# 第 1 章　JavaScript 与 HTML 基础

本章主要内容是介绍 JavaScript 和 HTML 基础知识，包括语法、结构、编程思想和调试工具等。JavaScript 语言功能丰富，本章只挑选低代码编程需要的知识讲授。换言之，掌握了本章内容的读者，就能够使用低代码构件进行快速编程了。如果已经掌握 JavaScript 和 HTML 知识，只需要对调试工具、变量命名规范了解一下即可。

## 1.1　JavaScript 概述

JavaScript 简称 JS，是一种解释性语言，即需要被浏览器解释、编译成二进制，然后才能执行，解释和编译过程在浏览器后台执行，用户是看不到的。

### 1.1.1　历史

JavaScript 语言的发明人是布兰登·艾奇。1995 年，时年 34 岁的他在美国网景公司任职时，为网景浏览器开发出 JavaScript。

JavaScript 最开始被定义为一种脚本语言，只是让页面动起来，但是其发展速度超出了预期，一度使得解释其执行的浏览器频繁崩溃。

直到 2001 年微软发布了 IE 6，推出了 DOM 技术，首次实现对 JavaScript 引擎的优化和分离，JavaScript 才开始更加广泛地应用于页面中。

微软一直引领 JavaScript 发展到 2006 年，其间还发明了 Ajax 技术，但 Ajax 第一次广泛使用是在 Google 搜索引擎中——自此 Google 接过了 JavaScript 的接力棒，成为 JavaScript 技术的领航者。2008 年 Google 发布了 Chrome 浏览器，其最大的特点就是针对 JavaScript 进行了引擎优化，运行速度超过了 IE 6 数倍。

2011 年，随着移动设备成熟，在苹果公司牵头下，HTML 5 发布，JavaScript 语言进一步扩容，ECMAScript 5.1 规范发布，JavaScript 开始支持触摸事件等，能够实现语音功能和播放视频等。

此后的十年，JavaScript 一直在不断完善中，有一些人工智能、更多的硬件交互，以及虚拟影像功能正在逐渐添加进来。

### 1.1.2　作用与用法

浏览器刚刚诞生的时候，只能显示用 HTML 语言编写的静态网页，交互非常简单，通过 href 实现单击 a 标签就可跳到相应的地方。JavaScript 诞生后，因为可以处理更多的事

件,网页的互动性大大加强。

JavaScript 语言也可以跟 HTML 语言和 CSS 语言交互使用,可以增强网页的动态效果。

在 HTML 页面中使用 JavaScript 语言编程,需要使用<script>标签包裹起来。例如:

**代码 1-1** 在 HTML 页面中使用<script>标签。

```
1  <html><head></head><body>
2  <script>
3      document.write("世界,你好!");
4      document.write("Hello,world!");
5  </script>
6  </body>
7  </html>
```

上面代码的第 1 行为 HTML 代码的起始符,第 2 行为<script>起始符,第 3、4 行为 JavaScript 代码,第 5 行为</script>结束符,第 6、7 行为 HTML 代码的结束符。

上面的 script 标签写在 body 里面,实际上 JavaScript 代码可以写在任何 HTML 标签里面。

### 1.1.3 调试工具

JavaScript 由浏览器解释和执行,所以浏览器可以作为 JavaScript 调试工具。

现在,市场占有率最大的浏览器是 Google 公司的 Chrome 浏览器。在菜单中选择“更多工具”→“开发者工具”命令,即可打开浏览器的调试工具,如图 1-1 所示。

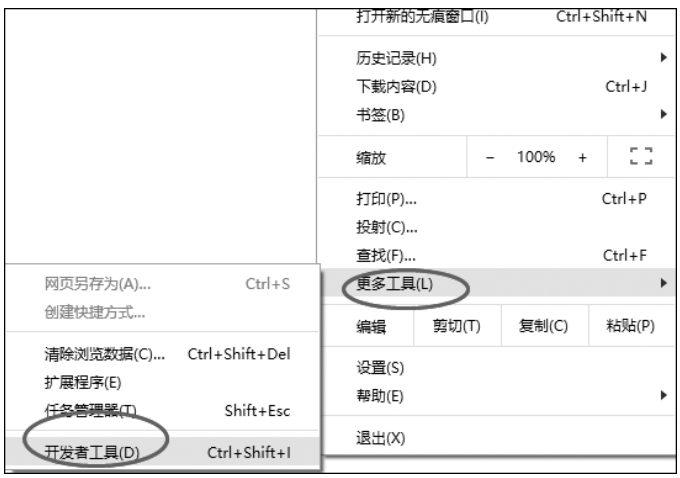


图 1-1 打开 Chrome 浏览器的 JavaScript 调试工具

调试工具中比较常用的调试窗口有两个:一个是 Console,另一个是 Network。

(1) Console 的作用主要有两点:一是打印所有 JavaScript 使用 console.log 输出的调试信息,二是用红字显示错误和异常。

(2) Network 可以显示已访问的网址和参数信息,包括以 get 和 post 方式提交的参数,这个功能非常有用。本书后面几章都会用到查看参数的功能。

除了 Google 浏览器外，微软的 IE、Edge 浏览器的调试工具也非常实用，因为前端开发通常要测试代码的浏览器兼容性，所以即使用 Chrome 调试完毕，也需要用其他浏览器再次调试一遍，直到所有浏览器都能兼容为止。

从笔者的开发经验来看，有时候 Chrome 浏览器不能准确定位错误；而 IE 和 Edge 浏览器可以准确定位错误，按 F12 键即可打开微软这两款浏览器的控制台。与 Chrome 不同的是，一旦关闭控制台，IE 和 Edge 浏览器会自动清空所有调试记录，这也是它们的不足之处。所以，平时调试用 Chrome 浏览器就足够了，当 Chrome 不能定位错误时，可以换用 IE 或者 Edge 辅助查找。

## 1.2　语　　法

JavaScript 语法采用了一些 Java 的语法，并做了适当修改，这也是将其命名为 JavaScript 的原因之一。

### 1.2.1　变量

JavaScript 中的变量是弱类型的，类似于数学中的未知数，可以被赋予任何值，被赋予的值保存在浏览器的内存中，一旦浏览器刷新或者关闭，就会清空所有内存中的值。

所谓弱类型，是指 JavaScript 在定义变量时，不需要区分是整数类型还是字符串类型，都使用关键字 var 来定义变量，这一点与 C 语言、Java 语言不同。相同的是它们都使用等号来为变量赋值，如“var a=1;”。

**注意**：末尾的分号表示一个语句结束，这一点与其他语言相同。另外，定义变量的语句也支持多个变量相继赋值，只需用逗号隔开即可，例如：

```
var a=1, b=2, c=3, d='世界', e='你好',f=true,g=false;
```

在对变量进行命名时，有一些规则需要注意。

(1) 变量名称是区分大小写的，a 和 A 是不同的变量。

(2) 变量名称只能以大小写字母、数组、下画线和美元 $ 符号组成。

(3) 变量名称不能以数字开头。

(4) 字符串变量需要用半角单引号或者双引号包裹。

(5) 布尔变量只有 true 和 false 这两种值。

在给变量命名时，为了便于记忆和阅读，通常要采用有含义的字符串，如英语 school 要比 ssss 容易记忆，也比拼音 xuexiao 容易阅读。老师或者其他书籍上一般会提到这个命名规则，本书的拔高之处在于，为了提高阅读速度，本书介绍的低代码构件支持中文命名。例如，“var name='大熊猫';”这个语句采用中文命名的方式，可以写为“$.名称='大熊猫';”语句，即用中文名称方式来定义变量名称，只需要在汉字前面加上美元符号和点即可，这对于英文不佳的程序员来说绝对是一个福音，对于英语好的程序员来说也节省了编写注释的工作量。

中文编程一直是中国人的梦想，编程原理在 1.4 节会详细介绍，后面章节也会通过构件

案例来提高熟练度。

### 1.2.2 运算符

运算符用于执行程序代码运算,通常用来对一个或者多个变量、常量进行计算。

运算符有很多种类,低代码常用到四种运算符:算术运算符、关系运算符、逻辑运算符和三目运算符。下面进行详细介绍。

(1) 算术运算符是指用来进行加、减、乘、除数学运算的运算符,加、减跟数学中的相同,就是+、-,乘号用*,除号用/。

(2) 关系运算符是指比较数字和字符串大小的运算符,大于号是>,小于号是<,等于号是==,因为一个等号是赋值,大于或等于是>=,小于或等于是<=。

(3) 逻辑运算符是指对布尔变量进行逻辑运算,使用感叹号"!"。

(4) 三目运算符较为复杂,通常用来进行选择性返回值,由"?"和":"组成。例如,"2>1? 2: 1"的意思是:如果 2 大于 1,则返回 2;否则返回 1。同理,"a>b? a: b"的意思是:如果 a 大于 b,则返回 a;否则返回 b。

### 1.2.3 语句

除了给变量赋值语句外,JavaScript 还有多种语句,为了简化编程,本书介绍的低代码编程只用到两种语句,一种是条件语句,另一种是循环语句。条件语句中,这里只介绍最常用的 if else 语句;循环语句中,这里也只介绍最常用的 for 语句。例如下面的条件语句代码:

```
if(1+1==2){var a=2;a++;}else{var b=3;b--;}
```

这条语句的意思是:如果 1+1 等于 2,则定义一个变量 a,让它等于 2,然后 a 加 1;否则定义变量 b,让 b 等于 3,然后 b 减 1。这条语句的代码执行结果显然是 a 等于 3。

可以看出,其规律是:if 后面要跟小括号,小括号里面是一个逻辑判断语句,即数字比大小,除了数字可以比大小外,字符串变量也可以用等号来比较是否相等,若相等则返回 true,否则返回 false。

if 和 else 的执行语句若是超过一句,则需要用左右大括号包裹起来,左右大括号必须要成对出现。

for 循环语句可以这样写:

```
var a=0;for(var i=0;i<100;i++)a+=1;
```

这条语句的意思是:定义变量 a,让其等于 0,然后循环执行 100 次,每次都将 a 加 1,这条语句执行的结果显然是 a 变成了 100。

for 语句后面同样跟一个小括号,小括号里面需要有一个变量 i 来控制循环次数,后面的执行语句若是超过一句,也需要用大括号包裹起来,因为上面的例子中没有超过一句,所以就没有使用。

### 1.2.4 方法

定义 JavaScript 方法的关键字是 function,不需要声明返回类型,这个定义方法的语法与 Java 和 C 语言都不同,相同的是,方法的代码也要用大括号包裹起来。例如,定义一个

hello 方法，就可以写作：

```
function hello(){alert('世界,你好!');}
hello();
```

代码共有两行，第 1 行是定义方法 hello()，hello()方法的功能是：显示一个提示框，提示框上输出“世界，你好!”字样。

第 2 行是调用 hello()方法，调用语句的语法是在方法名称后面紧跟小括号，小括号里面可以写参数，就像在 alert()里面写了一个字符串作为参数。

这是第一次调试，详细步骤如下。

(1) 打开 Chrome 浏览器，输入网址 http://www.chofo.com/demo/pc.htm，如果网站已经打开，则要刷新网页，清空之前的执行记录。

(2) 按照 1.1.3 小节介绍的方法打开“开发者工具”，屏幕是宽屏，工具窗口通常显示在右侧，后面为了减少缩放及截屏更清晰，也会调整到下面。

(3) 复制代码，并粘贴到 Console 控制台中。

(4) 按 Enter 键查看代码执行结果。

如果代码成功执行，执行结果如图 1-2 所示。

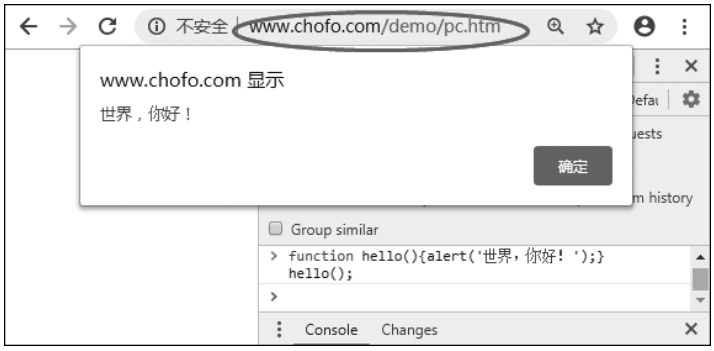


图 1-2　在 Console 控制台中弹出 hello 对话框

alert()是 JavaScript 的内部方法，它可以接受字符串作为参数。我们自定义的方法也可以带参数，这样就可以向方法传值，可以定义一个参数，也可以定义多个参数，多个参数之间由逗号(,)分隔。例如，我们可以改造 hello()方法为多语言的：

```
function hello(lg){if(lg=='中文')alert('世界,你好!');else alert('hello,
world!');}
```

这个 hello()方法的功能是：如果传的字符串是“中文”，则弹出对话框输出的是中文“世界，你好!”，否则输出英文“hello，world!”。

每个方法都可以定义返回值，因为 JavaScript 是弱类型的，所以不需要像 Java 或者 C 那样声明返回的类型，只需要像 Java 或者 C 那样使用 return 语句返回即可。在使用 return 语句时，方法就会停止执行，并返回指定的值。例如：

**代码 1-2**　定义 hello()方法。

```
1   function hello(lg){
```

```
2       if(lg=="中文"){alert("世界,你好!");return true;
3       }else alert("hello,world!");
4       return false;
5   }
6   hello("中文");
```

上面代码的主要功能是定义 hello 方法并接受参数 lg,当 lg 为“中文”时,返回 true 类型,其他一律返回 false 类型。代码详细介绍如下。

第 1 行定义了 hello()方法,hello()方法有一个参数 lg。

第 2 行用 if 语句判断 lg 是否等于“中文”,这里注意 if 后面的语句是用大括号包裹的,所有大括号中的语句都要符合 if 中的判断才可以执行。

第 3 行是 else 语句,else 后面可以跟大括号,也可以不跟,不跟的时候只执行一条语句。

第 4 行是返回语句。

第 6 行调用 hello()方法。

按照前面说的 4 个步骤将代码复制并粘贴到 Console 控制台中,代码执行结果如图 1-3 所示。


图 1-3　在 Console 控制台中执行带参数的 hello()方法

从图 1-3 中可以看出,调用 hello()方法以后,alert()方法显示出了提示框,说明执行了 if 语句,单击提示框中的“确定”按钮,Console 控制台输出了 true,说明执行了 return 语句。所有的代码执行结果均符合预期。

### 1.2.5　思考题 1-1：输出多语言的 Hello world

这一小节我们写一个方法,用来输出多语言的 Hello world,同时学习 JavaScript 非常重要的方法 eval()。eval()的作用是将一个字符串转换为可执行的 JavaScript 变量或者语句。

**代码 1-3**　输出多语言 Hello world 源代码。

```
1   function hello(lg){
2       var a0="世界,你好!",a1="世界、こんにちは!",a2="Hello,world!";
3       for(var i=0;i<3;i++){
```

```
4            console.log(eval('a'+i));
5            if(lg==i)return eval('a'+i);
6        }
7    }
8    hello(2);
```

从上面代码可以看出，代码中 hello()方法的功能是：接受 lg 参数，并根据传递的参数动态输出并返回不同的问候语。方法的实现过程如下。

第 1 行定义了 hello()方法，hello 方法有一个参数 lg。

第 2 行定义了 3 个字符串变量，变量的值是中、日、英三种语言的问候语。

第 3 行开始进行 for 循环判断。

第 4 行使用 eval()方法。eval()方法是 Java 语言和 C 语言中没有的方法，在上面的代码中，eval('a'+i)的意思是：字符 a 加上数字 i 构成了一个新的字符串，经过 eval()转换后，当 i 等于 0 时就变成了变量 a0；当 i 等于 1 时就变成了变量 a1；当 i 等于 2 时就变成了变量 a2。转换完毕向 Console 控制台输出 eval 解释的结果，看看是否跟第 2 行定义的一致。

第 5 行对参数 lg 和循环 i 进行比较，如果相同，则用 return 返回问候语。

第 8 行是调用 hello()方法，参数为 2，即返回 a2 变量的值，也就是英文问候语。

将代码复制并粘贴到 Console 控制台中，结果如图 1-3 所示，为了节省篇幅，截屏时只截取了代码部分，如图 1-4 所示。

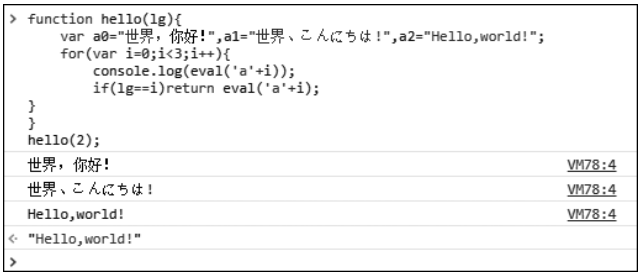


图 1-4　在 Console 控制台中执行带参数含 for 循环的 hello()方法

从图 1-4 中可以看出，调用 hello(2)后的返回值是“Hello，world!”，即表示该方法的代码执行结果均符合预期。

## 1.3　数　　组

数组是一种采用下标的方式组织和索引数据的数据结构。使用数组组织数据的优点是速度快、代码少，方便机器阅读。

### 1.3.1　定义数组

JavaScript 用多种方式定义数组。

(1) 使用 Array：“var a＝new Array();”。

(2) 使用中括号：“var a＝[];”。

建议采用第二种,原因就是代码量少。定义数组时既可以初始化长度,也可以进行初始赋值。

```
var a= ["序列","用户名","性别","电话","地址"];
```

采用 length 属性获得数组长度,如 a.length 是获得数组元素的值。使用中括号加下标,如 a[0]是获得数组 a 的第 0 个元素,数组的下标从 0 开始,这一点与 Java 一样。

### 1.3.2 数组常见操作方法

JavaScript 数组操作方法非常丰富,基本满足了日常的操作需求,如表 1-1 所示。

**表 1-1 数组的操作方法**

| 名 称 | 解 释 |
|---|---|
| join | 将数组连接成一个字符串 |
| sort | 对数组进行排序 |
| push | 在数组末尾增加一个值 |
| pop | 删除数组末尾的值 |
| unshift | 在数组开头增加一个值 |
| shift | 删除数组开头的第一个值 |
| splice | 删除任意数组任意位置的值,并插入某个新值 |
| slice | 获取数组的某个片段 |
| delete | 删除整个数组 |
| concat | 链接两个数组成为一个新的数组 |

这里挑选最常使用的四个方法进行举例说明。

(1) join 使用举例

```
var a= ["序列","用户名","性别","电话","地址"];
console.log(a.join());
```

将代码复制并粘贴到 Console 控制台中,代码执行结果如图 1-5 所示。


图 1-5 数组用 join()方法连接后输出结果

(2) sort 使用举例

```
var a= ["序列","用户名","性别","电话","地址"];
a.sort();
console.log(a.join());
```

将代码复制并粘贴到 Console 控制台中,代码执行结果如图 1-6 所示。

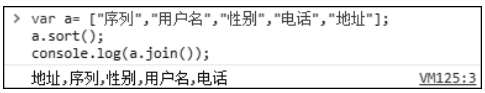


图 1-6　数组用 sort()方法排序后输出结果

(3) push 使用举例

```
var a= ["序列","用户名","性别","电话","地址"];
a.push("民族");
console.log(a.join());
```

Console 控制台中代码执行结果如图 1-7 所示。

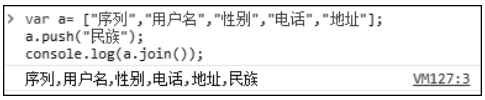


图 1-7　数组用 push()方法追加一列后输出结果

(4) slice 使用举例

```
var a= ["序列","用户名","性别","电话","地址"];
var b=a.slice(1,3);
console.log(b.join());
```

代码执行结果如图 1-8 所示。

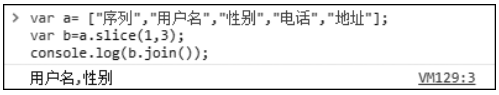


图 1-8　数组用 slice()方法截取两列后输出结果

可以自行执行以上代码,看一下反馈的结果,结合表 1-1 中的说明,即可明白各个方法的用处。

### 1.3.3　二维数组映射数据表

一行多列的数组叫作一维数组,多行多列的数组叫作二维数组。只需嵌套中括号即可定义二维数组:

```
var arr=[[],[]];
```

使用二维数组可以非常方便地将一个关系数据库的表或视图映射到客户端浏览器内存中,比如要存储一个用户列表时,可以用代码 1-4。

**代码 1-4**　定义二维数组映射用户表数据。

```
var t_user_grid=[["序列","用户名","性别","电话","地址"],
["1","张三","男","1366666666","北京市海淀区万寿路"],
["2","李四","男","1588888888","北京市朝阳区 CBD"]];
```

在上面的代码中,将二维码数组的变量名称定义为以 t_开头、以_grid 结尾,两个下画

线中间的是用户的英文翻译,这种命名规则是低代码编程中常用的二维数组命名规则。主要是为了说明映射的是关系型数据库中的哪个表格。

二维数组映射数据表的优点是显而易见的。因为将数据库表格中长期不变的数据或者较少更新的数据一次性地从数据库中查出,传输到前端并存储在 JavaScript 数组中,可以大大减少前后端的握手次数,提高程序运行速度。二维数组此时就相当于前端的内存数据库表格。

二维数组用下标来定位数据,行列也都是从 0 开始计数,比如第 0 行第 0 列的值 users[0][0]等于"1",第 2 行第 4 列的值 users[2][4]等于"北京市朝阳区 CBD"。

后面章节即将讲解的周服的低代码构件就巧妙地使用了内存数据库技术,所以其构件和组件的两种数据会频繁地用到二维数组,一种是从数据库里面读出的数据,会被 Java 构件以 JavaScript 二维数组的形式输出到前端;另一种是构件的参数,有的是二维数组,这个二维数组内容有的是从数据库读出的数据,有的可能是程序员自己定义的。

后面会大量用到二维数组,需要熟练掌握。

### 1.3.4 思考题 1-2:根据姓名查找同学录中的同学信息

这一小节我们来做一个思考题,即根据姓名查找同学录中的同学信息。

**代码 1-5** 根据姓名查找同学录中的同学信息。

```
1  var t_student_grid=[["序列","姓名","性别","年龄","籍贯","手机号","班级"],
2      ["1","张三","男","20","北京","1366666666","1 班"],
3      ["2","李四","男","20","上海","1588888888","1 班"],
4      ["3","王花","女","19","北京","1361111111","1 班"],
5      ["4","赵月","女","19","上海","1581111111","1 班"],
6  ];
7  function info(name){
8      for(var i=0;i<t_student_grid.length;i++){
9          console.log("循环"+i+"="+t_student_grid[i].join(","));
10         if(name==t_student_grid[i][1])return t_student_grid[i].join(",");
11     }
12 }
13 info("赵月");
```

上面代码的主要功能是:先定义一个二维数组来保存所有学生信息,然后定义一个方法,方法的主要功能是对二维数组进行 for 循环遍历,一旦匹配到了正确的学生信息,就返回。代码的详细说明如下。

第 1 行定义了 t_student_grid 二维数组,以 _grid 结尾表示该数组是映射数据库中的 t_student 表格,二维数组的第一行是表头。

第 2～6 行都是数组的内容,这些内容通常都是从数据库中读出来的。

第 7 行定义了 info()方法,info()方法有一个参数 name。

第 8 行使用 for 循环语句对 t_student_grid 进行遍历。

第 9 行向控制台输出二维数组的每一行内容,使用了 join()方法。

第 10 行用 if 语句判断数组的第 1 列,即姓名是否和参数 name 相同,若相同则终止循环,返回该学生信息。