基于 Ajax 的客户端 MVC 模型

黄史
北京邮电大学电子工程学院，北京 （100876）
E-mail：hs8210@gmail.com

摘 要：随着 Ajax 的盛行，Web 客户端的代码也日益复杂。本文首先分析 Ajax 与 MVC 三层模型各自的优势，然后提出在 Ajax 的客户端采用 MVC 三层模型来优化代码结构的思想，最后通过一个例子“BookInfo”给出具体的实现。

关键词：Ajax、JavaScript、MVC 模型

1. Ajax

Ajax(Asynchronous JavaScript and XML)并不是一种新的技术，而是将 Web 开发的几种技术结合在一起，从而使得这些技术一起完成以前不能完成的功能：Web 的异步交互功能。Ajax 涉及到的技术包括：

1. 使用 XHTML 和 CSS 进行标准化的数据呈现
2. 使用 DOM 实现动态的显示和交互
3. 使用 XML，XSLT，JSON，Text 等方式进行数据交换与处理
4. 使用 XMLHttpRequest 进行异步数据的读取
5. 使用 JavaScript 绑定和处理所有数据，实现业务逻辑

以下是传统 Web 交互与 Ajax 交互方式的比较[1]：

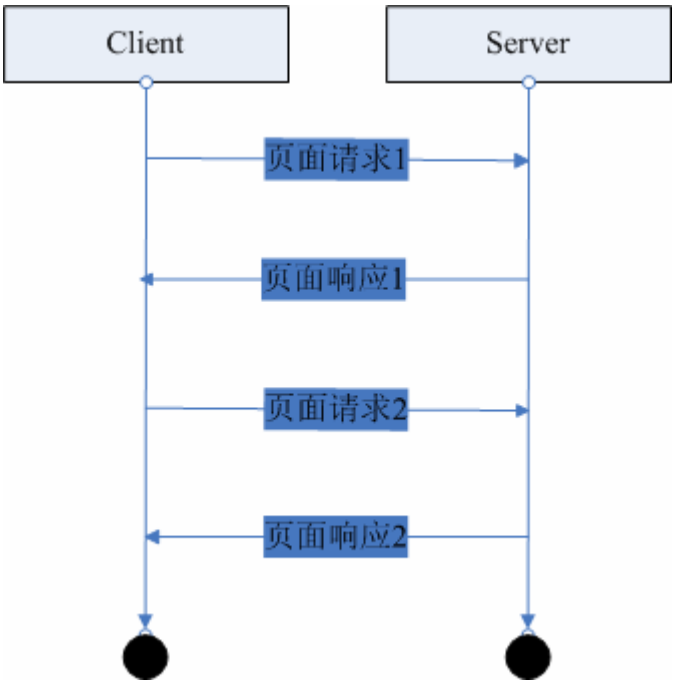

图 1 传统 Web 交互图

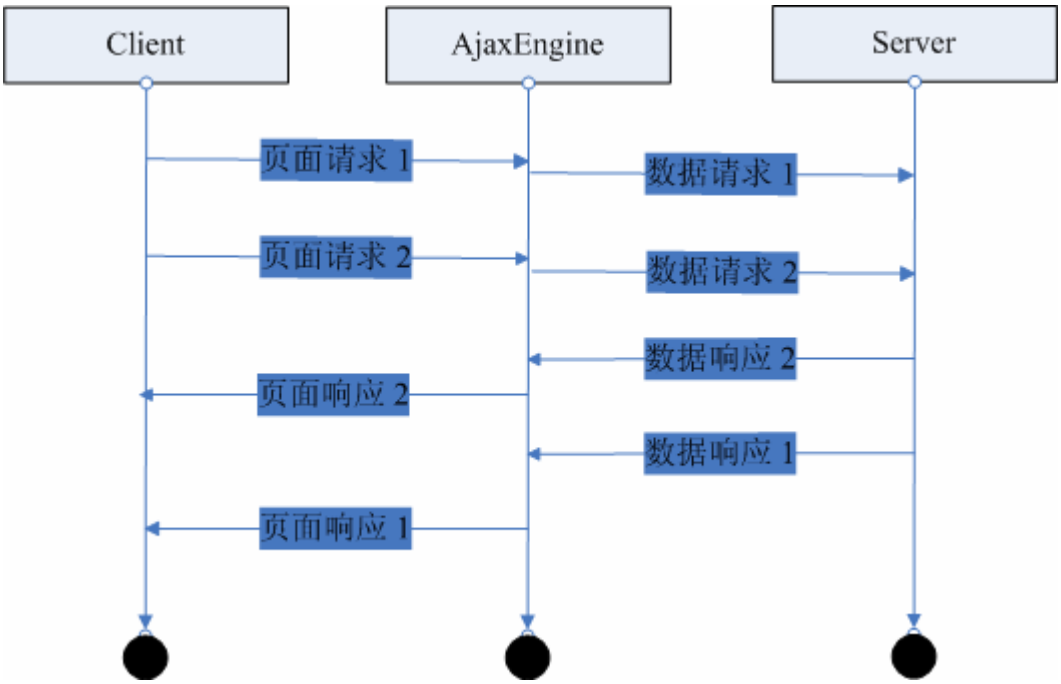

图 2 Ajax Web 交互图

从图 1 可以看出，传统 Web 采用的是同步的消息交互方式，用户触发一个 HTTP 请求到服务器，服务器对其进行处理后再返回一个新的 HTML 页面到客户端，客户端重新显示从服务器得到的结果。当服务器处理客户端请求时，客户端只能空闲等待，哪怕是一次很小的交互，服务器都要返回一个完整的 HTML 页面，极大地浪费了用户等待时间和网络带宽。这种交互方式效率低，用户体验度差，极大消耗了网络带宽。

图 2 中，Ajax 采用的是异步调用的方式，用户向 AjaxEngine 发起请求，AjaxEngine 判断请求是否需要向服务器请求数据，若需要则以后台方式发起数据请求，此时在网络中仅传输交互的数据，而非整个页面，大大节省了网络带宽，同时客户端仍可进行其他操作。服务器响应后 AjaxEngine 会自动调用客户端定义好的回调函数，执行相应的操作。一方面，减小了客户端与服务器之间的数据传输；另一方面，操作之间没有阻塞，大大提高了用户体验度，实现了页面无刷新，提高响应速度，在操作方式上更接近桌面应用程序。Google 所建立的 GMail 和 Google Maps 就是 Ajax 应用最好的例子。

2. MVC(Model-View-Controller)模型

MVC（M：模型，V：视图，C：控制器）模型[2]描述的是将程序与用户交互的部分和完成其他繁重工作、科学计算或业务逻辑等部分很好的一种分离的方式。

1. 模型是指从现实世界中挖掘出来的对象模型，是应用逻辑的反映。模型封装了数据和对数据的操作，是实际进行数据处理和计算的地方。
2. 视图是应用和用户之间的接口，它负责将应用显现给用户和显示模型的状态。
3. 控制器负责视图和模型之间的交互，控制对用户输入进行反应。它主要负责两方面的动作：把用户的请求发送到相应的模型；将模型的改变及时的反映到视图上。

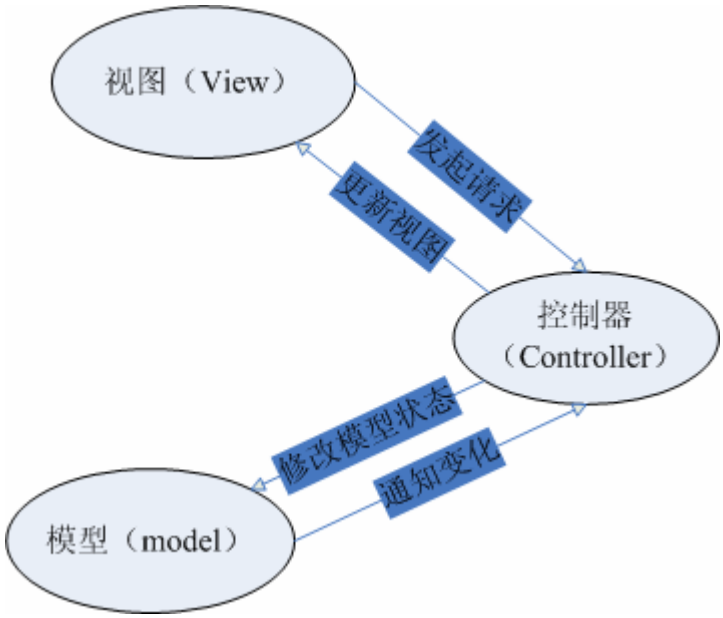

图 3 MVC 三层结构图

MVC 模式将这些对象解耦合，大大提高了程序设计的灵活性和代码的复用性。举个简单的例子：实现一个记录 Button 被点击次数的程序，Button 每被点击一次，页面上就会实时地更新点击总次数。此时，Button 就是视图；数据模型每被调用一次，返回值就加一；而控制器则是视图与模型之间的纽带，它维护 Button 的 OnClick 事件，Button 每被调用一次就调用一次数据模型，并通知视图更新点击的次数。通过控制器，可以将视图和模型进行任意的搭配（例如可以将 Button 换成图片），在获得各种灵活的功能的同时只需要编写少量的代码。如果考察更为复杂的 UI 组件，MVC 的结构设计将使我们获益良多。

3. Ajax 客户端 MVC 模型

下面我们来考虑一下 Ajax 的客户端。由于 Ajax 主要通过 JavaScript 与服务器进行交互，因此客户端需要编写大量的 JavaScript 代码，甚至不可避免的会参与一部分的业务逻辑。

假定客户端采用 Xml 数据格式与服务器进行通信，客户端通过 JavaScript 调用服务器端的函数来发起数据请求，服务器收到请求后，经过业务处理模块获取所需数据，按照定好的协议将结果封装成 Xml 数据返回给客户端。这时可以将整个服务器后台看成是一个数据库，页面看成用户界面，JavaScript 就是界面与数据库之间传递数据和逻辑控制的工具。这样一来，Ajax 的 Web 客户端和传统的桌面应用程序就很类似了，这样采用 MVC 结构对客户端结构进行设计必将使客户端程序具有更多的灵活性和便利性，更好的代码重用性。

下面先介绍一下基于 Ajax 技术的客户端的 MVC 三层结构[3]，然后通过一个例子给出如何在客户端实现 MVC 模型。

3.1 Ajax 客户端的 MVC 结构

3.1.1 模型

传统的 Web 应用，所有的模型都位于服务器端。而在 Ajax 应用中，模型可以分布在客户端和服务器端，以便客户端在向服务器端提交请求之前能根据情况做一些预处理的工作，如果这些工作需要了解业务领域的事情，那么在客户端创建一个业务模型可以使客户端能够做更多的事情，减轻服务器端的负载，这无疑是 Ajax 技术的一大优势所在。

3.1.2 视图

保持视图与逻辑的分离，页面设计师通过 Css（样式表）来控制页面的布局。

3.1.3 控制器

MVC 中控制器的角色是作为模型和视图之间的仲裁者将两者解耦。在 Ajax 客户端应用中，控制器层由事件处理函数组成，通过回调函数通知视图进行更新。

3.2 BookInfo 的 MVC 实现

下面将以一个查看书籍详细信息的例子 BookInfo 来阐述如何在 Ajax 的客户端实现 MVC 结构。BookInfo 的功能很简单，页面显示书籍封面图片，当我们把鼠标放在封面的图片上的时候，旁边将显示书籍的详细信息；当鼠标离开封面图片的时候详细信息将消失。

首先，定义书本详细信息的 Xml 数据格式：

```
<bookinfo>
<id>书本 id</id>
<name>书名</name>
<author>作者</author>
<publish_date>出版日期</publish_date>
<publish_company>出版社</publish_company>
<price>价格</price>
<detai_url>详细信息链接</detail_url>
</bookinfo>
```

3.2.1 BookInfo 的模型

模型层通过解析服务器返回的 Xml 数据构建相应的模型，为了简单，构建如下模型：

function Book(id, name, author, publish_date, publish_company, price, detail_url)

在模型中，我们将解析到的书本详细信息作为 Book 对象的属性值。

3.2.2 BookInfo 的视图

视图层也相当的简单，“ book1”和“ book2”放置书本的封面图片，“bookinfo_detail”负责显示书本的详细信息，具体代码段如下：

```
<div>
    <div id='book1' class="bookImage"><img src="image/mcsc.gif"/></div>
    <div id='book2' class="bookImage"><img src="image/autoCAD.gif"></div>
</div>
<div id='bookinfo_detail'>
```

3.2.3 BookInfo 的控制器

控制器主要由两部分组成：事件选择器和对象查看器。其中事件选择器负责接受视图对象所要监听的事件和相应的回调函数。它实现了观察者模式（Observer），为每个注册事件维护一个监听函数列表，这样就可以管理多个事件回调。事件选择器的主要方法如下：

```
//注册视图节点和事件类型
DomEvent.EventSelector=function(eDom, eventType,)
//添加需要监听的事件回调函数
```

```
DomEvent.EventSelector.prototype.addListener=function(listener)
//删除监听的回调函数
DomEvent.EventSelector.prototype.removeListener=function(listener)
//执行所有监听的回调函数
DomEvent.EventSelector.prototype.excute=function(e)
//视图节点对应的回调方法，负责调用 excute 来执行回调函数
DomEvent.EventSelector.callback=function(event)
```

对象查看器负责遍历 Book 对象的所有属性，并把所有属性名和属性值装配成 Html 的 table 来显示。这种由控制器直接生成视图层代码的方式，在一定程度上减轻了 UI 手工编码的工作量。对象查看器的主要方法如下：

```
//指定所要查看的对象和视图层用作显示的 div
viewer.BookViewer=function(obj, div)
//通过对象的属性名称取出属性值，并装配到表格中
viewer.BookPropertyViewer=function(bookViewer, name)
//创建一个 Text 型的 div 来显示某一个属性值
viewer.BookPropertyViewer.prototype.createTextDiv=function()
//清空视图层用作显示的 div
viewer.RemoveBookViewer=function(div)
```

3.2.4 BookInfo 的 MVC 流程

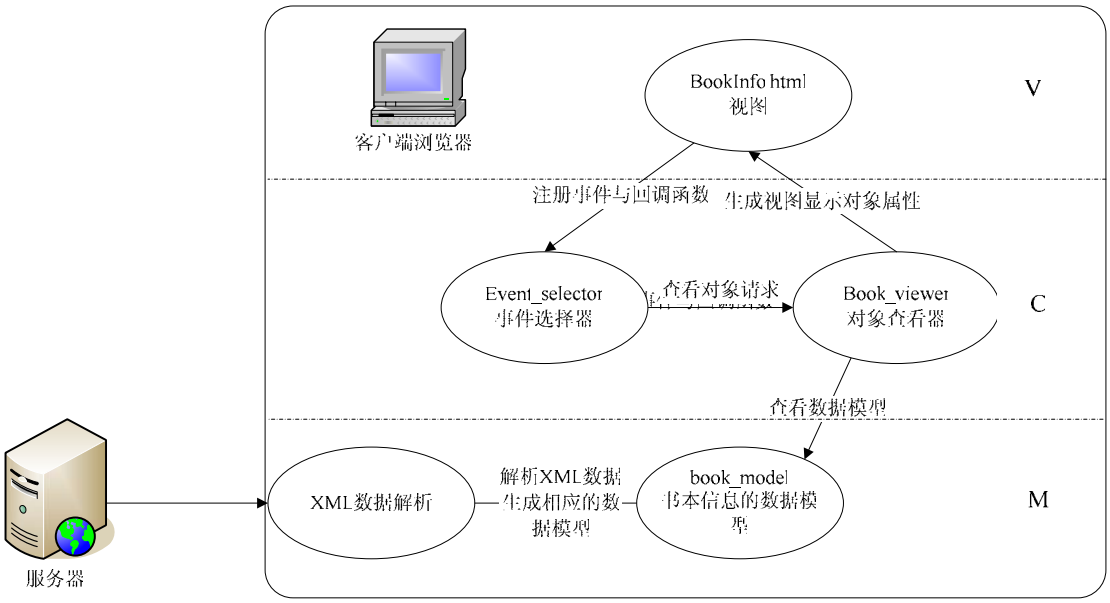

图 4 BookInfo 的 MVC 结构图

下面我们看控制器是如何把视图层和模型关联起来的。如图 4 所示，首先，通过解析服务器返回的 Xml 数据得到了书本详细信息，并通过 Book 模型 book_model 构造模型实例，并在构造函数中将实例放入一个全局数组 Books 中：

```
var book1=new Book(0, "MCSE：Windows 2000 Directory Services Design Exam Notes",
                   "Robert King Gary Govanus", "2000-11-1", "电子工业出版社", "￥30.00",
                   "http://www.china-pub.com/computers/common/info.asp?id=1588");
```

视图层通过事件选择器注册事件并添加需要监听的回调函数：

```
var b1 = document.getElementById('book1');
```

```
//为 b1 注册“onmouseover”事件，并监听显示书本详细信息的 showDetail 回调函数
var eSel1 = new DomEvent.EventSelector(b1, "onmouseover");
eSel1.addListener(showDetail);
//为 b1 注册“onmouseout”事件，并监听清除书本详细信息的 removeDetail 回调函数
var eDel1 = new DomEvent.EventSelector(b1, "onmouseout");
eDel1.addListener(removeDetail);
```

其中，showDetail 函数从模型的全局数组中获取模型实例，通过对象查看器生成视图层代码进行显示：

```
function showDetail(e)
{
    var book_div = document.getElementById('bookinfo_detail');
    viewer.BookViewer(Books[0], book_div);
}
```

RemoveDetail 函数则负责清除显示对象属性的视图层：

```
viewer.RemoveBookViewer=function(div)
{
    while(div.firstChild!=null)
        div.removeChild(div.firstChild);
}
```

最终的效果图如下所示：

图 5 BookInfo 效果图

由上述例子可见，采用客户端的 MVC 三层结构设计，使得视图层与业务逻辑解耦。整个视图页面都没有出现类似“onclick＝xxx()”的代码，页面设计师不必理会每个页面节点将会关联什么事件，执行什么函数，从而可以自由的设计页面布局。事件回调函数的编写人员只需要简单的为页面节点注册相应的事件就能将视图与模型连接起来，并且可以直接调用控制器生成显示模型的代码，减少手工编写页面代码量。

4. 结束语

本文采用了 MVC 结构对 Ajax 客户端代码进行重构，使得代码层次分明，特别是在复杂的 Web 应用中将能发挥更大的作用。随着 Ajax 技术应用越来越广泛，客户端的代码将会日益增多，也更加复杂。采用模式设计将使客户端的代码更容易维护，相信在不久的将来会有更多原本在服务器端进行的模式设计运用在客户端结构设计中。

参考文献

[1] dragonson ．Ajax 程序设计入门．http://dev.yesky.com/284/2043284.shtml．2005.07.25

[2] 王家骐，于海霞．基于 MVC 设计模式的 WEB 应用框架研究．http://dev.yesky.com/143/2689643.shtml．2006.11.28

[3] Dava Crane，Eric Pascarello，Darren James．Ajax In Action ．Manning Publications Co．2006

MVC Pattern on Ajax Web Client

Huangshi

(School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China)

Abstract

Coding task in web client is getting more and more complicated when Ajax technology goes popular in web applications. In this paper, we analyze the advantages of Ajax technology and MVC pattern, and introduce how to apply MVC pattern to the design of Ajax Web client to make the coding task more effective. An example named “BookInfo” will be shown at last to demonstrate the advantage of MVC pattern using in Ajax Web client.

Keywords： Ajax, JavaScript, MVC pattern