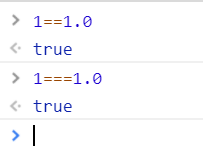
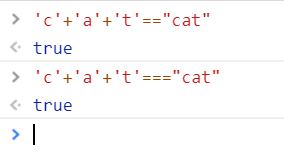
1. 红宝书第12章DOM1和DOM2
2. 数字类型在内部被表示为64位浮点数



1. js中所有字符都是16位的



1. 文字在div底部

方法一：line-height设置为div的接近2倍

如果是p元素要将p元素的margin-top margin-bottom设置为0px

方法二：position:absolute

1. hr边缘不光滑

.hr2{width: 100%;height: 1px;border: none;border-top: 1px solid red;}



用after代替

p::after{content: "";width: 100%;height: 1px;

border-top: 1px solid red;display: block;}

1. 下划线且首字母大一点

p::first-letter{

font-size: 2em;

}

p::after{content: "";width: 100px;height: 1px;

border-top: 1px solid black;display: block;}

1. 首字母变大

单行变大

p::first-letter{

font-size: 2em;

}

多行变大：

p::first-letter{

font-size: 2.5em;

float: left;

margin-top: -10px;

}

p{word-break: break-all;}

1. 私有变量和特权方法
2. 定义

私有变量：在函数内定义的变量，不能在函数外访问到。包括参数，局部变量和函数内部定义的其他函数

特权方法指能访问私有变量的公有方法，比如闭包

1. 创建方式

构造函数： 每个实例都会创建同一组方法

var privateVar=100;

function myObject(){

//私有变量和私有函数

var privateVar=10;

function privateFun(){

alert(this.privateVar);//100

alert(privateVar);//11

return true;

}

//特权方法

this.publicMethod=function(){

privateVar++;

return privateFun();

}

}

new myObject().publicMethod();

静态私有变量

var privateVar=100;

(function(){

//私有变量和私有函数

var privateVar=10;

function privateFun(){

alert(privateVar);//11

alert(this.privateVar);//100

return false;

}

myObject=function(){};

myObject.prototype.publicMethod=function(){

privateVar++;

return privateFun();

}

})();

new myObject().publicMethod();



1. 作用

利用私有和特权成员，隐藏不应该被直接修改的数据

function Person(name){

this.getName=function(){

return name;

}

this.setName=function(value){

name=value;

}

}

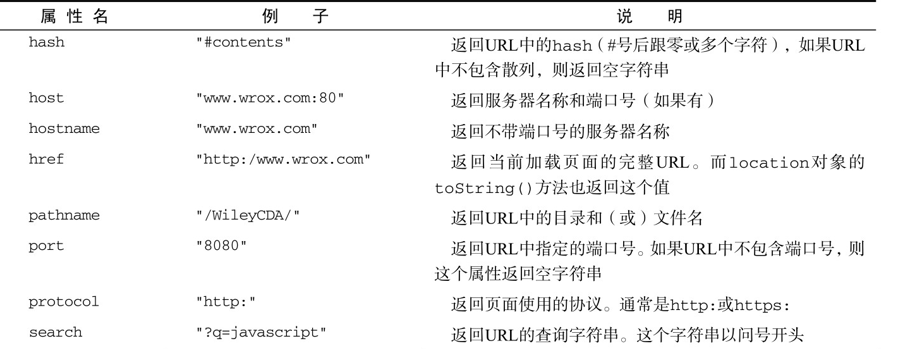
var person=new Person("name1");

console.log(person.getName());//name1

person.setName("name2");

console.log(person.getName());//name2

1. BOM的location对象
   1. location既是window对象的属性，也是document对象的属性，还可以单独使用
   2. 属性



var URL = 'http://localhost:8080/index.html#text?q=test';

console.log(location.hash);//#text?q=test

console.log(location.host);//localhost:8080

console.log(location.hostname);//localhost

console.log(location.href);//http://localhost:8080/index.html#text?q=test

console.log(location.pathname);///index.html

console.log(location.port);//8080

console.log(location.protocol);//http:

console.log(location.search);''

location.search只能取到?后面和#前面的内容

* 1. 解析查询字符串

function getQueryStringArgs(){

//取得查询字符串并去掉开头的问号

var qs = location.search.length > 0 ? location.search.substring(1) : "";

//保存数据的对象

var args = {};

//取得每一项

var items = qs.length ? qs.split("&") : [];

var item,name,value;

var len = items.length;

//逐个将每一项添加到args对象中

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

item = items[i].split("=");

name = decodeURIComponent(item[0]);

value = decodeURIComponent(item[1]);

if(name.length){

args[name] = value;

}

}

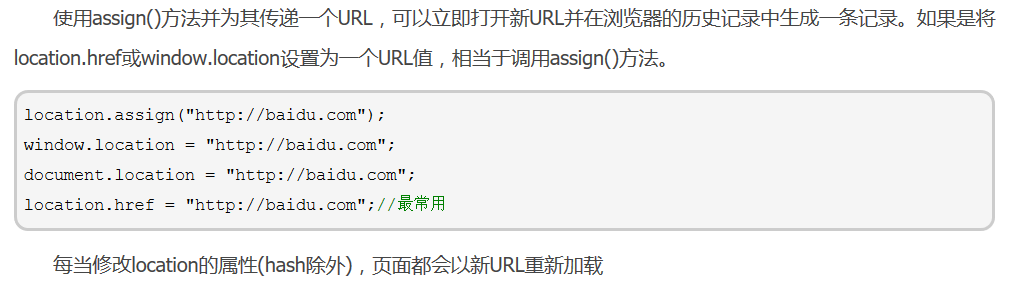
return args;

}

location.search = "?name=abc&password=123&callback=fn";

console.log(getQueryStringArgs());//[name: "abc", password: "123", callback: "fn"]

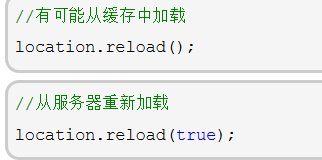
* 1. assign



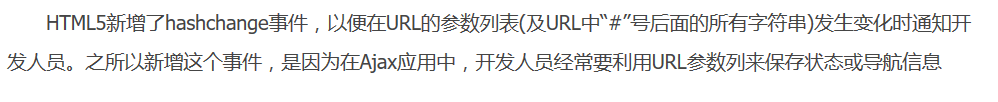
* 1. replace 位置改变但不生成记录



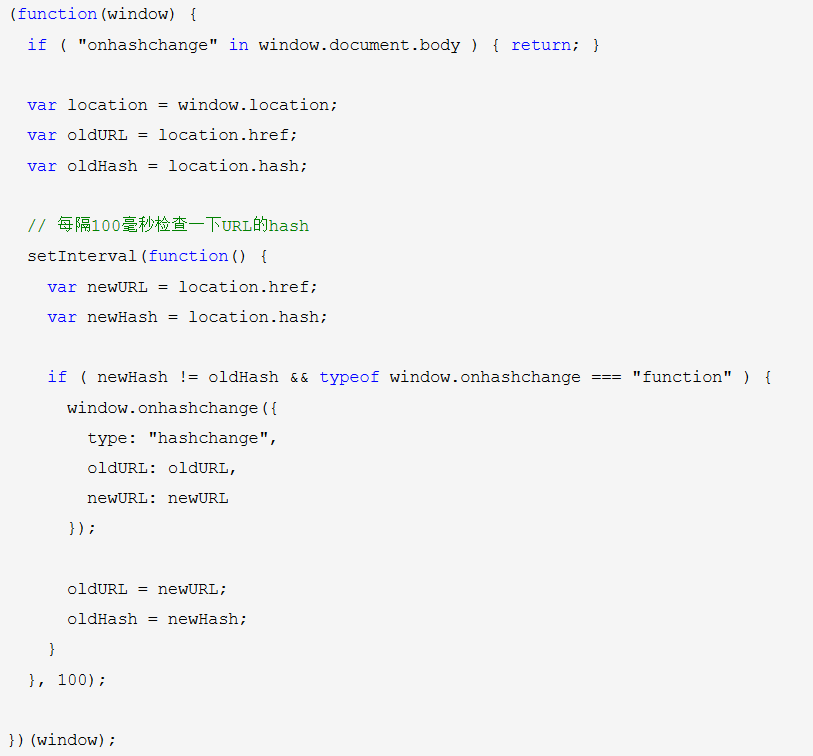
* 1. reload重新加载页面 reload()之后的代码可能不会执行



* 1. hasChange



兼容性写法



1. calc()

.col-md-1{width: calc(8.333% - 20px);}

符号左右必须有空格