



模块 1

JavaScript 简介



学习目标

- (1) 熟悉 JavaScript 的用途和发展状况。
- (2) 了解 JavaScript 开发工具。
- (3) 熟悉 JavaScript 的基础知识。



任务描述

制作一个网页,网页自动弹出一个文本框,提示用户输入密码。获取用户输入的密码后,判断用户密码是否正确。如果正确,则在弹出的提示框中提示“密码输入正确”,如果不正确,则提示“密码输入错误”。

分析:任务中有用户密码判断,单纯用 HTML 代码是无法实现的,需要用到 JavaScript 脚本语言。用户输入密码需要用 `window.prompt()` 函数,然后从文本框中获取所输入的密码,把输入的密码和用户密码相比对,根据用户输入的密码正确与否,用 `alert()` 弹出相应的提示。



任务准备

要完成密码判断任务,需要做如下准备:

- (1) 安装 Dreamweaver 软件。
- (2) 安装浏览器。
- (3) 了解 JavaScript 基础知识。



1.1 JavaScript 概述

1.1.1 JavaScript 介绍

JavaScript 是 Web 开发领域中一门功能强大的脚本编程语言,也是一种通用的、跨平台的、基于对象和事件驱动并具有安全性的编程语言。JavaScript 不需要经过编译,而是直接嵌入在 HTML 页面中,通过浏览器的 JavaScript 引擎直接解释运行,把原本只做显示的静态页面转变成具有良好交互性能的页面程序。

对于编写网页所起的作用,HTML、CSS 和 JavaScript 分别代表了结构、样式和行为,结构是网页的框架,样式是网页的外观,行为是网页的交互逻辑,如表 1-1 所示。

表 1-1 比较 HTML、CSS 和 JavaScript

语 言	作 用	说 明
HTML	结构	构建网页结构
CSS	样式	美化页面
JavaScript	行为	提升用户体验

下面通过一些示例来展示 JavaScript 能够编写的效果,如图 1-1 所示。



图 1-1 JavaScript 在网页中的应用



从图 1-1 可以看出,利用 JavaScript 可以实现网页中常见的交互效果。JavaScript 可以使网页的交互性更强,提升用户体验。

1.1.2 JavaScript 的诞生与发展

20 世纪 90 年代,Internet 普及得越来越广泛,网页的内容也更加丰富和复杂。但与此同时,大部分用户只能通过 28.8 Kb/s 的 Modem 来连接网络,网络速度慢且上网流量贵,有时一些简单的操作却要花费用户大量的时间、精力与金钱。如用户填写完自己的个人资料,单击“提交”按钮,等待 1 分钟后,看到的却是一条提示你忘记填写出生年月的返回信息。提高用户上网体验成为浏览器开发公司的迫切需求。1994 年,网景(Netscape)公司发布了人类历史上第一个比较成熟的浏览器——Navigator 浏览器 0.9 版,轰动一时。但是,这个版本的浏览器只能用来浏览,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。

1995 年,Netscape 公司的布兰登·艾奇(Brendan Eich)成功为即将发行的 Netscape Navigator 2.0 浏览器设计出了一种脚本语言。后来 Netscape 公司与 Sun 公司联手完成了这种脚本语言开发,并将其命名为 LiveScript。由于当时 Sun 公司推出的 Java 语言正备受关注,为了利用 Java 这个时髦词汇从而更好地推广自己的脚本语言,在 Netscape Navigator 2.0 发行前夕,Netscape 公司将 LiveScript 改名为 JavaScript,这就是 JavaScript 1.0 版本。

搭载 JavaScript 1.0 版本的 Netscape Navigator 2.0 浏览器大获成功。紧接着,Netscape 公司在 Netscape Navigator 3.0 浏览器中发布了 JavaScript 1.1 版本。与此同时,Microsoft(微软)公司开始进军浏览器市场,发布的 IE(Internet Explorer) 3.0 搭载了一个 JavaScript 的类似版本,叫作 Jscript。Microsoft 公司的这一举动极大地推动了 JavaScript 的进一步发展。在 Microsoft 进入浏览器这一领域之后,市场上存在 3 种 JavaScript 版本,即 Netscape Navigator 3.0 中的 JavaScript、IE 中的 Jscript 以及 CEnvr 中的 ScriptEase。但这三个版本却没有统一的语法或特性,这一突出的问题降低了用户的使用体验,这个语言的标准化显然势在必行。

1997 年,JavaScript 1.1 作为一个草案提交给 ECMA(欧洲计算机制造商协会),最终由来自 Netscape、Sun、Microsoft、Borland 和其他一些对脚本编程感兴趣的公司的程序员组成了 TC39 委员会,该委员会被委派来标准化一个通用、跨平台、中立于厂商的脚本语言的语法和语义。TC39 委员会制定了“ECMAScript 程序语言的规范书”,即 ECMA-262 标准,该标准被国际标准化组织(ISO)采纳通过,作为各种浏览器开发使用的脚本程序的统一标准。

设计之初,JavaScript 语言嵌入网页中,用来控制浏览器的行为。今天的 JavaScript 承担了更多的责任,尤其是当 Ajax 技术兴起之后,浏览器和服务端可以进行异步交互,网站的用户体验得到了更好的提升。例如,当用户在百度的搜索框中输入关键字后,网页会自动补齐用户可能要搜索的内容,如图 1-2 所示。

另外,JavaScript 的用途已经不再局限于浏览器,Node.js 的出现使得 JavaScript 代码能够运行在服务器上,这极大地扩大了 JavaScript 的应用范围。



图 1-2 百度搜索框

1.1.3 JavaScript 的特点

1. JavaScript 是脚本语言

脚本(script)简单地讲就是一条条的文本命令,按照程序流程执行。常见的脚本语言有 JavaScript、VBScript、Perl、PHP、Python 等,而 C、C++、Java、C# 这些语言不属于脚本语言。它们的区别在于,非脚本语言一般需要经过编译、链接、生成独立的可执行文件才能运行;而脚本语言依赖于解释器,只在被调用时自动进行解释或编译。脚本语言缩短了传统语言编写→编译→链接→运行的过程。

脚本语言通常都有简单、易学、易用的特点,语法规则比较松散,使开发人员能够快速完成程序的编写工作,但其缺点是执行效率不如编译型的语言快。不过,由于计算机的运行速度越来越快,Web 应用的需求变化也越来越快,人们更加重视软件的开发速度,脚本语言带来的执行效率慢已经可以被忽视了。

2. JavaScript 可以跨平台

JavaScript 不依赖操作系统,只要有浏览器就能运行。目前,市面上绝大部分浏览器都支持 JavaScript。在移动互联网时代,利用手机等各种移动设备上网的用户越来越多,JavaScript 的跨平台性使得它在移动端也起着重要的作用。例如,JavaScript 可以搭配 CSS3 编写响应式的网页,或者将网页编写成具有移动 App 的交互方式,从而缩短 App 的开发和更新周期。JavaScript 还可以结合 HTML5 中的 Canvas 技术,在网页上制作精美的动画和游戏。





3. JavaScript 支持面向对象

面向对象是软件开发中的一种重要的编程思想,其优点众多,可以极大地提高程序的开发效率和速度,降低成本。例如,基于面向对象思想编写的 jQuery 框架,大大地提高了 JavaScript 的开发快捷性,使 JavaScript 的开发效率出现了第一次飞跃。近几年,Web 前端开发技术日益受到重视,又诞生了 Bootstrap、AngularJS、Vue.js、Backbone.js、React、webpack 等框架和工具。



1.2 开发工具

JavaScript 的开发工具主要包括浏览器和代码编辑器两种软件。浏览器用于执行、调试 JavaScript 代码,代码编辑器用于编写代码。本节将针对这两种开发工具进行讲解。

1.2.1 浏览器

浏览器是访问互联网上各种网站所必备的工具。由于浏览器的版本比较多,作为 JavaScript 开发人员需要解决各种浏览器的兼容性问题,确保用户使用的浏览器能够准确执行自己编写的程序。常见的浏览器如表 1-2 所示。

表 1-2 常见浏览器

开 发 商	浏 览 器	特 点
Microsoft	Internet Explorer	Windows 操作系统内置浏览器,用户数量多
	Microsoft Edge	只内置于 Windows 操作系统,速度更快
Google	Chrome	市场占有率高,稳定、简洁
Mozilla	Firefox	简洁,市场占有率略低于 Chrome
Apple	Safari	只应用于 iOS、Mac OS 操作系统

在表 1-2 列举的浏览器中,Internet Explorer 历史最悠久,主要版本有 6、7、8、9、10、11,但版本 6、7、8 由于发布时间早,兼容性较差,已经被市场淘汰。Chrome 浏览器是各方面都比较优秀的浏览器,除了占据内存大,没有其他明显缺点,本书将选择 Chrome 浏览器进行详解。

1.2.2 代码编辑器

1. Adobe Dreamweaver

Adobe Dreamweaver 简称 DW,中文名称为“梦想编织者”,最初由美国 Macromedia 公司开发,2005 年被 Adobe 公司收购。Adobe Dreamweaver 是一个集网页制作和网站管理于一身的所见即所得的网页编辑器,降低了网页的开发难度和 HTML、CSS 的学习门槛,适合新手使用。用户借助经过简化的智能编码引擎,可以轻松地创建、编码和管理动态网站;访问代码提示,即可快速了解 HTML、CSS 和其他 Web 标准;使用视觉辅助功能可以减少错



误并提高网站开发速度。其缺点是可视化编辑功能会产生大量冗余代码,不适合开发结构复杂、需要大量动态交互的网页。

2. HBuilder

HBuilder 编辑器是 DCloud 推出的一款支持 HTML5 的 Web 开发编辑器。HBuilder 通过完整的语法提示和代码输入法、代码块等,大幅提升 HTML、JavaScript、CSS 的开发效率,同时还包括最全面的语法库和浏览器兼容性数据,帮助开发者快速完成开发,解决浏览器碎片化问题。

3. Sublime Text

Sublime Text 是一个轻量级、跨平台的代码编辑器,同时支持 Windows、Linux、Mac OS X 等操作系统,也是优秀的 HTML 和 CSS 编辑器。Sublime Text 是由程序员 Jon Skinner 于 2008 年 1 月开发出来的,它最初被设计为一个具有丰富扩展功能的 Vim。Sublime Text 具有美观、友好的用户界面和强大的功能,如代码缩略图、Python 的插件、代码段等,还可自定义键绑定、菜单和工具栏。Sublime Text 的主要功能包括拼写检查、书签、完整的 Python API、Goto 功能、即时项目切换、多选择、多窗口等。

4. WebStorm

WebStorm 是 JetBrains 公司旗下一款 Web 前端开发工具,JavaScript、HTML5 开发效率高,支持许多流行的前端技术,被中国广大 JavaScript 开发者誉为“Web 前端开发神器”“最强大的 HTML5 编辑器”“最智能的 JavaScript 编辑器”等。与 IntelliJ IDEA 同源,继承了 IntelliJ IDEA 强大的 JavaScript 部分的功能。

考虑到学习 JavaScript 的开发人员,大都会先学习 HTML+CSS3 的开发,而 HTML+CSS3 学习者中使用 HBuilder 的人数较多,所以本书选择使用 HBuilder 进行代码编写,其软件界面如图 1-3 所示。



图 1-3 HBuilder 软件界面





1.3 使用基础

1.3.1 JavaScript 引入方式

在网页编写 JavaScript 时,可以通过嵌入式、外链式和行内式这 3 种方式来引入 JavaScript 代码。下面针对这 3 种方式,分别进行讲解。

1. 嵌入式

嵌入式就是使用<script>标签包裹 JavaScript 代码,直接编写到 HTML 文件中,具体如下:

(1)使用 HBuilder 创建一个“test.html”文件。在“文件”菜单栏下执行“新建”→“7.html 文件”命令,即可创建一个 HTML 文件,如图 1-4 所示。



图 1-4 创建 HTML 文件

创建的 HTML 文件代码如图 1-5 所示。



图 1-5 HTML 文件代码



(2)从图 1-8 可以看出,新建的“test.js”文件中没有任何代码。在其中添加代码,如图 1-9 所示。



图 1-9 添加代码

然后使用<script>标签的 src 属性引入文件,如图 1-10 所示。其中,<script>标签的 type 属性用于告知浏览器脚本的类型。src 属性是一个文件路径或者 URL 地址,可以指定相对路径、绝对路径或者 URL 地址。



图 1-10 通过 src 属性引入文件

3. 行内式

行内式是将 JavaScript 代码作为 HTML 标签的属性值使用。例如,单击超链接时,弹出一个警告提示框,代码如图 1-11 所示。



图 1-11 标签属性嵌入 JavaScript 代码

图 1-11 代码实现了单击链接时,弹出一个警告框提示“将进行跳转”。由于现在网页开始提倡结构、样式、行为相分离,即分离 HTML、CSS、JavaScript 三部分的代码,避免直接写在 HTML 标签的属性中,从而便于更新和维护。因此实际开发中不推荐使用行内式。

1.3.2 输出语句

利用输出语句可以输出一段代码的执行结果,在学习 JavaScript 的过程中会经常用到输出语句。接下来介绍 3 种常用的输出语句。



1. alert()

alert()用于弹出一个警告框,确保用户可以看到某些信息,在前面的示例中已经演示过。利用 alert()可以很方便地输出一个结果,经常用于测试程序。

2. console.log()

console.log()用于在浏览器的控制台中输出内容。例如,在“test.html”文件中编写如下代码:

```
console.log("学习 JavaScript 程序开发");
```

使用 Chrome 浏览器打开“test.html”文件,按 F12 键启动开发者工具,然后单击“Console”控制台选项卡,如图 1-12 所示。

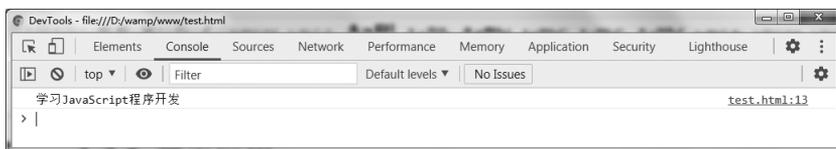


图 1-12 浏览器控制台输出

从图 1-12 中可以看出,控制台显示了输出结果“学习 JavaScript 程序开发”,其最右边的“test.html:13”表示输出的代码来自“test.html”文件中的第 13 行。

3. document.write()

document.write()用于向 HTML 文档页面中输出内容,具体示例如下:

```
document.write("学习 JavaScript! ");
```

1.3.3 注释

在 JavaScript 开发过程中,使用注释是为了提高代码的可读性或者把目前不需要的代码进行暂时的屏蔽。在程序解析的过程当中,注释会被自动忽略。JavaScript 语言支持单行注释和多行注释 2 种注释方式。

1. 单行注释“//”

```
<script type = "text/javascript">  
    console.log("学习 JavaScript 程序开发"); // 向控制台打印一句话  
</script>
```

上述示例中,“//”和其后面的“向控制台打印一句话”是一个单行注释。以“//”开始,到该行结束或者 JavaScript 标记结束之前的内容都是注释。

2. 多行注释“/* */”

```
<script type = "text/javascript">  
    /*  
    alert("我爱中国");
```





```
console.log("江西信息应用职业技术学院");
* /
console.log("jxcia");
</script>
```

上述示例中，“/*”和“*/”之间的内容是多行注释，多行注释以“/*”开始，以“*/”结束。同时，多行注释可以嵌套单行注释，但不能再嵌套多行注释。



任务实施

完成验证用户输入密码的代码如下：

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>验证用户输入的密码</title>
  <script type="text/javascript">
    var password = prompt('请输入密码:'); // 获取用户输入的密码
    if(password == 'jxcia')
    {
      alert("密码输入正确");
    } else {
      alert("密码输入错误");
    }
  </script>
</head>
<body></body>
</html>
```

上述代码中，`var password = prompt('请输入密码:')`获取用户输入的密码，然后判断用户输入的密码是否与“jxcia”相同，如果相同则提示“密码输入正确”，否则提示“密码输入错误”。通过浏览器测试，运行结果如图 1-13 所示。



图 1-13 用户输入密码



输入错误的密码“helloworld”后,单击“确定”按钮,运行结果如图 1-14 所示。

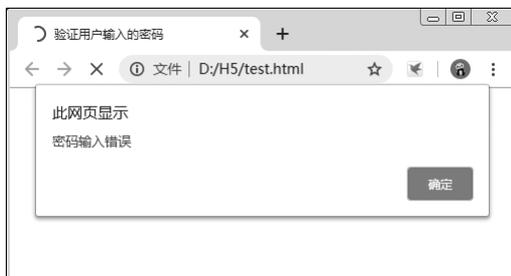


图 1-14 用户密码输入错误

模块小结

本模块首先提出一个验证用户输入密码的任务并进行分析,针对任务介绍了 JavaScript 的应用领域、发展状况及特点,浏览器、代码编辑器相关内容,JavaScript 的基础知识(引入方式、输出语句和注释)。最后利用相关知识完成任务。

自测题

一、判断题

1. JavaScript 来源于 Java 语言,它们是由同一家公司开发的。 ()
2. JavaScript 的多行注释里面还可以嵌套多行注释。 ()
3. 嵌入式引入 JavaScript 代码,可以使 JavaScript 代码和 HTML 代码处于同一网页,便于代码检查,比较推荐。 ()
4. JavaScript 只能在 Windows 系统上运行。 ()

二、选择题

1. 下面不是 JavaScript 引入方式的是()。
A. 嵌入式 B. 并联式 C. 外链式 D. 行内式
2. 下面不是 JavaScript 的输出语句的是()。
A. printf() B. console.log()
C. alert() D. document.write()





模块 2

JavaScript 基本语法



学习目标

- (1) 掌握 JavaScript 变量的定义及赋值。
- (2) 掌握 JavaScript 的数据类型。
- (3) 掌握 JavaScript 的表达式及运算符。
- (4) 熟悉 JavaScript 流程控制语句,并具备运用控制语句解决问题的能力。



任务描述

九九乘法表是学习数学必备的知识,它体现了数字之间的乘法规律,如何使用 JavaScript 输出九九乘法表呢?

分析:用 JavaScript 输出九九乘法表需要使用循环嵌套的知识来完成。首先考虑九九乘法表有九行,需要一个循环变量来控制行数的变化,由于每行的列数不一样,因此还需要一个循环变量来控制每行的列数。观察九九乘法表会发现以下规律:

- (1) 每行中公式的列数等于该行的行号,例如第 4 行共有 4 列公式;
 - (2) 每行的被乘数的取值范围在“1 到每行中的列数”,例如第 4 行中,被乘数的取值范围在 1 到 4;
 - (3) 乘数的值等于该行的行号,例如第 4 行中所有公式的乘数的值都为 4。
- 只要根据上述规律使用 JavaScript 语言编写代码,即可将九九乘法表显示在网页中。



任务准备

要完成九九乘法表任务,需要掌握如下知识:

- (1) 基本的网页输出语句。
- (2) 循环语句的基本结构及用法。
- (3) 循环嵌套的用法。



2.1 变 量

2.1.1 标识符

在 JavaScript 中,变量、数组、函数等都需要一个名称,这个名称被称为标识符。用户自己定义的标识符要遵循 JavaScript 标识符命名规则,具体规则如下:

- (1)标识符由字母(A~Z,a~z)、下划线(_)、数字(0~9)和美元符号(\$)组成。
- (2)标识符不能以数字开头。
- (3)标识符不能使用 JavaScript 中的关键字命名,如 case、var、false 等。
- (4)JavaScript 中的标识符严格区分字母的大小写,如 abcd 和 abcd 表示两个不同的标识符。

用户在定义标识符时还需要注意在定义标识符时尽量做到“见名知意”。如用 gender 表示性别、name 表示姓名等,养成良好的编程习惯。

合法标识符的举例:A123、school、name_2、a_23、get_age、\$ stu、\$ s_3。

非法标识符的举例:12age、a-b、a&b、M. teacher、new、ab#、abc¥。

2.1.2 保留关键字

保留关键字是 JavaScript 语言中规定使用的专用词,它们有特定的含义,如定义变量用关键字 var,定义函数用关键字 function。所有的保留关键字都不能作为标识符使用,否则会造成语法错误。JavaScript 的保留关键字如表 2-1 所示。

表 2-1 JavaScript 保留关键字

break	case	catch	class	const	continue
debugger	default	delete	do	else	export
extends	false	finally	for	function	if
import	in	instanceof	new	null	return
supper	switch	this	throw	try	true
typeof	var	void	while	with	yield

除此之外,JavaScript 中还保留了一些未来可能会用到的关键字,如表 2-2 所示。





表 2-2 JavaScript 预保留关键字

abstract	arguments	await	byte	boolean	char
double	enum	eval	final	float	goto
implements	int	interface	let	long	native
package	private	protected	public	short	
synchronized	throws	transient	volatile	static	

2.1.3 变量的使用

变量是内存中的一块存储空间,是用来存储数据的。变量的名字就是内存地址的别名,在程序的执行过程中其内的值是可以改变的。变量有两个基本特征,即变量名和变量值,变量名的命名规则遵循标识符的命名规则。在使用 JavaScript 变量前要先对变量进行声明。由于 JavaScript 是一种弱类型的语言,所有的 JavaScript 变量都是由关键字 var 进行声明的,其语法格式如下:

```
var valuname1[, valuname2, valuname3, ...];
```

var 是声明变量的关键字,所有变量的声明都用该关键字。valuname1 是声明的变量名。一次可以声明一个变量,也可以声明多个变量,多个变量之间要用逗号隔开。

```
var $ name, $ age, $ school;  
var math_score, English_score, chinese_score;  
var room_001, room_002;
```

上述代码中,用 var 声明的变量都没有赋初始值,所以默认为 undefined。行末的分号(;)可以省略,一般我们都会保留,表示一行的结束。

2.1.4 变量的赋值

变量的赋值就是给变量一个初始值,JavaScript 中使用等号(=)实现变量的赋值。变量的赋值有两种方式。

(1)在声明变量的同时给变量进行初始化赋值。

```
var name = "lucy"; // 声明变量的同时赋值
```

(2)声明完变量后,再使用赋值语句进行赋值。

```
var name; // 声明变量  
name = "lucy"; // 变量赋值
```

在 JavaScript 中,虽然变量可以不先声明而直接对其进行赋值(上述代码中可以省略声明变量那行),但还是建议在使用变量之前就对该变量进行声明。这是因为 JavaScript 采用的是动态编译,运行过程中不易发现代码中的错误,特别是变量命名方面的错误,所以要养成良好的编程习惯。



2.2 数据类型

2.2.1 数据类型分类

JavaScript 是一种弱类型的语言,即数据在定义时不需要指明是什么类型,其数据类型可以通过数据赋值时的值来推断。JavaScript 的数据类型可以分为三类:基本数据类型、引用数据类型和特殊数据类型,如图 2-1 所示。引用数据类型会在后续模块中介绍。



视频
数据类型

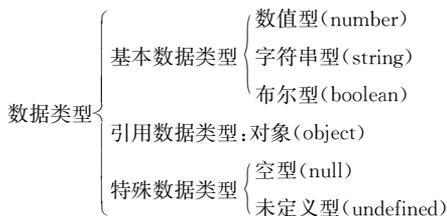


图 2-1 数据类型分类

2.2.2 基本数据类型和特殊数据类型

1. 数值型

数值型是 JavaScript 中最基本的数据类型。JavaScript 并不区分整型和浮点型,所有的数字都是数值型。当一个数字直接出现在 JavaScript 中,那么这个数字被称为数值直接量,JavaScript 常用的数值直接量如下:

(1)十进制。在 JavaScript 中,十进制用 0~9 的数字来表示,如 23、-30、51、-103。

(2)八进制。在 JavaScript 中,八进制用 0~7 的数字来表示,第一位数字前必须加 0,如 023、065、0347、02。

(3)十六进制。在 JavaScript 中,十六进制用 0~9 的数字和 A~F(或者小写字母 a~f)的字母来表示,在字母和数字前要先添加 0X(或者 0x)开头,表示是十六进制数,如 0Xff、0X1234、0xA23、0xEC。

下面通过输出不同进制下的数字 25,加深对这三种进制数的理解。

```
document.write("不同进制下数字 25 的输出结果:"); // 输出字符串
document.write("<p>十进制:" + 25); // 输出十进制数 25
document.write("<p>八进制:" + 025); // 输出八进制数 025
document.write("<p>十六进制:" + 0x25); // 输出十六进制数 0x25
```

执行上面的代码,得到的运行结果如图 2-2 所示。



图 2-2 不同进制下数字 25 的输出结果

(4)浮点型。浮点型数据只能用十进制数表示,表示形式有以下两种:

①一般表示形式:由整数部分、小数点和小数部分组成,如 1.45、-5.6、15.0。

②指数表示形式:由数字、e(或 E)和指数组成,如 3.14e5、5e3、1.56E-6。e 或 E 后面的整数表示 10 的指数次幂,如 3.14e5 表示 3.14×10^5 。

下面通过代码了解浮点型数据的用法,具体如下:

```
document.write("浮点型数据表示:");  
document.write("<p>" + 3.14e5);  
document.write("<p>" + 5e3);  
document.write("<p>" + 1.56E-6);
```

执行上面的代码,得到的运行结果如图 2-3 所示。

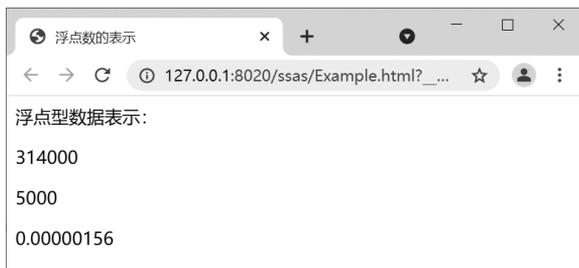


图 2-3 浮点型数据输出结果

2. 字符串型

字符串是由字符组成的序列,主要由字母、数字、汉字或者其他特殊字符组成。字符串数据必须用单引号或者双引号括起来,单引号和双引号可以互相嵌套,即单引号中的字符串可以出现双引号,双引号中的字符串可以有单引号。

```
'it is a sunny day' // 单引号括起来的字符串  
"it is a sunny day" // 双引号括起来的字符串  
"hello, my name is 'lucy' " // 双引号中括起来的字符串出现单引号  
'hello, my name is "lucy" ' // 单引号中括起来的字符串出现双引号
```

通过上述代码可知,单引号和双引号可以互相嵌套,但是单引号中包含单引号,或者双引号中包含双引号,程序则会出错。解决此错误的方法是通过转义字符“\”对其进行转义。



```
'it\'s a sunny day' // 输出 it's a sunny day
"hello, my name is \"lucy\" " // 输出 hello, my name is "lucy"
```

除了可以对单引号和双引号进行转义外,JavaScript 中还有其他的特殊字符需要转义,具体如表 2-3 所示。

表 2-3 JavaScript 常用转义字符

转义字符	描 述	转义字符	描 述
\b	退格	\v	垂直制表符
\n	换行符	\\	反斜杠
\t	水平制表符 Tab	\r	回车符
\f	换页	\OOO	八进制数,范围 000~777
\'	单引号	\xHH	十六进制数,范围 00~FF
\"	双引号	\uhhhh	4 位十六进制数值 hhhh 指定的 Unicode 字符

3. 布尔型

布尔型(Boolean)只有两个值,一个是 true,表示结果为真,一个是 false,表示结果为假,常用于逻辑判断。

```
var result1 = true; // 把布尔型 true 赋值给 result1
var result2 = false; // 把布尔型 false 赋值给 result2
```

4. 空型

空型只有一个值为 null,用于定义空的或者不存在的引用。这里需要注意 null 和空字符串 " 或者 0 不一样,null 一般用于给对象进行初始赋值。

5. 未定义型

未定义型只有一个值为 undefined,当定义了一个变量未进行初始化时,该变量的默认值为 undefined。例如“var a;”定义了变量 a,但是没有给变量 a 进行初始赋值。undefined 和 null 的区别是 null 表示一个变量被赋了一个空值,而 undefined 则表示该变量未被赋值。

2.2.3 数据类型检测

由于 JavaScript 语言是弱类型语言,JavaScript 中的变量的数据类型是由上下文运行过程决定的,变量的加法演示示例如下:

```
var num1 = "23", num2 = 12, sum = 0; // 定义变量并赋值
sum = num1 + num2; // 变量进行加法运算
document.write(sum); // 输出加法结果:2312
```

通过上述案例的输出结果可以看出,变量没有进行加法运算,而是进行了字符串的拼接。由于两个变量的数据类型不同,直接相加并不能得到正确的答案。做数字运算前,要先



视频
数据类型检测