项目一

创建 JavaScript 程序

本项目主要内容

- 搭建开发环境
- 学习 JavaScript 语法
- 在网页中添加 JavaScript 程序
- 使用外部 js 文件

JavaScript 是一种可以给网页增加交互性的脚本语言，能为用户提供更好、更令人兴奋的体验。JavaScript 能够实现一些任务控制，可以创建活跃的用户界面，并能为用户提供页面间导航的反馈。例如，在一些网站上，当鼠标指针停留在按钮上时，会突出显示按钮。

任务一　搭建开发环境

在网页设计中，HTML、JavaScript 和 CSS 文件必须是纯文本格式的。JavaScript 的源程序是文本文件，因此可以使用任何文本编辑器来编写程序源代码，如 Windows 操作系统中的“记事本”程序。为了更快速地编写程序并且降低出错的概率，通常会选择一些专业的代码编辑工具。专业的代码编辑器有代码提示和自动完成功能，比如 Adobe Dreamweaver（以下简称 DW）、Notepad++、Editplus。本书只介绍 DW 的安装。

本书安装的版本是 Dreamweaver CS6。官方简体中文版的安装文件及破解补丁都可以在其官方网站下载：https://www.adobe.com/cn/products/cs6/dreamweaver.html。安装步骤如下：

（1）运行 Dreamweaver CS6 安装程序，如图 1-1 所示。第一步是将其解压到指定目录中，解压完成后，会自动运行安装程序。

（2）安装程序可能会弹出错误提示，如图 1-2 所示，单击“忽略”按钮。

图 1-1　Dreamweaver 安装程序

图 1-2　错误提示

（3）在“欢迎”页面，如果没有安装序列号，那么可以选择“作为试用版安装”，如图 1-3 所示。

图 1-3 “欢迎”页面

（4）在“Adobe 软件许可协议”页面，单击“接受”按钮，如图 1-4 所示。

图 1-4 “Adobe 软件许可协议”页面

（5）在“选项”页面，选择软件安装的路径，如“E:\软件安装列表”，如图 1-5 所示。

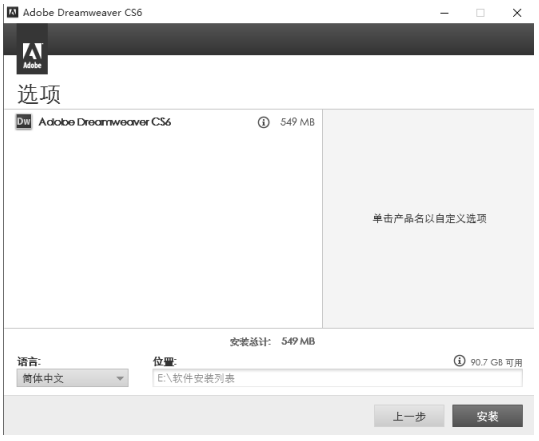

图 1-5 “选项”页面

（6）选择路径后，单击“安装”按钮，开始显示安装进度，页面如图 1-6 所示。

图 1-6 “安装”页面

（7）安装完成后，页面如图 1-7 所示。

图 1-7 “安装完成”页面

任务二　学习 JavaScript 语法

下面主要介绍 JavaScript 中变量、各种数据类型和运算符的概念。

一、变量

在 JavaScript 中，声明变量的关键字是 var，而不像其他编程语言那样区分整型、浮点、字符型等数据类型，且使用不同的关键字（int、double、string、char）。

与代数一样，JavaScript 变量可用于存放值（如 x = 2）和表达式（如 z = x + y）。变量可以使用短名称（如 x 和 y），也可以使用描述性更好的名称（如 age、sum、totalvolume）。变量命名有如下规则：

- 变量名称必须以字母、$（不推荐）或符号（不推荐）开头。
- 变量名称区分大小写（例如，y 和 Y 是不同的变量）。
- 变量不能和 JavaScript 的关键字同名。

【提示】JavaScript 语句和 JavaScript 变量都区分大小写。

JavaScript 声明变量的语句如下：

```
var  x = 2;
var  y = 3;
var  z = x+y;
```

二、基本数据类型

1．数值型

JavaScript 中用于表示数字的数据类型称为数值型，不区分整型、浮点型。数值型用双精度浮点值来表示数字，可以表示（−253，+253）区间中的任何值。数字的值可以使用普通的记法，也可以使用科学记数法。定义数值型数据的语句如下：

```
var  a = 123;
var  b = 25;
var  c = 100;
```

2．布尔型

布尔型是只有“真”和“假”两个值的数据类型。作为逻辑表达式的结果，真用“true”表示，假用“false”表示。事实上，非 0 值即为“真”，0 值即为“假”。布尔型数据通常用来表示某个条件是否成立。定义布尔型数据的语句如下：

```
var d = 'true';
var e = 'false';
```

3．字符串型

在 JavaScript 中，字符串型数据是用引号引起的文本字符串。例如，"你好"或'你真是个聪明的孩子'。每个字符串数据都是 String 对象的实例，其主要用于组织处理由多个字符构成的数据串。字符串可以是引号中的任意文本，可以使用单引号或双引号：

```
var name = 'amy';
var class = "1 班";
var grade = "89";
```

三、复合数据类型

除了基本的数据类型，JavaScript 还支持复合数据类型。复合数据类型包括对象和数组两种。对象其实就是一些数据的集合，这些数据可以是字符串型、数值型和布尔型，也可以是复合型。JavaScript 的对象及其作用如表 1-1 所示。

表 1-1　JavaScript 的对象及其作用

名　称	作　用
Object	所有对象的基础对象
Array	数组对象，封装了数组的操作和属性
ActiveXObject	活动控件对象
Arguments	参数对象，正在调用的函数的参数
Boolean	布尔对象，提供同布尔型等价的功能
Date	日期对象，封装日期相关的操作和属性的对象
Error	错误对象，保存错误信息
Function	函数对象，用于创建函数
Global	全局对象，所有的全局函数和全局常量归该对象所有
Math	数学对象，提供基本的数学函数和常量
Number	数字对象，代表数值数据类型和提供数值常数的对象
RegExp	正则表达式对象，保存正则表达式信息的对象
String	字符串对象，提供串操作和属性的对象

1．日期对象

JavaScript 将与日期相关的所有特性封装进 Date 对象中，包括日期信息及其操作，主要用来进行与时间相关的操作。Data 对象的一个典型应用是获取当前系统时间，使用前首先创建该对象的一个实例，三种创建方法的语法分别如下：

```
date = new Date();                                    // 直接创建
date = new Date( val );                               // 指定日期创建
date = new Date( y,m,d[,h[,min[,sec[,ms]]]] );        // 指定年、月、日、时、分、
                                                      // 秒创建
```

参数说明：

（1）val 是必选项。表示指定日期与 1970 年 1 月 1 日 00:00:00 相差的毫秒数。

（2）y，m，d 分别对应年、月、日，必选。h，min，sec，ms 分别对应时、分、秒、毫秒，可选。

这三种创建方法中，根据需要选择一种即可。第一种方法创建一个包含创建时间值的 Date 对象，第二种方法创建一个和 1970 年 1 月 1 日 00:00:00 相差 val 毫秒数的 Date 对象，第三种方法创建一个指定年、月、日、时、分、秒的 Date 对象。

【范例 1-1】显示程序运行时的本地时间。

```
<script language = "javascript">          // 脚本程序开始
var cur = new Date();                     // 创建当前日期对象 cur
var years = cur.getFullYear();            // 从日期对象 cur 中取得年份
var months = cur.getMonth();              // 取得月份
var days = cur.getDate();                 // 取得日期
var hours = cur.getHours();               // 取得（小）时数
var minutes = cur.getMinutes();           // 取得分（钟）数
var seconds = cur.getSeconds();           // 取得秒数
                                          // 显示取得的各个时间值,输出日期信息

alert( "此时时间是：" + years + "年" + (months+1) + "月"+ days + "日" + hours
                                  + "时" + minutes + "分"+ seconds + "秒" );
</script>
```

上述代码中，alert()是 window 对象的方法，其作用是弹出警告对话框。也可以写成 window.alert()，通常省略 window 对象。

运行结果如图 1-8 所示。

图 1-8　显示程序运行时的本地时间

2. 数学对象

数学（Math）对象封装了与数学相关的特性，包括一些常数和数学函数，主要用于简单的基本数学计算。Math 对象和全局对象一样，不能使用 new 运算符创建，需要在程序运行时由 JavaScript 环境创建并初始化。

【范例 1-2】从 Math 对象中获取圆周率常数，计算一个半径为 2 单位的圆的面积。

```
<script language = "javascript">          // 脚本程序开始
var r = 2;                                // 定义变量表示半径
var pi = Math.PI;                         // 从 Math 对象中读取圆周率 PI 常量
var s = pi*r*r;                           // 计算面积
alert("半径为 2 单位的圆的面积为：" + s + "单位" );// 显示圆的面积
</script>
```

运行结果如图 1-9 所示。

图 1-9　计算一个半径为 2 单位的圆的面积

3. 全局对象

全局（Global）对象是所有全局方法的拥有者，用来统一管理全局方法，全局方法也就是全局函数。该对象不能使用 new 运算符创建对象实例，所有的方法直接调用即可。

4. 字符串对象

字符串（String）对象封装了与字符串有关的特性，主要用来处理字符串。通过 String 对象，可以对字符串进行剪切、合并、替换等操作。可以调用该对象的构造函数创建一个实例。其实，在定义一个字符串型变量时也就创建了一个 String 对象实例。调用 String 对象的方法或属性的形式为“对象名.方法名”或“对象名.属性名”，其构造函数如下：

```
String([strVal]);
```

参数 strVal 是一个字符串，可选。上面的函数创建一个包含值为 strVal 的 String 对象。

【范例 1-3】使用字符串的形式输出诗歌《静夜思》。

```
<script language = "javascript">                    // 脚本程序开始
var comment = "静夜思"+"<br/>"+"李白"+"<br/>"+"床前明月光，"+"<br/>"+"疑是
              地上霜。"+"<br/>"+"举头望明月，"+"<br/>"+"低头思故乡。";
document.write(comment);                            // 输出内容
</script>
```

运行结果如图 1-10 所示。

图 1-10　输出诗歌《静夜思》

其中，write()方法的作用是向网页文档中输出文本内容，它是文档对象 document 的方法。
是 HTML 标签，其作用是在网页中插入简单的换行符。“+”运算符用于把文本值或字符串变量连接起来。

5. 数组对象

数组（Array）对象是 JavaScript 中另一种重要的数据类型。内部对象 Array 封装了所有与数组相关的方法和属性，其内部存在多个数据段组合存储。可以形象地将其理解为：有很多连续房间的楼层，每个房间都可以存放货物，提取货物时只需要给出楼层号和房间编号即可。

四、运算符

1. 基本算术运算符

基本算术运算符用于执行变量与/或值之间的算术运算，如表 1-2 所示。

表 1-2　基本算术运算符（y=5）

运 算 符	描　　述	例　　子	结　　果
+	加	x = y+2	x = 7
−	减	x = y−2	x = 3
*	乘	x = y*2	x = 10
/	除	x = y/2	x = 2.5
%	求余数（保留整数）	x = y%2	x = 1
++	累加	x = ++y	x = 6
−−	递减	x = −−y	x = 4

注　意

“+”运算符用于字符串运算时，把文本值或字符串变量连接起来。若需要把两个或多个字符串变量连接起来，则可使用“+”运算符。

```
txt1 = "What a very";
txt2 = "nice day";
txt3 = txt1+txt2;
```

2. 复合运算符

复合运算符如表 1-3 所示。

表 1-3　复合运算符（x=10，y=5）

运 算 符	例　　子	等 价 于	结　　果
=	x = y		x = 5
+=	x+ = y	x = x+y	x = 15
−=	x− = y	x = x−y	x = 5
*=	x* = y	x = x*y	x = 50
/=	x/ = y	x = x/y	x = 2
%=	x% = y	x = x%y	x = 0

这些运算符的优先级如表 1-4 所示。

表 1-4　运算符的优先级

运 算 符	描　　述
.、[]、()	字段访问、数组下标、函数调用及表达式分组
++、--、-、~、!	一元运算符
*、/、%	乘法、除法、取模
+、-、+	加法、减法、字符串连接
<<、>>、>>>	移位
<、<、=、>、>=	小于、小于等于、大于、大于等于
==、!=、===、!==	等于、不等于、严格相等、非严格相等
&	按位与
^	按位异或
|	按位或
&&	逻辑与
||	逻辑或
?:	条件
=、oP=	赋值、运算赋值
,	多重求值

五、数据类型的转换

1. 隐式类型转换

程序运行时，系统根据当前上下文的需要，自动将数据从一种类型转换为另一种类型的过程称为隐式类型转换。此前的代码中，大量使用了 window 对象的 alert()方法和 document 对象的 write()方法。可以向这两个方法中传入任何类型的数据，这些数据最终都会被自动转换为字符串型。

2. 显式类型转换

与隐式类型转换相对应的是显式类型转换，此过程需要手动将某数据转换为目标类型。要将某一类型的数据转换为另一类型的数据需要用到特定的方法，比如前面用到的 parseInt()、parseFloat()等方法。例如：

```
var PI = 3.1415;
var a = parseInt(PI);    //经过转换后，a 的值为 3
```

任务三　项目实施

一、任务目标

（1）熟练掌握在网页中添加 JavaScript 程序的方法。

（2）熟练掌握使用外部 js 文件的方法。

二、任务内容

1. 在网页中添加 JavaScript 程序

由于 JavaScript 程序是要结合网页使用的，因此在学习 JavaScript 前需要具备一定的网页基础知识。目前需要了解的 HTML 基础知识如表 1-5 所示。

表 1-5　HTML 基础知识

标　签	意　义
<html>	包含网页的 HTML 部分
<head>	包含网页的页头部分
<script>	包含网页的脚本或对外部脚本文件的引用
<src>	包含外部脚本的位置
<title>	包含网页的标题
<body>	包含网页的内容
<h1>…<h6>	这些标签的内容作为标题信息。<h1>的内容是最大尺寸的标题，<h6>的内容是最小尺寸的标题
<a>	链接到另一个网页
<href>	指定当单击链接时，应该转到哪里
<id>	分配给链接的 id

通常，脚本可以放在 HTML 页面上的两个位置：<head>和</head>标签之间（头脚本，header script），或者<body>和</body>标签之间（体脚本，body script）。有一个标出脚本的 HTML 容器标签，这个标签以<script>开头，以</script>结束。其中，单行注释以//开头。多行注释以/*开始，以*/结尾。例如：

```
<script type = "text/javascript" >  //这是<script>开始标签
/*写入"Hello, World!"，每行 JavaScript 代码的末尾都使用分号结束*/
document.write("Hello, World!");    //输出 Hello, World!
</script>                           //这是<script>结束标签
```

需要注意的是，<script>标签的 type 属性代表文件类型；如果每一行只有一条语句，那么在 JavaScript 行的末尾使用分号是可选的，但建议每一句都以分号结尾；一个页面上可以有任意数量的<script>标签，因此也会有多个脚本。

2. 使用外部 js 文件

在 HTML 页面上直接编写的 JavaScript 脚本只能被当前页面使用，因此，这种脚本有时候称为内部脚本。但是，制作网站的过程中需要让多个 HTML 页面共享一个脚本。这要通过包含外部脚本的引用来实现，也就是只包含 JavaScript 的单独文件。这些外部文件称为 js 文件，文件名都以.js 后缀结尾。各个 HTML 页面只需在<script>标签中添加 src 属性，就可以调用 js 文件。这就大大减少了每个页面上的代码。更重要的是，这会使站点更容易维护。当需要对脚本进行修改时，只需修改 js 文件，所有引用这个文件的 HTML 页面会自动受到修改的影响。

三、操作步骤

1. 在网页中添加 JavaScript 程序

下面编写一个简单的 JavaScript 程序，来了解其语法结构和在 HTML 页面中的嵌入方法。创建 test01.html，具体代码如下：

```
<html>                          //HTML 文档开始
<body>                          //文档体开始
<script type = "text/javascript">           //脚本程序
document.write("Hello, World!");            //输出经典的 Hello, World!
</script>                       //脚本结束
</body>                         //文档体结束
</html>                         //HTML 文档结束
```

运行结果如图 1-11 所示。

图 1-11　第一个 JavaScript 程序

2. 使用外部 js 文件

创建 test02.html，在<script>标签中引用外部 js 文件 test1.js。test02.html 的代码如下：

```
<!DOCTYPE html>
<html>
<head>
<title>My second script</title>
<script type = "text/javascript" src = " test1.js"></script>
</head>
<body><h1 id = "helloMessage">
</h1>
</body>
</html>
```

程序中的脚本文本 test1.js 是一个外部 js 文件，其具体代码如下：

```
document.write ("Hello, World!");
```

test02.html 的运行结果与图 1-11 相同。

四、拓展内容

在完成以上要求的实训内容后，可以进一步尝试在一个网页中使用多对<script> </script>标签添加 JavaScript 代码，也可以尝试在两个网页中调用同一个外部 js 文件中的 JavaScript 代码。