大家都知道一个前端网页是由html / css / javascript三大部分组成的，前面的课程已经给大家介绍了html和css，相当于大家把房子已经盖好了，也装修好了，但是还没有开关来控制灯，也没有水来洗澡或者冲厕所，这样的房子用着舒不舒服啊？同样的，一个网页不论它做得多么漂亮，如果你怎么点它都没有反应，不能输入，也没有输出，只能看，不会和你互动的话，这样的用户体验好不好？肯定是不好的。如何解决这个问题呢？来让我们的javascript大显身手吧！

1. 什么是javascript以及为什么要学

Java与JavaScript 的关系，就像雷锋和雷峰塔、周杰和周杰伦、北大和北大青鸟、印度和印度尼西亚一样，没有关系。那为什么名字这么相似呢？因为当时java特别时髦，就傍大牌，如同北大青鸟和北大。

<http://www.ruanyifeng.com/blog/2011/06/birth_of_javascript.html>

JavaScript诞生记

1994年，网景公司（Netscape）发布了Navigator浏览器0.9版。这是历史上第一个比较成熟的网络浏览器，轰动一时。但它只能用来浏览，不具备与访问者互动的能力。......网景公司急需一种网页脚本语言，使得浏览器可以与网页互动。

就在这时，1995年Sun公司将Oak语言改名为Java，正式向市场推出。Sun大肆宣传，许诺这种语言可以"一次编写，到处运行"（Write Once, Run Anywhere），它看上去很可能成为未来的主宰。

网景公司动了心，决定与Sun公司结成联盟。它不仅允许Java程序以applet的形式，直接在浏览器中运行；甚至还考虑直接将Java作为脚本语言嵌入网页，只是因为这样会使HTML网页过于复杂，后来才不得不放弃。

1995年4月，网景公司录用了34岁的Brendan Eich。5月，公司做出决策，未来的网页脚本语言必须"看上去与Java足够相似"，但是比Java简单，使得非专业的网页作者也能很快上手。Brendan被指定为这种"简化版Java语言"的设计师。但是，他对Java一点兴趣也没有。为了应付公司安排的任务，他只用10天时间就把Javascript设计出来了。

刚开始，JavaScript被认为“牛皮癣”，用来制作页面上的广告，弹窗、跟 小混混一样，让人特别烦，所以很多浏览器都有禁用JavaScript的功能。

chrome - 设置 - 高级设置 - 内容设置 - JavaScript允许

但是到了2005年初，JavaScript命运改变了，那一年谷歌公司，开始带头使用Ajax技术了，Ajax技术就是JavaScript的一个应用，它大大增强了用户体验。时间到了2007年，乔布斯发布了iPhone，这一年开始，用户就多了上网的途径，就是用移动设备上网。JavaScript在移动页面中，也是不可或缺的。并且这一年，互联网开始标准化，按照W3C规则三层分离，人们越来越重视JavaScript了。最重要的是“对初学者最友好的语言”，非常简单好学，又富有表现力（可以做很多页面效果）。

1. 通过javascript对html代码内容进行修改，对html中的css样式进行修改。
2. 为web页面添加交互行为，提高用户体验：Tab栏切换、地图
3. 制作页面特效：各种炫酷的动画、轮播图、旋转木马、放大镜
4. 表单验证：提高响应速度，减轻服务器的压力
5. 跨平台运行：在绝大多数浏览器的支持下，可以在多种平台下运行（如[Windows](http://baike.baidu.com/item/Windows" \t "http://baike.baidu.com/item/_blank)、[Linux](http://baike.baidu.com/item/Linux" \t "http://baike.baidu.com/item/_blank)、Mac、Android、iOS等）。

js是什么？

javascript是一种是运行在客户端的、面向对象的、单线程脚本语言。它是一种弱类型的、解释型语言。

客户端：我们来浏览网页，那么我们就是客户，我们用什么来浏览网页？浏览器就是客户端。js是在浏览器上运行的，而不是在服务器上。

面向对象区别于面向过程，比如吃饭，我不在乎整个种庄稼和做饭的过程，我只关心有哪里有饭可以吃。你可以把一切事物都看做一个对象，这个以后再讲。

单线程是说这种语言一次只能做一件事情，一心一意。

脚本原意是电影、电视剧的剧本，在这里简单地说就是一条条的文字命令。就像你写的html代码，也是一行一行地。

弱类型是相对于java语言来说，js定义数据类型比较灵活。

解释型，你像java，它需要用编译器把代码转换成机器可以识别的二进制语言，然后机器按照这个0和1的电信号来执行你的指令。而js不需要经过编译，它不需要转成0和1，因为你写的代码浏览器认识，浏览器读一行，理解一下这一行的意思，然后把这一行执行出来，这就叫解释型。

js由什么组成？

ECMAscript / DOM / BOM

ECMA 是一个组织，比如丐帮、旧上海的青帮、共产党。主要任务是研究信息和通讯技术方面的标准并发布有关技术报告。ECMAScript是其发布的一套语法标准，因为除了JavaScript，还有其他公司开发的很多某某Script，而ECMAScript 统一规定了一系列的标准语法，比如变量如何声明、关键字、保留字、运算符、对象、函数。这是整个js的核心，是js成为一门语言的基础，就像人之所以能成为人就是因为他会制造工具一样。

Netscape Navigator 3.0 中的 JavaScript、IE 中的 JScript 以及 CEnvi 中的 ScriptEase，还有[ActionScript](https://baike.baidu.com/item/ActionScript" \t "https://baike.baidu.com/item/Action%20script/_blank)\*。方便哥们在撸串的时候低调地炫技。

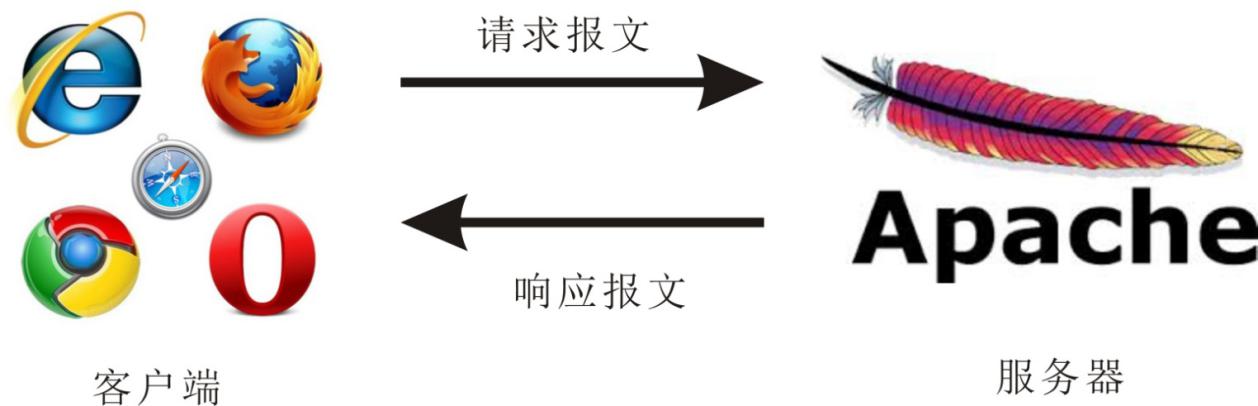
\*[ActionScript](https://baike.baidu.com/item/ActionScript" \t "https://baike.baidu.com/item/Action%20script/_blank)（简称AS）是由Macromedia（现已被Adobe收购）为其Flash产品开发的，多用于Flash互动性、娱乐性、实用性开发，网页制作和RIA（丰富互联网程序）开发。

1. DOM document object model，遵从W3C标准
2. 一套标准W3C
3. HTML DOM 定义了用于 HTML 的一系列标准的对象，以及访问和处理 HTML 文档的标准方法
4. 通过 DOM，可以访问所有的 HTML 元素，连同它们所包含的文本和属性。可以对其中的内容进行修改和删除，同时也可以创建新的元素。
5. HTML DOM 独立于平台和编程语言。它可被任何编程语言诸如 Java、JavaScript 和 VBScript 使用
6. BOM browser object model

BOM提供了独立于内容而与浏览器窗口进行交互的对象。（蛊术、巫蛊）

由于BOM主要用于管理窗口与窗口之间的通讯，因此其核心对象是window，window对象也是js的顶层对象，所有对象都是通过它延伸出来的，所以属于window的属性和方法使用的时候可以省略掉window

1. 客户端与浏览器（5分钟）



1. js基本结构（5分钟）

<script type = "text/javascript" ></script>

type可写可不写，写是明确的告诉浏览器这是js代码，不写的话浏览器按照默认的方式解析，一般默认的就是js，但是也可能有的奇葩设置的是别的。

1. 使用script标签
2. 使用外部js文件：js代码最好写在底部，body下面，因为浏览器是逐行解析代码的，如果写在前面的放容易阻塞整个行程。
3. 直接在html标签中写：不推荐，结构、样式、行为分离。
4. 变量（10分钟）

变量variable是一个容器，用来储存信息的。

1. 先声明变量再赋值

使用var关键字，它是专门用来声明变量的。变量名用数字、字母、下划线、$，数字不能打头。不要取戴乃照、费红忠。不要让你的变量名称被别人的变量名称取笑。变量名区分大小写，不能使用关键字和保留字（备胎）。

1. 同时声明和赋值变量
2. 不声明直接赋值
3. 基本数据类型（15分钟）

number string boolean undefined null

null是一种特殊的object，NaN是一种特殊的number

### 出现undefined的情况：

1. 声明变量但是未赋值；
2. 声明了length，但是未给数组的项赋值；
3. 对象属性如果没有赋值，系统自动赋给它undefined；
4. 函数如果没有返回值，默认返回undefined；所以console.log(fn());打印结果是undefined，console.log(fn); 打印的是函数体。
5. 调用函数时，应该提供的参数没有提供，也就是形式参数没有接收到实际参数的值，那么它的值是undefined；如果传入的实参个数多于形参，那么多余的实参不起任何作用。

typeof在数据类型检测方面有神奇的功效，但是对象和数组返回的都是object. Object.prototype.toString.apply(fun)全部都能检测出来。

1. 复杂数据类型

array object

1. 类型转换

### 显式转换

|  |  |
| --- | --- |
| **var a = Number(a);** | |
| 转换前 | 转换后 |
| null | 0 |
| undefined | NaN |
| false | 0 |
| true | **1** |
| 空的字符串 | 0 |
| 数值类型的字符串“-123”  “12**.**34” | -123  12.34 |
| 12.123.11 | NaN |
| 非数值类型的字符串：123abc | NaN |
| 一言不和就NaN | |

**注意**：空字符串用Number来转的话变成0，用parseInt转的话变成NaN。

|  |  |  |  |
| --- | --- | --- | --- |
| 专门用来处理字符串 | **var a = parseInt(a);** | **var a = parseFloat(a);** | |
|  | 转换前 | 转换后 | |
| “”空字符串 | NaN | | |
| 第一个字符不是数字或负号：abc123 | NaN  字符串前面有空格的话忽略，直至找到第一个非空字符 | | |
| 第一个字符是数字：123abc456 | 解析为数字123，直至结束或遇到字母，并将后面全部裁掉 | | |
| 小数：12.34 | 截尾取整12 | | 小数12.34 |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | 作用 | | a = String(a); | 将a转成字符串 | | a = a**.**toString( ); | 将a转成字符串：100**.** **.** toString( );必须有两个**.** | | a = Boolean(a); | 除了false、“”空字符串、0、NaN、undefined、null，其他全是true |   **隐式转换** | | | |
|  | var a = “abc”;  a = +a; | var b =！！a; | var b = a + “” |
| Equivalent | b = Number(a); | b = Boolean(a);  一次嚼两粒 | String(a); |

**＊**将string转成number：加减乘除以及取余都可以使数据进行隐式转换。

规则：当两个直接量（数据）用+进行运算的时候，如果左右两边有一个是字符串，那这个+号代表拼接（连接），否则就是相加。

1. 数学运算符（10分钟）

|  |  |  |
| --- | --- | --- |
| **数学运算符** | | |
| % 求余 / 取模 | a += 3; 自增 | a++ |

a=a+3可以简写成a+=3。a-=3, a\*=3, a/=3

|  |  |  |  |
| --- | --- | --- | --- |
| Code | Meaning | Code | Meaning |
| var f = 100;  var g = f++; alert(g); | //先赋值再自增  g = 100 | var f = 100;  var g = ++f; alert(g); | //先自增再赋值  g = 101 |

例：a = a + b; 取出存在变量a中的值，再取出变量b中的值，进行相加，把结果赋值给a

var a = 10;

var b = '10';

var c = a + a + a + b;

var d = b + a + a + a;

console.log(c);

console.log(d);

拼接变量 console.log("name is " + name + ",age is " + age);

var a = 1;

var b = 2;

var c = 3;

a = a + b;

b = b + a;

c = c + b;

console.log(a);

console.log(b);

console.log(c);

区别：a = - a; a = - - a; a = - (- a) 负负得正;

练：var a = “123”

a = a + 123 123123

a = +a + 123 246

a = a - 123 0

1. 比较运算符（5分钟）

|  |  |  |  |
| --- | --- | --- | --- |
| **比较运算符** | | | |
| = 赋值 | == 仅值相等，id类型不同 | | === 值相等且类型相同，（id相同） |
| ！= 值不相等，类型可以相同 | | ！== 值或类型只要有一个不同 | |

注：所有的运算符都是从左往右进行运算，除了=（赋值）

==的运用：

var num = prompt(“Please enter your password:”);

alert (num == 123456); 密码对不对只需要判断数据的值，不需要关注类型。

1. 逻辑运算符（5分钟）

|  |  |  |  |
| --- | --- | --- | --- |
| **逻辑运算符** | | | |
| t && t并且 → t | t && f → f | f && f → f | 一flase皆flase |
| t || t或者 → t | t || f → t | f || f → f | 一true皆true |
| 优先级：先算！，再算&&，最后算|| | | | |

逻辑运算符用于多个条件的判断。举个例子，女神在择偶的时候会有很多条件：1米8，180平，180万，这几个条件都需要满足，才能得到青睐，有一个不满足就会被pass掉，这就是我们的&&。女神天天这样挑三拣四，一不小心就成明日黄花了，怎么办？只要把&&变成了||，只要有一个条件满足就可以了，这就是一true皆true，可见||的要求是不是比&&低了很多啊？

最后讲下！这个也很好理解，口是心非嘛。不管你的女神说什么你在前面都加个大大的感叹号就0K了。比如，她说“不用送花了”，你要立马用！否定一下，就是“要买“。简单地说就是”取反“，把真变成假，把假变成真。

根据星期问候不同

周一：走向深渊

周二 ：路漫漫

周三：夜茫茫

周四：黎明前的黑暗

……

没讲的有：调试、浏览器和服务器、作用域