第二篇　对象篇

第 8 章　JavaScript 面向对象基础

学习了前面 7 章的内容，读者已经打下了编写完整的 JavaScript 程序的基础。从本章开始，将学习 JavaScript 应用开发层面的知识，主要包括一些常用对象的使用方法。

这里引入了一个新的概念，即面向对象。虽然在前面已经使用过很多对象，但没有对这个概念进行专门的讲解。面向对象设计方法是早在二三十年前，为了解决“软件危机”而被提出来的。如今很多优秀的设计方法都基于面向对象的设计方法，面向对象设计方法能更好地实现复杂系统的组织和代码复用。

JavaScript 是十几年前才被设计出来的语言，它的一些语言特性支持以面向对象的方法进行系统设计，甚至其内置的功能都是以对象的形式提供的。本章将带领读者学习 JavaScript 面向对象的特性。

- 了解面向对象的基本概念。
- 掌握对象的定义和使用方法。
- 掌握 JavaScript 的对象层次结构。
- 理解掌握事件概念和使用方法。

以上几点是对读者在学习本章内容时所提出的基本要求，也是本章希望能够达到的目的。读者在学习本章内容时可以将其作为学习的参照。

8.1　面向对象概念

早期面向结构的设计方法最经典的一句话是“自顶向下，逐步细化”，说明了设计的一般过程。以数据为中心，以分层的方法组织系统。与面向过程相联系的是一套相关的设计方法论，其中包含了许许多多的概念和术语。

面向对象设计方法被提出以后，随之而来的也是一整套设计方法。一切基于对象，是面向对象的核心，接着引入了很多早期开发者闻所未闻的新概念。学习面向对象设计方法，首先应该引入新的设计理念。这些基本的设计理念都体现在语言特性中，如封装、聚合、继承和多态等。对于 JavaScript 的初级用户，只在需要自定义新类型时能定义自己的类型并将其实例化，使应用的结构因为它而变得更清晰、易于维护即可。

8.1.1　面向对象中的语言

早期的程序员对 Pascal、Basic 等语言可能记忆犹新，它们都是典型的面向过程的语言。面向过程的语言将能完成特定任务的程序段独立起来做成函数，整个系统都是函数相互调用来处理数据，系统以数据流为中心。很显然，系统的各模块间是紧耦合关系。随着系统复杂度的增加，系统变得难以维护，最终实现不了当时硬件对软件规模的要求，于是产生了“软件危机”。当然了，编程语言也难辞其咎。

后来因为面向对象的流行，Pascal 被人们改造成支持面向对象的 Object Pascal。Basic 语言也被改进了，增加了一些面向对象的特性。真正很好地支持面向对象的语言有很早以前就设计出来的 Smalltalk 和 C++，以及后来的 Java 和 C#，除 C++外的三种语言都是纯面向对象的语

言。其他一些流行的脚本语言也支持一些面向对象的特性，但那不是主要目的，如 Python、PHP、JavaScript 等。

面向对象的语言，要求至少能提供以下的功能。

- 封装：此特性可隐藏对象内部的实现细节，对外提供一致的访问接口。
- 聚合：将多个对象组合起来，实现更复杂的功能。
- 继承：简单的代码复用机制，使子类拥有父类的特性。
- 多态：以一致的方式使用不同的实现，实现接口不变性。

以上 4 点是面向对象语言一般特性，读者可能觉得抽象难懂。不懂也可以先将其放下继续学习下面的内容，JavaScript 的用户不需要深入掌握。

8.1.2　对象的构成方式

现实世界中的事物的特性都包含状态和行为两种成分，通过这两种成分即可描述一个事物的客观特征。例如这样来描述一只鱼：红色、重 1kg、长 10cm 且会游动。把上例中的鱼抽象出一个类，其有一些属性和行为方法，属于该类的任何一个对象都是这样拥有属性和行为方法。将前述模型抽象到编程语言中以后，对象也就由属性（数据）和方法（处理数据的方法）组成。

属性描述了对象的状态，方法是对象具有可实施的动作。前面所举的例子中，“色、重、长”都是鱼的属性，“游动”是鱼的行为（即方法）。对象就是以这样的方法来将数据和处理数据的方法包装在一起的。

8.2　对象应用

严格地讲，JavaScript 不是一种面向对象的语言，因为它没有提供面向对象语言所具有的一些明显特征，如继承和多态。因此，JavaScript 设计者把它称为“基于对象”，而不是“面向对象”的语言。在 JavaScript 中仅将相关的特性以对象的形式提供，开发者一般也只需要掌握内置对象的使用方法和构造简单的对象即可。本节将介绍如何使用对象。

8.2.1　详解对象声明和实例化

每一个对象都属于某一个类，类是所有属于该类的对象所具有的属性和方法的抽象描述。例如一条金鱼就是属于鱼类，所以得到一只具体的金鱼前首先明确鱼类。

JavaScript 中没有类的概念，创建一个对象只要定义一个该对象的构造函数并通过它创建对象即可。构造函数的定义方法在第 6 章中已经讲过，使用函数对象的 this 指针可以为函数对象动态添加属性。这里的对象的属性和方法也是通过 this 指针动态添加。

例如，欲创建一个 Card（名片）对象，每个对象又有这些属性：name（名字）、address（地址）、phone（电话）；则在 JavaScript 中可使用自定义对象，下面分步讲解。

（1）定义一个函数来构造新的对象 Card，这个函数称为对象的构造函数。

```
01  function Card( _name, _address, _phone )          // 定义构造函数
02  {
03      this.name=_name;                               // 初始化“名字”属性
04      this.address=_address;                         // 初始化“地址”属性
05      this.phone=_phone;                             // 初始化“电话”属性
06  }
```

this 关键字表示当前对象即由函数创建的那个对象。

（2）在 Card 对象中定义一个 printCard 方法，用于输出卡片上的信息。

```
function printCard()                                    // 打印信息
{
    line1="Name:"+this.name+"<br>\n";                   // 读取 name
    line2="Address:"+this.address+"<br>\n";             // 读取 address
    line3="Phone:"+this.phone+"<br>\n"                  // 读取 phone
    document.writeln(line1,line2,line3);
}
```

（3）修改 Card 对象，在 Card 对象中添加 printCard 函数的引用。

```
function Card(name,address,phone)                       // 构造函数
{
    this.name=name;                                     // 初始化 name、address、phone
    this.address=address;
    this.phone=phone;
    this.printCard=printCard;                           // 创建 printCard 函数的定义
}
```

（4）即实例化一个 Card 对象并使用。

```
Tom=new Card( "Tom", "BeiJingRoad 123", "0851-12355" );  // 创建名片
Tom.printCard();                                         // 输出名片信息
```

上面分步讲解是为了更好地说明一个对象的产生过程，但真正的应用开发则是一气呵成。其中有太多的地方需要运用编程技巧，并灵活设计。将上述几步合成，如下例所示。

【范例 8-1】创建一个卡片对象，卡片上标有“名字”、“地址”和“电话”等信息。名片对象提供一个方法以输出这些信息，如示例代码 8-1 所示。

示例代码 8-1

```
<script language="javascript">                          // 脚本程序开始
function Card( name,address,phone )                     // 构造函数
{
        this.name=name;                                 //初始化名片信息
        this.address=address;
        this.phone=phone;
        this.printCard=function()                       // 创建 printCard 函数的定义
        {
            line1="Name:"+this.name+"<br>\n";           // 输出名片信息
            line2="Address:"+this.address+"<br>\n";     // 读取地址
            line3="Phone:"+this.phone+"<br>\n"          // 读取电话信息
            document.writeln(line1,line2,line3);        // 输出
        }
}
Tom=new Card( "Tom","BeiJingRoad 123","0851-12355" );   // 创建 Tom 的名片
Tom.printCard()                                         // 输出名片信息
</script>                                               <!-- 脚本程序结束 -->
```

【运行结果】打开网页文件运行程序，其结果如图 8-1 所示。

【代码解析】该代码段是声明和实例化一个对象的过程。代码第 2～14 行，定义了一个对象类构造函数 Card（名片）。名片包含三种信息，即三个属性，以及一个方法。第 15、16 行创建一个名片对象并输出其中的信息。