第3章

# JavaScript 对象编程

学习目标

(1) 理解 JavaScript 中的对象。
(2) 掌握 JavaScript 的 Date、Math、String 等对象。
(3) 掌握 JavaScript 的数组声明以及使用方法。
(4) 掌握正则表达式的属性和方法。
(5) 掌握自定义对象的使用方法。

JavaScript 是一种基于对象的语言，它支持 3 种对象：内置对象、用户自定义对象以及浏览器对象。常用的内置对象包括 Date 对象、Math 对象、String 对象、Array 对象及 RegExp 对象等。下面将介绍常用内置对象的属性和方法及如何使用 JavaScript 的内置对象。

## 任务 3.1 在页面动态显示系统时间

### 任务描述

在“我的博客”首页动态显示客户端系统时间，效果如图 3-1 所示。

### 任务分析

在页面动态显示客户端系统时间，可以通过以下步骤实现。
(1) 完成静态页面设计，标识要显示系统时间的位置，此例已完成该步骤。
(2) 定义函数，使用日期和时间对象，获取客户端系统时间。
(3) 给 id 为 time 的标签赋值。
(4) 使用定时函数，每隔一秒重新调用一次函数。

日期和时间对象(Date 对象)主要提供了获取和设置日期与时间的方法。JavaScript 中提供了两个定时器函数：setTimeout()和 setInterval()。

### 3.1.1 Date 对象的创建

在 Web 应用中，经常碰到需要处理时间和日期的情况。JavaScript 脚本内置了核心对象 Date，该对象可以表示从毫秒到年的所有时间和日期，并提供了一系列处理时间和日期


图 3-1　在首页显示动态时钟

的方法。

要使用 Date 对象,必须先使用 new 运算符创建它,Date 对象的构造函数通过可选的参数,可生成过去、现在和将来的 Date 对象,创建 Date 的常见方式有 3 种。

(1) 不带参数

```
var myDate = new Date();
```

创建一个含有系统当前日期和时间的 Date 对象变量 myDate。

(2) 创建一个指定日期的 Date 对象

```
var myDate = new Date("2014/05/01");
```

使用代表日期和时间的字符串创建一个特定日期的 Date 对象,上述语句创建了 2014 年 5 月 1 日的 Date 变量 myDate。

(3) 创建一个指定时间的 Date 对象

```
var myDate = new Date(2014,6,1,10,30,20,50);
```

上述语句创建了一个包含确切日期和时间的 Date 变量 myDate,即 2014 年 6 月 1 日 10 点 30 分 20 秒 50 毫秒。

### 3.1.2　Date 对象的常用方法

Date 对象提供了很多操作日期和时间的方法,方便程序员在脚本开发过程中简单、快捷地操作日期和时间。表 3-1 列出了其常用的方法。

表 3-1　Date 对象常用方法

| 方　法　名 | 描　　述 |
|---|---|
| getYear() | 返回年份数 |
| getFullYear() | 返回年份数 |
| getMonth() | 返回月份数(0～11) |
| getDate() | 返回日期数(1～31) |
| getDay() | 返回星期数(0～6) |
| getHours() | 返回时数(0～23) |
| getMinutes() | 返回分钟数(0～59) |
| getSeconds() | 返回秒数(0～59) |
| getMilliseconds() | 返回毫秒数(0～999) |
| getTime() | 返回对应日期基线的毫秒 |
| toGMTString() | 以 GMT 格式表示日期对象 |
| toLocaleString() | 返回日期的字符串表示,其格式根据系统当前的区域设置来确定 |

**【实例 3-1】** 显示计算 1＋2＋3＋…＋10000 之和所需的运行时间(毫秒数)。

实现代码如下:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>运行时间计算</title>
    <script type="text/javascript">
      var t1,t2,htime,sum=0;
      t1=new Date();
      document.write("循环前的时间是:"
       +t1.toLocaleString()+":"+t1.getMilliseconds()+"<br/>");
      for(var i=1;i<=10000;i++){
          sum+=i;
      }
      t2=new Date();
      document.write("循环后的时间是:"
       +t2.toLocaleString()+":"+t2.getMilliseconds()+"<br/>");
      htime=t2.getTime()-t1.getTime();
      document.write("执行 10000 次循环用时:"+htime+"毫秒");
    </script>
  </head>
  <body>
  </body>
</html>
```

运行代码效果如图 3-2 所示。

**说明:**

(1) 方法 toLocaleString()返回的时间字符串不包括毫秒数,故在输出中使用了方法 getMilliseconds()。

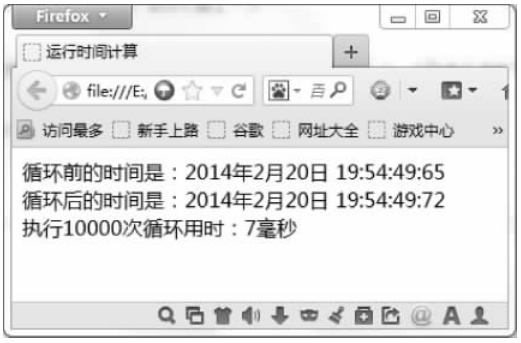


图 3-2　执行 10000 次循环所需时间

(2) 把两个 Date 对象变量的 getTime()方法的返回值进行相减，即 t2. getTime()－t1. getTime()，其效果相当于求这两个时间间隔的毫秒数。

### 3.1.3　定时器函数

JavaScript 中提供了两个定时器函数：setTimeout()和 setInterval()，下面来了解这两个函数。

(1) setTimeout()。

setTimeout()用于在指定的毫秒后调用函数或计算表达式。其语法格式如下：

```
setTimeout("调用的函数名称",等待的毫秒数)
```

使用 clearTimeout()方法可以清除 setTimeout()创建的定时器。

例如：

```
var mytimer = setTimeout("show()",1000);
clearTimeout(mytimer);
```

(2) setInterval()。

setInterval()可按照指定的周期(以毫秒计)来调用函数或计算表达式。其语法格式如下：

```
setInterval ("调用的函数名称",周期性调用函数之间间隔的毫秒数)
```

setInterval()会不停地调用函数，直到窗口被关闭或使用 clearInterval(对象)清除定时器。

**说明：**

setTimeout()只执行函数一次，如果要多次调用函数，需要使用 setInterval()或者在被调用的函数体里调用 setTimeout()。

### 3.1.4　任务实现

在页面动态显示系统时间的操作步骤如下。

(1) 完成静态页面设计，此例已将静态页面设计好，打开 index_Date. html 页面。关键 HTML 代码如下：

```
<body>
<div id="container">
  <div id="top">
      <ul>
        <li class="tdwidth"><a href="register.html">注册</a></li>
        <li><a href="#">登录</a></li>
        <li><a href="#">搜索</a></li>
        <li><a href="#">留言</a></li>
        <li><a href="photo.html">相册</a></li>
        <li><a href="index.html">首页</a></li>
        <li class="liTime" id="time"></li>
      </ul>
  </div>
```

(2) 添加脚本代码,定义函数,实现动态时钟效果,代码如下:

```
<script type="text/javascript">
  function showTime()
  {
      var myDate = new Date();                    //定义日期与时间变量
      var hour = myDate.getHours();
      var minutes = myDate.getMinutes();
      var seconds = myDate.getSeconds();
      if(hour < 10)
          hour = "0" + hour;
      if(minutes < 10)
          minutes = "0" + minutes;
      if(seconds < 10)
          seconds = "0" + seconds;
      document.getElementById("time").innerHTML = "当前时间为: " +
      hour + ":" + minutes + ":" + seconds;   //给 id 为 time 的标签赋值
      setTimeout("showTime()",1000);          //设置定时函数,1 秒执行一次 showTime()函数
  }
  window.onload = showTime;                   //页面加载时调用 showTime()函数
</script>
```

任务 3.1　在页面动态显示系统时间微课

## 任务 3.2　制作随机选号页面

### 任务描述

假定班上有 60 名同学,现制作一个提问选号器,如图 3-3 所示。单击“开始”按钮在页

面随机显示 1～60 的学号,单击“停止”按钮在页面显示选中学号。

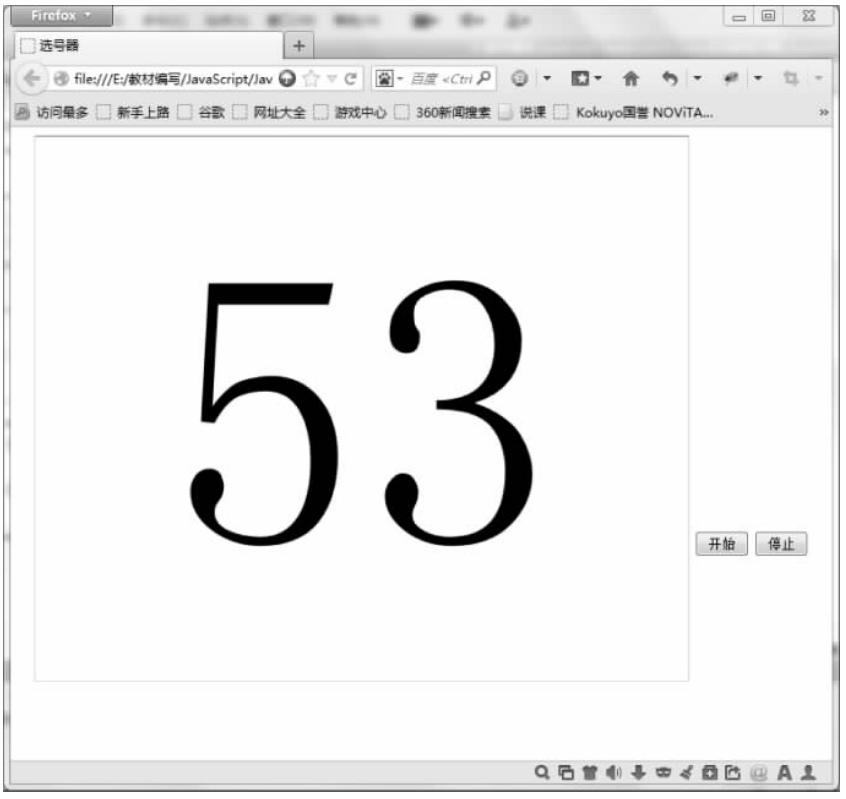


图 3-3　提问选号器页面

## 任务分析

可以通过以下步骤实现随机选号页面的制作。

(1) 产生 1～60 的随机整数,并在页面显示。

(2) 单击“开始”按钮时使用定时函数隔 60 毫秒产生一个随机整数。

(3) 单击“停止”按钮时清除定时函数。

现在问题的关键是产生 1～60 的随机整数。产生随机整数需要用到 Math 对象。Math 对象包含用于各种数学运算的属性和方法。

### 3.2.1　Math 对象的常用属性

Math 对象的内置方法可以在不使用构造函数创建对象时直接调用。使用 Math 对象的属性时,常用格式如下:

```
Math.属性
```

Math 对象的常用属性见表 3-2。

表 3-2　Math 对象的常用属性

| 属　性　名 | 描　　述 |
|---|---|
| E | 自然对数的底数 |
| LN2 | 2 的自然对数 |
| LN10 | 10 的自然对数 |
| LOG2E | 以 2 为底 e 的对数 |
| LOG10E | 以 10 为底 e 的对数 |
| PI | 圆周率 |
| SQRT1_2 | 1/2 的平方根 |
| SQRT2 | 2 的平方根 |

### 3.2.2　Math 对象的常用方法

Math 对象的方法是一些十分有用的数学函数，常用的方法见表 3-3。

表 3-3　Math 对象的常用方法

| 方　　法 | 说　　明 |
|---|---|
| ceil(数值) | 大于等于该数值的最小整数 |
| floor(数值) | 小于等于该数值的最大整数 |
| min(数值 1,数值 2) | 最小值 |
| max(数值 1,数值 2) | 最大值 |
| pow(数值 1,数值 2) | 数值 1 的数值 2 次方 |
| random() | 0～1 的随机数 |
| round(数值) | 最接近该数值的整数 |
| sqrt(数值) | 开平方根 |
| 还有 abs、sin(弧度)、cos、tan、asin、acos、atan、exp、log 等 | |

例如，计算 cos(PI/6)，可以写成：Math.cos(Math.PI/6)。

### 3.2.3　任务实现

完成如图 3-3 所示随机选号器制作的代码如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
    <title>选号器</title>
    <style type="text/css">
      .inputTxt{height:500px;width:600px;font-size:360px;text-align:center;}
    </style>
    <script language="javascript" type="text/javascript">
      var timer;
      function startScroll()
      {
          var num=Math.floor(Math.random()*60+1);     //产生 1-60 的随机整数
```

```
            document.myForm.myText.value = num;                    //文本框赋值
            timer = setTimeout("startScroll()",60);                //使用定时器函数,60 毫秒调用函数
        }
        function stopScroll()
        {
            clearTimeout(timer);                                   //清除定时器
        }
    </script>
  </head>
  <body>
    <center>
      <form name = "myForm">
        <input name = "myText" type = "text" value = "0" class = "inputTxt"/>
        <input name = "start" type = "button" value = "开始" onclick = "startScroll()" />
        <input name = "stop" type = "button" value = "停止" onclick = "stopScroll()" />
      </form>
    </center>
  </body>
</html>
```

任务 3.2　制作随机选号页面微课

## 任务 3.3　制作简单的焦点图效果

### 任务描述

焦点图效果是各大网站常用的效果,下面利用数组实现简单的焦点图效果,如图 3-4 所示,页面上 5 张图片 2 秒轮换显示,单击向右图片实现播放下一张图片,图片向后继续 2 秒轮换显示,单击向左图片实现播放上一张图片,图片向前继续 2 秒轮换显示。


图 3-4　简单焦点图效果

### 任务分析

实现简单焦点图效果可以采用以下步骤。

(1) 设计 HTML 页面,应用 CSS 美化页面。

(2) 定义数组,将轮换显示的图片地址保存到数组中。

(3) 定义两个全局变量,一个变量用于控制定时器,另一个变量用于控制数组下标。

(4) 定义函数实现图片的轮换显示。在函数中改变图片的地址,使用定时器函数,2 秒更换图片地址,实现图片的轮流显示。

(5) 单击上一张或下一张按钮时将定时器清除,再重新调用图片轮换显示函数。

数组是包含基本数据类型和组合数据类型的有序序列,JavaScript 中数组也是最常使用的对象之一,下面介绍数组的创建及其常用属性和方法。

### 3.3.1 数组的创建

数组是值的有序集合,由于弱类型的原因,JavaScript 中数组十分灵活、强大,不像 Java 等强类型高级语言,数组只能存放同一类型或其子类型元素,JavaScript 在同一个数组中可以存放多种类型的元素,而且长度也是可以动态调整的,可以随着数据增加或减少自动对数组长度进行更改。

**1. 创建数组**

在 JavaScript 中,可以使用构造函数创建数组,也可以直接使用方括号创建数组,具体有以下几种创建方法。

(1) 使用无参构造函数,创建一空数组。

```
var arr = new Array();
```

(2) 一个数字参数构造函数,指定数组长度。

```
var arr = new Array(5);
```

表示创建一个长度为 5 的数组,由于数组长度可以动态调整,作用并不大。

(3) 带有初始化数据的构造函数,创建数组并初始化参数数据。

```
var arr = new Array("HTML","JavaScript","DOM")
```

(4) 使用方括号,创建空数组,等同于调用无参构造函数。

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

(5) 使用中括号,并传入初始化数据,等同于调用带有初始化数据的构造函数。

```
var arr = ["HTML","JavaScript","DOM"];
```

**2. 为数组元素赋值**

在声明数组时可以直接为数组元素赋值,如前面创建数组中的(3)和(5),也可以分别为数组元素赋值,例如:

```
var colors = new Array(4);          //创建长度为 3 的空数组
colors[0] = "red";                  //为数组第一个元素赋值
colors[1] = "yellow";               //为数组第二个元素赋值
colors[2] = "blue";                 //为数组第三个元素赋值
colors[3] = "green";                //为数组第四个元素赋值
```

在 JavaScript 中,可以将不同数据类型的值存放在一个数组中,例如:

```
var person =["张平",20,"男"];
```

### 3.3.2 数组的访问

可以通过数组的名称和下标直接访问数组元素,访问数组的语法格式如下:

```
数组名[下标];
```

例如,colors[1]表示访问数组中的第 2 个元素,colors 是数组名,1 表示下标,在访问数组元素时,要注意下标值不能越界。

### 3.3.3 数组的常用属性和方法

数组是 JavaScript 中的一个对象,它有一组属性和方法,表 3-4 是数组的常用属性和方法。

**表 3-4 数组的常用属性和方法**

| 名 称 | 描 述 |
|---|---|
| length | 数组的属性,返回数组长度 |
| reverse() | 反转数组。返回反转后的数组,原数组的值也会反转 |
| join() | 用括号内的符号将数组中的所有元素连起来 |
| sort() | 进行升序排序 |
| concat(a1) | 连接数组,将括号中的 a1 接在数组后面,可以连接多个数组 |

**【实例 3-2】** 创建数组,将其输出,重新排序后再次输出,使用"-"将数组元素连接输出。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>数组的应用</title>
    <script type="text/javascript">
        var fruit=new Array(5);
        fruit[0]="Apple";
        fruit[1]="Orange";
        fruit[2]="Banner";
        fruit[3]="Peach";
        fruit[4]="Grape";
        document.write("<h3>输出数组:</h3>");
        for(var i=0;i<fruit.length;i++)
        {
            document.write("<h5>第"+i+"个元素是:"+fruit[i]+"</h5>");
        }
        document.write("<h3>反转后的数组:</h3>");
        fruit.reverse();
        for(var i=0;i<fruit.length;i++)
        {
```