**Javascript学习**

js的引用

**-**行内式的引用js



-内嵌式的引用js

<script>不需要行为，打开页面就可以执行</script>

-外链式的引用js最优

<script src=”./test.js”> </script>

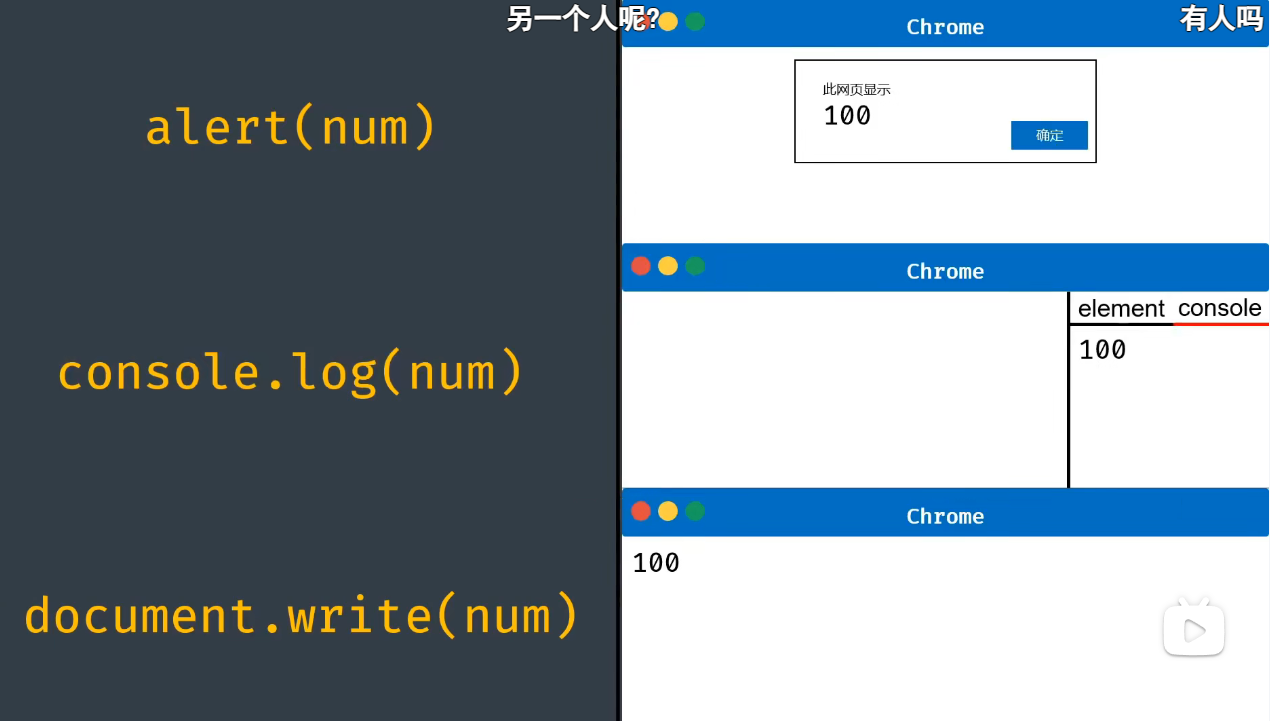
Js的变量

不能以数字开头，可以下划线开头，区分大小写，不可用关键字取名

//单行注释

/\*多行注释\*/

浏览器的三种输出



Js的基本数据类型

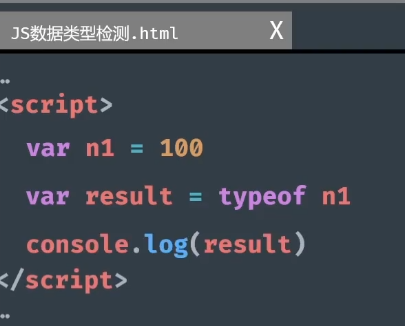
字符串类型 单双引号配套就行

布尔类型 var a = true；/false

**空类型 var a = null;表示有值，但值是空值；**

**var a = undefined; 表示没有值**

typeof返回数值类型



Null返回object

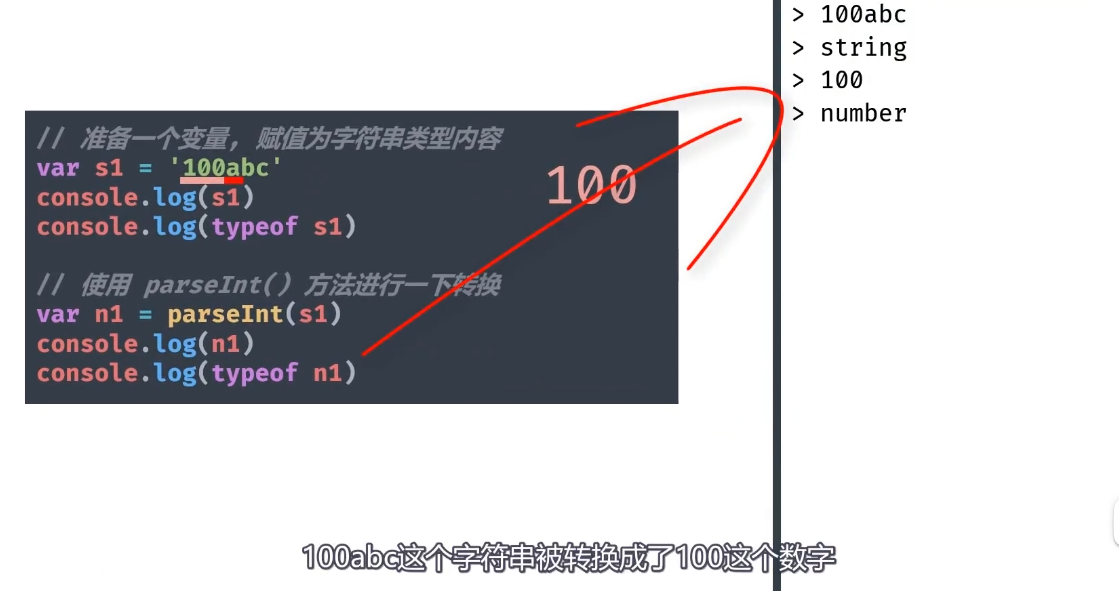
强转换数值类型

**转数值**

Number()改为数值类型 //要是不能正确转换，则显示NaN，如字母用这个转换

ParseInt逐位查看

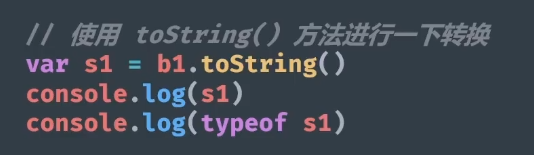
Parsefloat同上，不过可以查看到小数



**转字符串**

String（要转换的内容）

要转换的内容.toString()



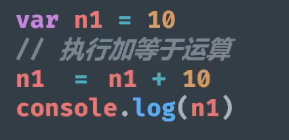
**转布尔**

1. NaN，‘’，undefined，null会返回false，其他有值都返回true

**运算符**

+加或者拼接

%取余

+=

==值相等 ===全等于

！=不相等 ！==不全等

&&与 ||或 ！取反（！true）

++的前置后置，后置先参与计算再加一，前置先加一

**If语句**

If – else if -else

Break跳出循环语句，当 break 语句用于循环语句时，会终止执行循环，并执行循环后代码(如果有的话)。

**Js函数**

Function name(形参){

定义了一个方法函数

}

name(实参) 调用了一个方法

**Js作用域**

自己有用自己的，没有用父级，最高到全局。全没有报错

赋值也是这样，但是全没有就直接定义一个值了

**Js数组常用方法**

Push(),追加数到末尾 arr.push(10)

Pop(),删除末尾数组

unshift 头部追加

shift 头部删除

reverse() 数组反转

splice(开始的索引号,多少个,要插入的数据) //可用于删除，也可用于替换

sort() 数组排序//**arr.sort(function (a,b){return a-b})从小到大排序 b-a从大到小；不写方法按索引排序，也就是不排**

join() 数组加东西//arr.join(’-’) 返回结果中，每个数组间隔了 “ -”

concat(其他数组)//数组拼接

slice(开始索引默认0号索引，结束索引) 切片数组

indexOf()//查找参数，有返回第一次出现的索引；没有返回-1

还有foreach,map,filter,every,some之类的呃

**Js常用字符串方法**

索引，转大小写，把首先遇到的字符串替换掉replace，去掉首尾的空格trim，和join类似的split不过是删除一样的符号然后返回数组

**Js常用数字方法**

Math.random() 获取0~1随机数，包含0但不包含1（17位小数）

Math.roud(数字) 对数字四舍五入取整

Math.ceil()/floor() 向上/下取整 **向下取整对于随机数平衡概率比较好0~10.99999**

**如乘以11向下取整来取0-10的随机数，每个数概率都是一样的**

Math.pow(2,5) 2的5次方

Math.sqrt(4) 4的平方根

Math.abs(-18) 18的绝对值

Math.max(3,2,1) 取其中最大值；min不说了

Math.PI 十七位派

**时间**

Var time = new Date()

**BOM**

window.innerWidth()/window.innerHeight取浏览器宽高尺寸

window.alert(‘我是提示框’)

window.confirm(‘返回true false的询问框’)

window.prompt(‘确定后返回输入的，取消返回null’)

window.onload()页面加载完成执行

window.onresize()页面尺寸发生变化执行

window.onscroll()滚动条滚动执行

window.history.back()/forward历史页面后退或前进

卷去的宽高

window.scrollTo(往右移多少,往下移多少)//瞬间定位

**window.scrollTo({left : xx,top : yy,behavior : ‘smooth’ }) 平滑定位**

**定时器**

setInterval(函数,时间毫秒) 间隔定时器，无限执行

**setInterval(function(){},1000)**

setTimeout(函数,时间毫秒) 延时定时器，执行一次

clearInterval(参)关闭定时器 clearTimeout()效果一样

**DOM**

**-元素的获取**

**e.target用于获取该标签元素（e是形参）**

document.getElementById(‘id名称’)

获取文档流中id名对应的一个元素，有则返回，无则null**（这个返回了包含id标签的整个div）**

document.getElement**s**ByClassName(‘元素类名，一部分也行？’) (多个，返回伪数组)

document.getElementsByTagName(‘标签名’)（一样返回伪数组，伪数组没有就是返回空数组）

document.querySelector(‘选择器，类或者标签都行’)（同类只取第一个，没有也返回null）

document.querySelectorAll(‘选择器，类或者标签都行’)（返回带索引的伪数组，记得命名）

**-元素的操作**

文本操作innerText = ‘新内容’

超文本内容innerHTML = ‘<h1>我可以修改原有的标签</h1>’

**-操作元素属性(不用来操作元素类名和样式)**

getAttribute(‘属性名’) 获取

setAttribute(‘属性名’,’属性名’) 修改

remove移除

**-操作元素类名和样式**

className = ‘类名’

style.样式名 = ’样式值’**(有-改成驼峰命名eg：background-color=>backgroundColor)**

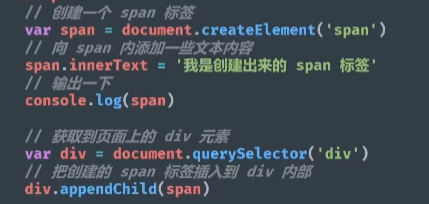
**-节点操作（节点都被var成了变量，节点操作基于DOM）**

创建节点

document.createElement（‘标签名称’）**返回一个创建好的标签元素，页面内却没这个标签**

插入节点

-父节点.appendChild（子节点）让子节点放入父节点内，并排列在最后位置



//这是创建节点通过插入节点来实例化

-父节点.insertBefore(要插入的子节点，哪一个子节点的前面)

**原型**

之所以重要是因为省空间



也是插入节点，先创建span标签节点，再往创建节点插入文本；之后检查div标签（父节点）和p标签，最后用固定格式把本来没有的span插进div中，p标签之前

删除节点

-父节点.removeChild(子节点)

-节点.remove()删除自己

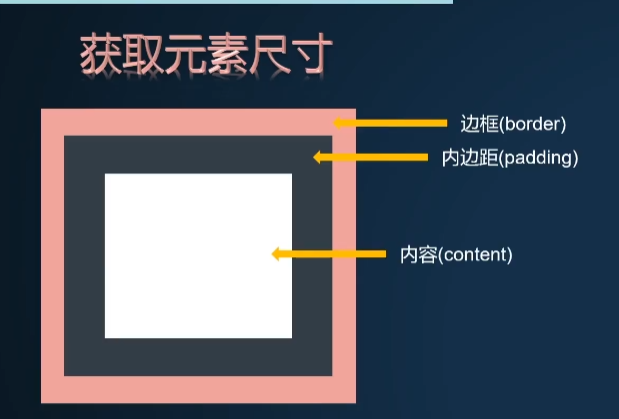
替换节点

父节点.replaceChild(要换上节点，要换下节点) 用创建的节点替换已有节点

克隆节点

节点.cloneNode(是否克隆内部的子节点true/false) 复制一份一模一样的节点

获取元素尺寸 元素.offsetHeight等等，自己找

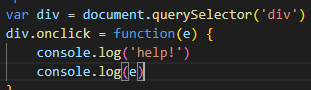


**-事件绑定 on+事件**



上面有误，Contextmenu右键鼠标

**-事件对象**



当在事件方法加入形参，并在控制台输出，会显示详细的信息，如点击位置等等（这个e）

**-鼠标事件**

offsetXY 触发容器如div，容器边缘到点击位置的距离

clientXY 任意大小浏览器左上角到点击位置的距离

pageXY 页面文档流到点击位置距离

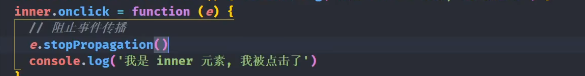
**-键盘事件**

事件对象.keyCode 可得知按下的键

**-事件传播**

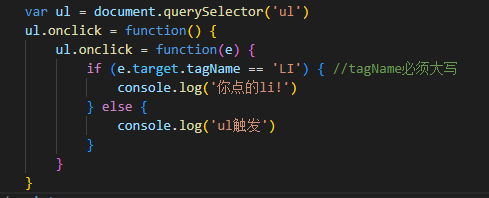
对于一个子父同框的标签，点击子级，会依次触发事件直到父级

为此，需要阻止事件传播 事件对象.stopPropagation()



**-事件委托**

也因为同框冒泡问题，用if语句判断点哪里（善用形参，target是找出元素标签）



**解构**

