大括号有空格时，ie6才支持

:before :after

默认为行内元素

继承可继承的属性

ie7中必须声明!DOCTYPE否则不起作用

::selection

匹配用户选中的部分

firefox浏览器需要加-moz-前缀

只支持双冒号写法

只支持颜色和背景色2个样式

钉子：

.box::before{display: block;content: "钉子";height: 50px;width: 50px;border-radius: 50%;background-color: black;color: white;font-weight: bold;text-align: center;line-height: 50px;}

.box::after{display: block;content: "";width: 0;height: 0;border: 25px solid transparent;border-top: 50px solid black;margin-top: -20px;}

图片叠加效果

body{margin: 0;}

.box{position: relative;margin: 30px auto;width: 300px;}

.box-img{position: absolute;z-index: 1;border: 5px solid gray;width: 100%;}

.box::before,.box::after{content: "";position: absolute;background-color: palevioletred;width: 300px;height: 200px;border: 5px solid gray;}

.box::before{left: -10px;top: 0;transform: rotate(-5deg);}

.box::after{top: 4px;left: 0;transform: rotate(4deg);}

1. js jq css选择器

<https://www.cnblogs.com/ooo0/p/6115324.html>

1. 继承

<https://www.cnblogs.com/humin/p/4556820.html>

1. 原生js常用方法
2. 布尔值

假值：0 “” null undefined false NaN

其余都为真值

1. str+num



如果str里面有字母不能转换成数字，则结果为NaN

1. undefined void 0

undefined是不可靠的

undefined可当作局部变量并赋值，当undefined作为全局变量时不能赋值

var undefined;

function foo2(){

undefined=1;

console.log(undefined);//undefined

}

foo2();

所以应该用void 0 或void(0)代替undefined

1. 字符串也有length属性

不能用length来判断是否为数组，应该用Object.toString.call(obj)===’[object Array]’

1. 创建一个随机数组或者打乱数组

arr.sort(()=>{

return Math.random()-0.5;

})

arr.sort((a,b)=>{

return a>Math.random()\*10;

})

注：a-b返回从小到大排序

1. 去除前、后、前后空格

//type:1-所有空格 2-前后空格 3-前空格 4-后空格

function trim(str,type){

switch(type){

case 1:return str.replace(/\s+/g,"");

case 2:return str.replace(/(^\s\*)|(\s\*$)/g, "");

case 3:return str.replace(/(^\s\*)/g, "");

case 4:return str.replace(/(\s\*$)/g, "");

default:return str;

}

}

1. 首字母大/小写 全部大/小写

text-transform:capitalize每个单词首字母大写 uppercase全部大写 lowercase全小写

1. 重复字符串

console.log("abc".repeat(3));

1. 替换字符串中的一部分

console.log("abcd".replace(/a/,"1"));