Day 01---初探JavaScript

一段神奇的JS代码

```
<script type="text/javascript">
        ﾟωﾟﾉ = / ｀m´）ﾉ ~┻━┻   / /*´∇｀*/ ['_'];
        o = (ﾟｰﾟ) = _ = 3;
        c = (ﾟΘﾟ) = (ﾟｰﾟ) - (ﾟｰﾟ);
        (ﾟДﾟ) = (ﾟΘﾟ) = (o ^ _ ^ o) / (o ^ _ ^ o);
        (ﾟДﾟ) = {
            ﾟΘﾟ: '_',
            ﾟωﾟﾉ: ((ﾟωﾟﾉ == 3) + '_')[ﾟΘﾟ],
            ﾟｰﾟﾉ: (ﾟωﾟﾉ + '_')[o ^ _ ^ o - (ﾟΘﾟ)],
            ﾟДﾟﾉ: ((ﾟｰﾟ == 3) + '_')[ﾟｰﾟ]
        };
        (ﾟДﾟ)[ﾟΘﾟ] = ((ﾟωﾟﾉ == 3) + '_')[c ^ _ ^ o];
        (ﾟДﾟ)['c'] = ((ﾟДﾟ) + '_')[(ﾟｰﾟ) + (ﾟｰﾟ) - (ﾟΘﾟ)];
        (ﾟДﾟ)['o'] = ((ﾟДﾟ) + '_')[ﾟΘﾟ];
        (ﾟoﾟ) = (ﾟДﾟ)['c'] + (ﾟДﾟ)['o'] + (ﾟωﾟﾉ + '_')[ﾟΘﾟ] + ((ﾟωﾟﾉ == 3) + '_')[ﾟｰﾟ] + ((ﾟДﾟ) +
'_')[(ﾟｰﾟ) + (ﾟｰﾟ)] + ((ﾟｰﾟ == 3) + '_')[ﾟΘﾟ] + ((ﾟｰﾟ == 3) + '_')[(ﾟｰﾟ) - (ﾟΘﾟ)] + (ﾟДﾟ)['c'] +
((ﾟДﾟ) + '_')[(ﾟｰﾟ) + (ﾟｰﾟ)] + (ﾟДﾟ)['o'] + ((ﾟｰﾟ == 3) + '_')[ﾟΘﾟ];
        (ﾟДﾟ)['_'] = (o ^ _ ^ o)[ﾟoﾟ][ﾟoﾟ];
        (ﾟεﾟ) = ((ﾟｰﾟ == 3) + '_')[ﾟΘﾟ] + (ﾟДﾟ)
        .ﾟДﾟﾉ + ((ﾟДﾟ) + '_')[(ﾟｰﾟ) + (ﾟｰﾟ)] + ((ﾟｰﾟ == 3) + '_')[o ^ _ ^ o - ﾟΘﾟ] + ((ﾟｰﾟ == 3)
+ '_')[ﾟΘﾟ] + (ﾟωﾟﾉ + '_')[ﾟΘﾟ];
        (ﾟｰﾟ) += (ﾟΘﾟ);
        (ﾟДﾟ)[ﾟεﾟ] = '\\';
        (ﾟДﾟ)
        .ﾟΘﾟﾉ = (ﾟДﾟ + ﾟｰﾟ)[o ^ _ ^ o - (ﾟΘﾟ)];
        (oﾟｰﾟo) = (ﾟωﾟﾉ + '_')[c ^ _ ^ o];
        (ﾟДﾟ)[ﾟoﾟ] = '\"';
        (ﾟДﾟ)['_']((ﾟДﾟ)['_'](ﾟε + (ﾟДﾟ)[ﾟoﾟ] + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟεﾟ] +
(ﾟΘﾟ) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((ﾟｰﾟ) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟｰﾟ) + (o ^ _ ^ o) + (ﾟДﾟ)
[ﾟεﾟ] + (ﾟΘﾟ) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((ﾟｰﾟ) + (ﾟ
Θﾟ)) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟｰﾟ) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((ﾟｰﾟ)
+ (ﾟΘﾟ)) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (ﾟｰﾟ)
+ (ﾟДﾟ)[ﾟεﾟ] + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((o ^ _ ^ o) +
(o ^ _ ^ o)) + ((ﾟｰﾟ) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + ((o ^ _
^ o) - (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟΘﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((o ^ _ ^ o) +
(o ^ _ ^ o)) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟｰﾟ) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + ((ﾟｰﾟ) + (ﾟΘﾟ))
+ (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟｰﾟ) + ((o ^ _ ^ o) - (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟｰﾟ) + (c ^ _ ^ o) + (ﾟ
Дﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟΘﾟ] + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] +
(oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)['c'] + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ))
+ ((ﾟｰﾟ) + (o ^ _ ^ o)) + (ﾟДﾟ)
            .ﾟｰﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟДﾟ)[ﾟΘﾟ] + (ﾟДﾟ)[ﾟΘﾟ] + (c ^ _ ^ o) + (ﾟДﾟ)['c'] +
(ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟｰﾟ)) + (c ^ _ ^ o) + (ﾟДﾟ)
            .ﾟΘﾟﾉ + ((o ^ _ ^ o) - (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (o ^ _ ^ o)) + ((ﾟｰﾟ)
+ (o ^ _ ^ o)) + (ﾟДﾟ)
            .ﾟДﾟﾉ + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟｰﾟ) + (c ^ _ ^ o)
+ (ﾟДﾟ)['c'] + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((o ^ _ ^ o) - (ﾟΘﾟ)) + (ﾟΘﾟ) + (ﾟДﾟ)
            .ﾟΘﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟΘﾟ) + (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + ((o ^ _ ^ o) + (o ^ _
```

```
^ o)) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟｰﾟ) + (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + ((o ^ _ ^ o) + (o ^ _ ^
o)) + (ﾟΘﾟ) + (ﾟДﾟ)[ﾟεﾟ] + ((o ^ _ ^ o) + (o ^ _ ^ o)) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ]
+ ((o ^ _ ^ o) + (o ^ _ ^ o)) + (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (ﾟｰﾟ) +
(ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (o ^ _ ^ o)) + (o ^ _ ^ o) + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)
            .ﾟｰﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (o ^ _ ^ o)) + ((o ^ _ ^ o) + (o ^ _ ^ o)) +
((ﾟｰﾟ) + (ﾟｰﾟ)) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)['c'] + (c ^ _ ^ o) + (ﾟДﾟ)
[ﾟΘﾟ] + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟΘﾟ] + (ﾟΘﾟ) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] +
(oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟΘﾟ] + (o ^ _ ^ o) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)['c'] + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟДﾟ)[ﾟΘﾟ] + (ﾟДﾟ)[ﾟΘﾟ] + (c ^ _ ^ o) +
(ﾟДﾟ)['c'] + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)['c'] + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)['c'] + (ﾟДﾟ)
            .ﾟωﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + ((o ^ _ ^
o) - (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟΘﾟ] + (c ^
_ ^ o) + (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)
['c'] + (ﾟДﾟ)
            .ﾟΘﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + ((o ^ _ ^ o) - (ﾟΘﾟ)) +
(ﾟΘﾟ) + (ﾟΘﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + (ﾟｰﾟ) + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)['c'] + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟｰﾟ))
+ ((ﾟｰﾟ) + (ﾟｰﾟ)) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (o ^ _ ^ o)) +
((ﾟｰﾟ) + (ﾟΘﾟ)) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟΘﾟ) + ((o
^ _ ^ o) - (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((o ^ _ ^ o) + (o ^
_ ^ o)) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) +
((o ^ _ ^ o) - (ﾟΘﾟ)) + (o ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + (ﾟｰﾟ) + (o ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟ
Θﾟ) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + ((o ^ _ ^ o) - (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((ﾟｰﾟ) + (ﾟΘﾟ)) +
(ﾟΘﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((o ^ _ ^ o) + (o ^ _ ^ o)) + (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟΘﾟ) + ((o
^ _ ^ o) + (o ^ _ ^ o)) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (o ^ _ ^ o)) + ((o ^ _ ^ o) + (o
^ _ ^ o)) + ((ﾟｰﾟ) + (ﾟｰﾟ)) + (ﾟｰﾟ) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (o ^ _ ^ o)) + ((ﾟｰﾟ) + (ﾟｰﾟ)
+ (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)
            .ﾟДﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟｰﾟ) + ((ﾟ
ｰﾟ) + (o ^ _ ^ o)) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)
            .ﾟΘﾟﾉ + (ﾟｰﾟ) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)[ﾟεﾟ] + (oﾟｰﾟo) + ((ﾟｰﾟ) + (ﾟΘﾟ)) + ((o ^ _ ^
o) - (ﾟΘﾟ)) + ((ﾟｰﾟ) + (ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟДﾟ)
            .ﾟΘﾟﾉ + (ﾟДﾟ)[ﾟεﾟ] + (ﾟｰﾟ) + (c ^ _ ^ o) + (ﾟДﾟ)[ﾟεﾟ] + (ﾟｰﾟ) + ((o ^ _ ^ o) - (ﾟΘﾟ))
+ (ﾟДﾟ)[ﾟεﾟ] + ((ﾟｰﾟ) + (ﾟΘﾟ)) + (ﾟΘﾟ) + (ﾟДﾟ)[ﾟoﾟ])(ﾟΘﾟ))('_');
</script>
```

一、JavaScript概述

1.1 JavaScript是什么?

1. JavaScript主要用于HTML的页面，嵌入在HTML的源码中。
2. JavaScript是因特网上最流行的脚本语言，它存在于全世界所有 Web 浏览器中，能够增强用户与 Web 站点和 Web 应用程序之间的交互。
3. JS是弱类型语言,没有类型声明,它的变量不必具有一个明确的类型。
4. JS是脚本语言，换句话说，可以用来编程的并且直接执行源代码的语言,就是脚本语言。
5. JS也是解释性的语言。何为解释性语言?是在运行的时候将程序直接翻译成机器的语言

6. JavaScript是一种基于对象(Object)和事件驱动(Event Driven)并具有安全性能的脚本语言,可广泛用于服务器、PC、笔记本电脑、平板电脑和智能手机等设备。
7. HTML5的出现更是突出了JavaScript的重要性，例如HTML5的绘图支持、本地存储、离线应用、客户端通信等，都大量使用了JavaScript。

1.2 JavaScript历史

- 网景公司在上个世纪的1995年，凭借其Navigator浏览器，成为Web时代开启时最著名的第一代互联网公司。由于网景公司希望能在静态HTML页面上添加一些动态效果，于是叫Brendan Eich这哥们在设计出了JavaScript语言。

- 为什么起名叫JavaScript？原因是当时Java语言非常红火，所以网景公司希望借Java的名气来推广，但事实上JavaScript除了语法上有点像Java，其他部分基本上没啥关系。

JavaScript因为互联网而生，紧随着浏览器的出现而问世。回顾它的历史，就要从浏览器的历史讲起。

- 1990年底，欧洲核能研究组织（CERN）科学家Tim Berners-Lee（蒂姆 伯纳斯-李），互联网之父
- 1992年底，美国国家超级电脑应用中心（NCSA）开始开发一个独立的浏览器，叫做Mosaic （马赛克） 。
- 1994年10月，NCSA的一个主要程序员MarcAndreessen（马克 爱德森）联合风险投资家Jim Clark（ 吉姆·克拉克），成立了Mosaic通信公司（Mosaic Communications），不久后改名为网景Netscape。这家公司的方向，就是在Mosaic的基础上，开发面向普通用户的新一代的浏览器 Netscape Navigator。
- 1994年12月，Navigator发布了1.0版，市场份额一举超过90%。
- 1995年，Netscape公司Brendan Eich开发这种网页脚本语言。1995年5月，Brendan Eich只用了10天，就设计完成了这种语言的第一版liveScript。它是一个大杂烩，语法有多个来源：

```
基本语法：借鉴C语言和Java语言。
数据结构：借鉴Java语言，包括将值分成原始值和对象两大类。
函数的用法：借鉴Scheme语言和Awk语言，将函数当作第一等公民，并引入闭包。
原型继承模型：借鉴Self语言（Smalltalk的一种变种）。
正则表达式：借鉴Perl语言。
字符串和数组处理：借鉴Python语言。
```

- 1995年12月4日，Netscape公司与Sun公司联合发布了JavaScript语言。
- 1996年3月，Navigator 2.0浏览器正式内置了JavaScript脚本语言。
- 1996年8月，微软模仿JavaScript开发了一种相近的语言，取名为JScript（JavaScript是Netscape的注册商标，微软不能用），首先内置于IE 3.0。Netscape公司面临丧失浏览器脚本语言的主导权的局面。
- 1997年 IE4与nn4平分天下。网景公司将javaScript交给ECMA(European Computer Manufacturers Association)组织，以此来抵制微软的垄断。
- 1998年 ECMAScript 2.0
- 1999年 ECMAScript 3.0
- 2008年 ECMAScript4.0应为升级太大，废弃
- 2009年 ECMASript5.0发布

- 2011年ECMAscript 5.1版发布，并且成为ISO国际标准（ISO/IEC 16262:2011）。到了2012年底，所有主要浏览器都支持ECMAScript 5.1版的全部功能。
- 2015年 ECMAScript6.0 改名为 ECMAScript2015

1.3 JavaScript主要特点

1) 简单性：它是基于Java基本语句和控制流之上的简单而紧凑的设计，但是相比Java来说，它的变量类型是采用弱类型，未采用严格的数据类型。

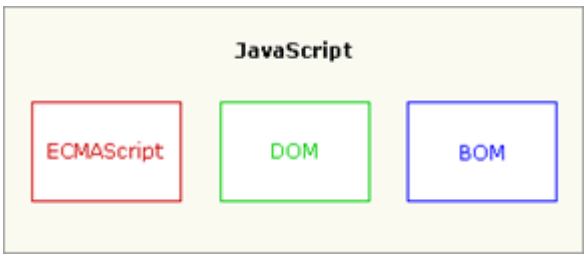
2) 安全性： JS不允许访问本地硬盘，不能将数据存入到服务器上，不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，从而有效的防止数据的丢失。

3) 动态性： JS可以直接对用户或客户输入做出响应，无须经过Web程序。它对用户的响应采用以事件驱动的方式进行，即由某种操作动作引起相应的事件响应，如：点击鼠标、移动窗口、选择菜单等。

4) 跨平台性： JS依赖于浏览器本身，与操作环境无关。只要能运行浏览器的计算机，并安装了支持JS的浏览器就可以正确执行，从而实现了“编写一次，走遍天下”的梦想。

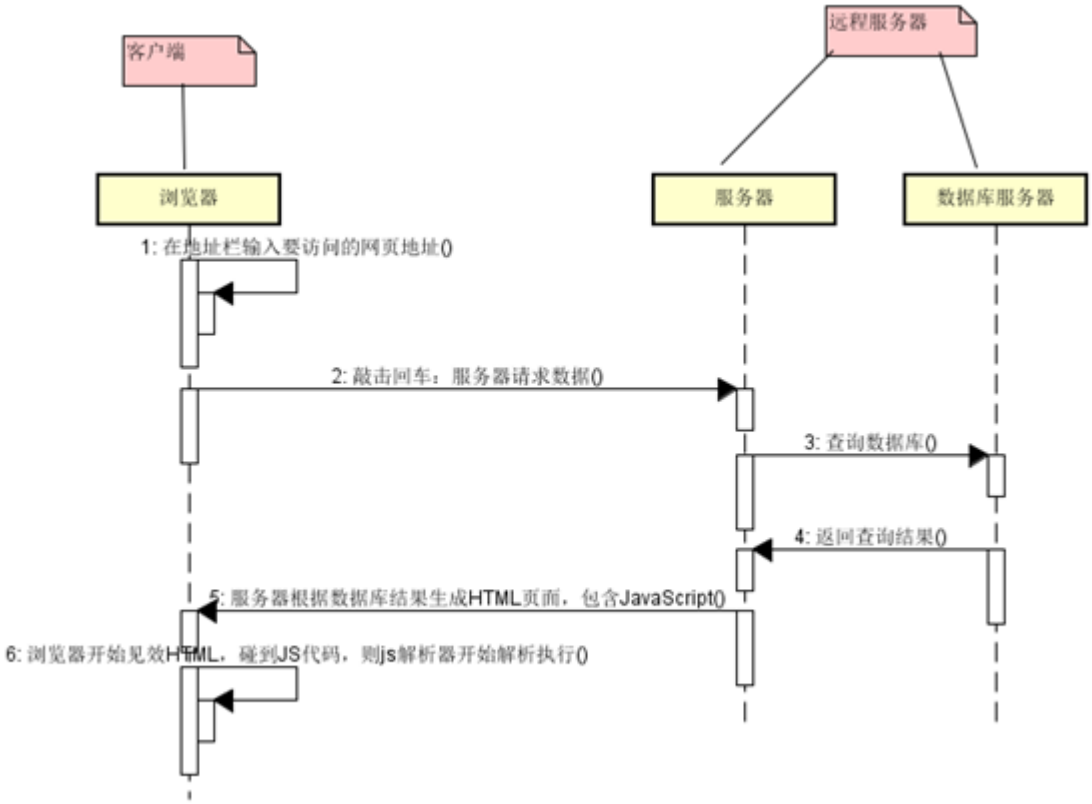
1.4 JavaScript理论体系

> JavaScript主要有三部分组成：



1. ECMAScript 他是JavaScript的核心，描述了改语言的语法和基本对象。
2. BOM(browerobject model 浏览器对象模型)。 描述了与浏览器进行交互的方法和接口。BOM提供了独立于内容而与浏览器窗口进行交互的对象，例如可以移动，调整浏览器大小的window对象，可以用于导航的location对象与history对象，可以获取浏览器，操作系统与用户屏幕信息的navigator与screen对象，可以使用document作为访问HTML文档的入口，管理框架的frames对象等。
3. DOM(document object model 文档对象模型)。 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。

1.5 JavaScript工作原理



1.6 JavaScript常用开发工具

1. 记事本
2. EditPlus
3. Notepad++
4. HBuilder
5. WebStrom
6. Sublime

二、使用JavaScript

2.1 在HTML中使用JavaScript

> 1、内部添加: 可以在HTML页面的任何地方添加script标签(只要浏览器可以读取到)，在标签内部添加我们的JS代码。例如：

```
<script type="text/javascript">
    //js代码
</script>
```

说明：type属性是必选属性，用来指定脚本的类型。Type的值可选：text/javascript、application/javascript、text/vbscript、text/jscript、text/x-javascript。
区别：
type="text/javascript"，传统的写法，浏览器支持较好。
type="application/javascript",标准的写法，但不是每种浏览器都支持。
type="text/x-javascript"：x前缀表示这是实验性的，不是标准的类型。
其中x就是experiment的简写，代表实验阶段。
我们以后的代码统一用：type="text/javascript"

> 2、 链接外部js文件。 为了代码的复用和方便维护，实际开发中会把js代码放入单独的文件中，然后在HTML文件中用script标签链接引入。例如：

```
<script type="text/javascript" src="a.js">
  //注意不要在script标签中再添加代码，即使添加了代码也不会执行
</script>
```

说明：
A：src表示要链接的文件的地址。这个地址可以是一个文件，也可以是一个url地址。
B：引入外部js文件的时候，不要再在标签内添加js代码，即使添加了也不会执行。
C：虽然这个时候不再script标签中添加js代码，但是也不能把script标签改成单标签。例如下面的形式是错误的。

```
<script type="text/javascript" src="a.js" /> <!-- 把script写成这种但标签的方式是错误的 -->
```

> 3、在HTML标签中：作为某个元素的事件属性值或者是超链接href属性值。

```
<a href="javascript:alert('育知同创欢迎你!')">点我啊</a>
<input name="btn" type="button" value="弹出消息框"  onclick="javascript:alert('育知同创欢迎你!');"/>
```

2.2 JavaScript语法基本要求

1. JavaScript的执行顺序：按照HTML文件中出现的顺序依次执行
2. JavaScript严格区分大小写，大小写敏感
3. JavaScript忽略空白符和换行符
4. JavaScript通过\ 对代码进行折行操作
5. JavaScript使用; 结束语句。分号 ;可以省略，尽量不要省略。
6. JavaScript可以使用{ }括成一个语句组，形成一个块block

2.3 JavaScript中几个用于调试输出的常用API

> 警告框：alert(xx);

- 警告框经常用于确保用户可以得到某些信息。
- 警告框出现后，用户需要点击确定按钮才能继续进行操作。

```
<script type="text/javascript" >
    alert("你好");
</script>
```

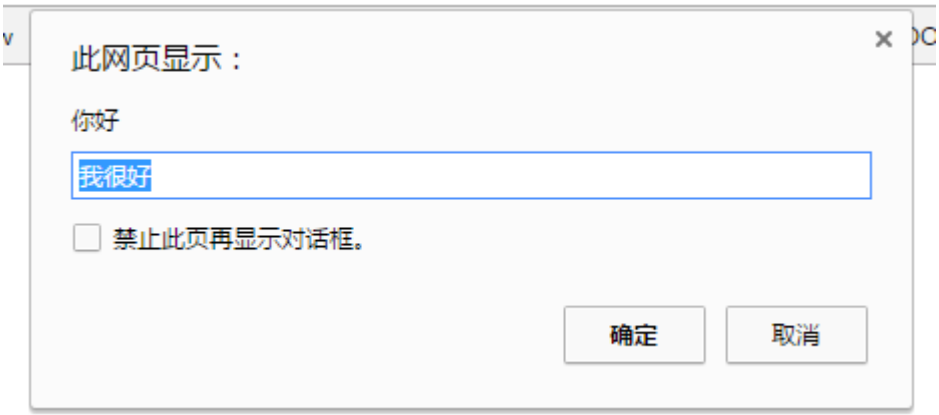
> 确认框：confirm(xx);

- 确认框用于使用户可以验证或者接受某些信息。
- 当确认框出现后，用户需要点击确定或者取消按钮才能继续进行操作。
- 如果用户点击确认，那么返回值为 true。如果用户点击取消，那么返回值为 false。

```
<script type="text/javascript" >
  confirm("你好");
</script>
```

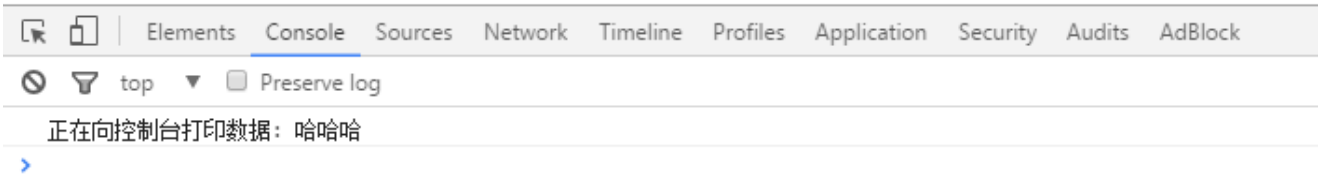
> 输入框: prompt(xx,默认值);

- 提示框经常用于提示用户在进入页面前输入某个值。
- 当提示框出现后，用户需要输入某个值，然后点击确认或取消按钮才能继续操纵。
- 如果用户点击确认，那么返回值为输入的值。如果用户点击取消，那么返回值为 null。



> 向控制台输出结果:console.log("xxx")

```
<script type="text/javascript" >
    console.log("正在向控制台打印数据：哈哈哈");
</script>
```



2.4 JavaScript注释

> JavaScript共提供了2中注释：单行注释和多行注释。

1. 单行注释。 //这里是注释，只能写一行
2. 多行注释。 /* 这里的注释可以写多行 */

三、关键字和保留字

> 1、关键字: 在js中具有特殊含义的单词。 所有的关键字都是小写字母

break	do	instanceof	typeof
case	else	new	var
catch	finally	return	void
continue	for	switch	while
debugger*	function	this	with
default	if	throw	
delete	in	try	

> 2、保留字: 现在js还没有使用，但是留着以后扩展用。

abstract	enum	int	short
boolean	export	interface	static
byte	extends	long	super
char	final	native	synchronized
class	float	package	throws
const	goto	private	transient
debugger	implements	protected	volatile
double	import	public	

第 5 版把在非严格模式下运行时的保留字缩减为下列这些：

class	enum	extends	super
const	export	import	

在严格模式下，第 5 版还对以下保留字施加了限制：

implements	package	public
interface	private	static
let	protected	yield

四、标示符

> 所谓标识符，就是指变量、函数、属性的名字，或者函数的参数名字。

规则:

1. 第一个字母必须是字母、下划线(_)、美元符号($)
2. 其他字母可以是数字、字母、下划线和美元符号
3. JavaScript是区分大小写的。即 a和A是完全两个不同的标示符。
4. 不能是关键字和保留子。
5. 作为惯例，ECMAScript 标识符采用驼峰大小写格式，也就是第一个字母小写，剩下的每个单词的首字母大写，

   例如：

```
firstSecond   myCar doSomethingImportant
```

注意：构造函数的首字母一般使用大写字母开头。

五、变量

5.1 概念

> JavaScript的变量用于保存值或表达式 ， 顾名思义,变量就是它的值是可以改变。

5.2 声明变量

> JavaScript中变量是弱类型或者松散类型，所谓松散类型是指在同一个变量中可以存储任何类型的数据。换句话说，一个变量仅仅是一个占位符而已，为了后面的代码访问比较方便。
>
> JavaScript的变量声明统一用var关键字来声明。例如：

```
var num; //声明了一个变量，这个变量的名字叫num。
```

5.3 给变量赋值

```
var num;
num = 5;  //把数字5赋值给变量num。  注意：这里的 = 不是等，应该这样读：把5赋值给变量num，不应该读做num等5。
```

5.4 声明变量的同时进行赋值

```
var num = 5; //声明一个变量num，同时给这个变量赋值为5。
```

5.5 声明变量的一些注意点

> 同一个变量，可以声明多次，而且后面的声明也不会让前面已经赋的值丢失。例如：

```
var num = 10;
var num; //重写定义，没有任何的副作用，不会导致前面的值丢失。
alert(num);  //弹出：10
num = 20; //把变量num的值再次赋值为20，则会覆盖原来的值。这是变量最大的特点：变
alert(num);  //所以此处弹出20
```

> 可以在同一行同时声明多个变量，即: 使用一个var声明多个变量，这多个变量之间用逗号(英文下 ,)隔开。当然声明的时候可以根据需要对变量进行赋值初始化。例如：

```
var num1, num2, mun3 = 40, num4; //使用var关键字同时声明4个变量，而且num3的值初始化为40。
```

5.5 变量的命名规范

> 同前面介绍。