基于 Ajax 的 PHP 框架构建

李 昕[1,2]

(1. 中南大学 信息科学与工程学院,湖南 长沙 410083;
2. 湖南财经高等专科学校 信息管理系,湖南 长沙 410205)

摘 要:Ajax 框架是多种现有技术的综合,它有效减少了 B/S 模式下无效的网络流量,大大改善了浏览器端的用户体验,已经成为动态 Web 页面技术的重要工作模式。通过两个基于 PHP 的 Ajax 框架搭建实例,表现了 Sajax,XOAD 这两种主流的 Ajax 框架的基本构建过程,并比较了其性能特征。实例表明 Ajax 框架的多个构建过程均是可行和简洁的。

关键词:PHP;Ajax;Sajax;XOAD

中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2007)08-0123-03

Building of PHP Framework with Ajax

LI Xin[1,2]

(1. College of Information Science and Engineering, Central South University, Changsha 410083, China;
2. Dept. of Information Management, Hunan Advanced Finance & Economy College, Changsha 410205, China)

Abstract: The Ajax framework is the combination of several current techniques. It reduces the redundant network flux, and improves the browser user's experience. This article gives two examples of Ajax framework building with PHP, and describes the main building process of two popular Ajax framework: Sajax and XOAD, and compares their characteristics. The examples of this paper show that these buildings of Ajax framework are feasible and compact.

Key words: PHP; Ajax; Sajax; XOAD

0 引 言

传统的动态页面技术一直是开发动态 Web 应用的主流技术。用户在向服务器提交 HTTP 请求后,往往需要等待较长时间更新页面,大量无须更新的内容也被迫同时刷新。近几年才被广泛应用的 Ajax 技术将传统的动态网页技术隐藏到了后台,动态程序反馈的结果被直接显示在静态页面上,而不再需要整个页面的刷新,B/S 模式的用户体验到了类似 C/S 模式客户端的显示效果。

Ajax(Asynchronous JavaScript and XML)其实是多种现有技术的综合,包括 JavaScript,XML 和 XSTL,XHTML 和 CSS,DOM,XMLHttpRequest 等,其核心的理念是在传统的浏览器和服务器的直接交互中,增加一个中间 Script 层,原来的 Browser - Server 架构变成了 Browser - Script - Server。浏览器只向 Script 层发送消息,Script 层使用 XMLHttp 向服务器发送请求,然后服务器在 XMLHttp 的回复中带上相关消息,最后使用 JavaScript 绑定和处理所有数据,使用 HTML DOM 模型处理界面。中间 Script 层又被称为 Ajax 引擎,它在应用中只和服务器交换有用的数据,而页面显示等不必要的数据不再重新加载,这使得应用过程更自然,操作更流畅[1]。

PHP 因其免费、开源等优点受到了许多 Web 开发人员的支持,得到了广泛的实际应用[2]。许多采用 Ajax 技术的 PHP 框架提供了丰富的、高质量的功能,其中较为流行的有 Sajax 和 XOAD。Sajax 是最早的 Ajax 化 Web 框架,它最初就是用 PHP 编写的,可以视为 Ajax 在 PHP 上的简化实现;XOAD 是一个比较完整的 Ajax + PHP 框架,原来的名称是 NAJAX,现在是 sourceforge 上的一个开源项目。文中正是通过简单实例来展现这两种框架的构建过程,并对比其性能特征。

1 Sajax 框架的构建分析

Sajax 是"Simple Ajax Toolkit"的简写,也就是"简单 Ajax 工具包",其最大特点就是简单,还可以支持 ASP,Cold Fusion,Perl,PHP,Python,Ruby,IO,LUA 等

收稿日期:2006-10-31
基金项目:湖南省教育科学"十一五"规划课题(2006XJ150)
作者简介:李 昕(1969-),女,副教授,硕士研究生,从事计算机网络、电子商务和数据库技术的教学与研究。

八种语言[3]。这种框架通过调用 Sajax Bridge 来回调服务端代码,从而产生客户端 JavaScript 函数,实现了 Web 的 UI 与服务端函数的绑定。Sajax 是一个 ORB 风格的远程调用层,它可以先根据需要编写服务端函数,然后再将 HTML 的 UI 直接绑定到这些函数上[4]。

Sajax 框架的编程主要涉及到三部分内容:客户端、中间层、服务器端[3]。Sajax 技术的重要支持站点(http://www.modernmethod.com/sajax/download.phtml)提供了例程压缩包下载,其中既有关键文件 Sajax.php,也有诸多实例。下面就以一个较简单的基于 PHP 的乘法计算的例子来展示 Sajax 框架的基本构建过程,客户端采用 HTML,中间层使用 JavaScript,服务器端使用 PHP。

1.1 构建后台 PHP 函数

以下这段 PHP 代码就是处理乘法运算,写成了一个简单的函数。

```
function multiply ( $x, $y)
{ return $x * $y; }
```

1.2 客户端显示的 HTML 编程

完成一个乘法运算,客户端当然要在页面中显示几个 Input,让用户输入数字,因此编写出以下 HTML 代码:

```
<input type="text" name="x" id="x" value="2" size="3" /> *
<input type="text" name="y" id="y" value="3" size="3" />
=<input type="text" name="z" id="z" value="" size="3" />
<input type="button" name="check" value="Calculate"
onclick="do_multiply (); return false ;" />
```

本段代码中有 3 个 text 类型的 input 项,其中 x,y 作为参与乘法的 2 个数,z 为乘法的积,并用来显示运算结果。而在第 4 个 button 项中调用了一个用户定义的函数 do_multiply(),其函数名与在第 1.1 节中定义的 PHP 函数名有些类似,区别是在其名称前面加了一个"do_"前缀。

1.3 JavaScript 的函数

为了客户端页面中 Input 输入的结果能够提交后台定义的 PHP 函数,中间层要编写一段 JavaScript 代码,实现第 1.2 节中出现的函数 do_multiply()。

```
function do_multiply() {var x, y;
x = document.getElementById("x").value; //获取 X 的值
y = document.getElementById("y").value; //获取 Y 的值
x_multiply(x, y, do_multiply_cb);
}
```

x,y 这两个参数是将会传递给 PHP 函数的变量,对应 PHP 中 multiply 函数定义中的参数 $x 和 $y。在此函数中,获取 Input 输入的结果后执行了一个 x_multiply()函数,这个没有定义的 JavaScript 函数就是在 Sajax 框架中将用 sajax_export()定义,并由 sajax_show_javascript()输出的函数,其名称就是在 sajax_export()中定义的函数名前面加"x_"。此函数的参数应该和第 1.1 节中定义的 PHP 函数的参数保持一致,最后附加的 do_multiply_cb 为 Ajax 在调用了 PHP 的函数后要执行的 JavaScript 函数,此函数的定义如下:

```
function do_multiply_cb(z) {document.getElementById("z").value=z;}
```

此函数有一个参数,返回值直接放入 Z 中,用以显示返回的结果。

1.4 Sajax 框架的构建完成

三个主要的子部分都完成后,把它们像搭积木一样拼接起来,就完成了一个非常简单而典型的 Sajax 框架。

```
<?
require("Sajax.php");
[第 1.1 节中的 PHP 函数]
sajax_init(); //Sajax 初始化函数,声明本页面将使用 Sajax
sajax_export("multiply"); //声明需要用 Ajax 生成的函数
sajax_handle_client_request(); //处理客户端需求函数
? >
<script>
<?
sajax_show_javascript(); //显示 Sajax 生成的 JavaScript 代码
? >
[第 1.3 节中的 JavaScript 函数]
</script>
<body>
[第 1.2 节中的 HTML 代码]
</body>
```

本段代码出现了几个重要的 Sajax 函数:sajax_init()、sajax_export()、sajax_handle_client_request()、sajax_show_javascript()。其中 sajax_export()是 Sajax 用于定义将要输出的 JavaScript 函数名称;sajax_handle_client_request()则要完成三方面的工作[4]:

* 生成客户端 Ajax 调用。
* 在服务端监听来自所生成方法的回调。
* 基于客户端的回调来调用原来的服务端的方法(带参数),然后返回结果。

sajax_show_javascript()是利用 Ajax 框架提供的桥接代码(bridge code),靠这一行 PHP 代码就可以生成和嵌入了几十行由 Sajax 框架生成的 JavaScript 代码。

Sajax 框架的基本构建过程可以简单小结如下:先将 Sajax 函数定义的 PHP 函数,在 HTML 页面中用 x_函数名的 JavaScript 命令来运行,再确定要传递的参数,并编写得到返回值的 JavaScript 函数,然后在得到返回值的函数中通过 DHTML 处理页面对象,就完全实现了无刷新的 PHP 交互功能。

2　XOAD

XOAD 是一个更新的 PHP 框架[4]。它与 Sajax 类似,都是基于 ORB 风格的远程调用模型,用以实现 JavaScript 层和服务端代码的绑定,但最大的不同是从面向过程变为了面向对象,因此 XOAD 框架不仅产生方法,还要产生 XOAD 对象和类。

2.1　后台系统的修改

这些修改包括把 multiply()函数作为 Multiply 类的静态方法,再把代码封装在 Multiply {} 类中,并在定义前加上关键字 static,使其成为静态函数(即 multiply($x, $y){……})。而且,将要输出的类还要被添加到元数据中。

```
function xoadGetMeta() {
XOAD_Client::mapMethods( $this, array('multiply'));
XOAD_Client::publicMethods( $this, array('multiply'));
}
```

这段代码给出了想要输出的方法映射,同时指定了可以访问的公共方法。通过 XOAD_ Client::privateMethods( ) 可以访问私有方法,而通过 XOAD_ Client::publicMethods 和 XOAD_ Client::privateMethods()可以直接访问变量。这种变化的确违背了面向对象中的封装原则,因此在使用这些方法时要特别注意。

2.2　前台的调整

客户端的调整首先是加载所有必需的东西,并定义好针对 XOAD 的基目录,本例中采用了“xoad”这个子目录。

```
define ('XOAD_BASE','xoad');
require_once ('xoad/xoad.php');
require_once ('Multiply. xoad.php');
XOAD_Server::allowClasses('Multiply');
if (XOAD_Server::runServer()) {
  exit;
}
```

这些能够远程调用的类还应该进行注册。XOAD 中有两种类:allowed 类和 denied 类,调用后者将返回错误结果,而调用前者将会正常执行[4]。一般情况下都会使用 XOAD_Server::allowClasses()来注册用户定义的类,如本例中的 Multiply 类。

XOAD 将 XHR 请求配置为返回到相同的 PHP 服务端页面,由 XOAD_Server::runServer()所登记的方法将用来处理 XHR 请求;如果不支持 XHR 回调则返回 false,否则将依照请求的参数和返回的值来绑定服务端方法。

HTML 部分的代码也必须进行修改。在 HTML 顶部的 head 元素中,要包含所有 XOAD 的 helper JavaScript 代码:

```
<? =XOAD_Utilities::header('xoad') ? >
```

XOAD 完全与类、对象相关,而不仅仅是函数。要注册一个类,并通过 JavaScript 来访问它,只需要另一个助手(helper)函数:

```
var obj = <? = XOAD_Client::register(new Multiply ()) ? >;
```

为了处理可能遇到的错误,还要为每个被调用的方法建立相应的处理器(handler),其名称的标准格式为 obj.on[方法名]Error():

```
obj.onMultiplyError = function(error) {
  document. getElementById("MultiplyError"). innerHTML = "Error:" + error.message;
  return true;
  }
```

XOAD 框架搭建的最后步骤则是将定义的方法再次封装到用户定义的类中:

```
function multiply (x,y) {
obj. multiply (x,y, assignZ);
}
```

至此,一个非常简单的 XOAD 框架就基本搭建完成,当然,还有许多其他功能可以进一步添加实现。

3　结　语

在上文构建的两种 Ajax 框架中,Sajax 框架是面向过程的,使用较为简单,但也存在一些缺点。最主要的缺点是产生的函数只能返回简单类型,完整对象的返回不得不靠创建封装函数、强制类型转换、手动解析等来完成;XOAD 框架则是面向对象的,重点是构建合适的类和对象,能够返回的类型要比 Sajax 丰富,其功能更为全面和强大。当然,这两种框架都实现了隐藏客户端 JavaScript 细节的高度抽象,在 Ajax 代码集成方面都体现了强大功能,因此也成为了两种较为流行的基于 PHP 的 Ajax 框架。

在全球范围内,现有超过 2000 万动态站点使用着 PHP,包括 Yahoo! 等著名网站,目前有超过半数的 Ajax - enabled 和 Web2.0 站点选择应用 PHP[5]。现在

(下转第 129 页)

```
<rdf:range rdf:resource="# 科研成果"/>
</rdf:Property>
<owl: Restriction>
<owl: minCardinality rdf:datatype="xsd:NonNegativeInteger">
1</owl:minCardinality>
</owl:Restriction>
…
</owl: Ontology>
```

从上面例子可以看出在 OWL 中引用了 XML Schema 中定义的数据类型,这是因为已知的数据类已经够丰富,没有必要利用本体语言定义新的数据类型,同时 OWL 还继承了 RDF 中的 Rdfs: subclass, Rdfs: Property, Rdf: subProperty, Rdfs: domain, Rdfs: Range 等一系列原语,并且进行了扩展如例子中 owl:Class, owl: minCardinality 等,从而实现了各词汇之间的精确定义。

4 结　论

分析比较了几种语义描述语言,并举例说明了 OWL 是如何描述信息并体现语义的。通过比较可以得出以下结论:XML 可以交换语法,但并没有语义;RDF 是元数据描述框架,在要求有一定语义和推理能力的情况下使用;OWL 是一种表达能力很强的本体语言,结合了 XML 和 RDF 的优点,可以用于语义要求比较精确的情况。

参考文献:

[1] Berners-Lee T, Handler J, Lassila O. The Semantic Web[J]. The Scientific American, 2001, 284(5): 34-43.

[2] Jena B M. A Semantic Web Toolkit[J]. IEEE Internet Computing, 2002(11, 12): 55-59.

[3] Berners-Lee T. XML and the Web[EB/OL]. 2000-09-06. http://www.w3/org/2000/Talks/0906-xmlweb-tlb/slide9-6.html.

[4] Brickley D, Guha R V. RDF vocabulary description language 1.0: RDF Schema[EB/OL]. 2002. http://Web.w3.org/TR/rdf-schema/.

[5] McGuinness D L, Harmelen F V. OWL Web ontology language overview[EB/OL]. 2004. http://Web.w3.org/TR/owl-features.

[6] W3C. Resource Description Framework Model and Syntax Specification[EB/OL]. 1999. http://www.w3.org/TR/REC-rdf-syntax/.

[7] Gil Y, Ratnakar V. Markup Languages: Comparison and Examples[EB/OL]. 2001-09-07. http://trellis.semanticweb.org/expect/web/semanticweb/comparison.html.

(上接第 122 页)

内)。但也存在缺点。例如,如果某个用户的几个请求消息有关联,一旦某个服务器在处理完前一个或几个消息后就不能正常工作,这样用户的后几个关联消息就会发送到其它的服务器上,这样消息处理就会出错。此外方案也只是采用简单的均衡,不能实时地根据各个服务器的实际负载情况加以调整。

参考文献:

[1] 陈　涛,陈启买. 分布式计算机系统负载平衡研究[J]. 计算机技术与发展,2006,16(9):33-34.

[2] 肖辽亮. NAT-PT 簇负载平衡的设计与实现[J]. 计算机技术与发展,2006,16(3):80-81.

[3] 张洪武. 服务器集群与均衡技术研究[D]. 重庆:重庆大学,2004.

[4] 修长虹. 基于 Linux PC 集群负载均衡的研究与实现[D]. 长春:吉林大学,2005.

[5] 尹康凯,王明伟,李善平. 高可用性集群中多个节点的心跳模型研究[J]. 计算机工程,2005,31(15):102-103.

[6] 胡　宁,刘亚萍. 高性能路由器中 ARP 协议关键问题的研究[J]. 计算机工程与科学,2006,28(10):29-30.

(上接第 125 页)

越来越多的人开始使用 Ajax 技术来开发 Web 应用。在应用 Ajax 开发上,Google 公司当仁不让,Gmail, GoogleGroups, Google Maps, Google Suggest 都应用了这项技术,Amazon 的 A9.com 搜索引擎也使用了类似的技术,微软公司也积极地推出了专门的 Ajax 工具——Atlas。通过与众多主流 Web 开发语言的充分结合,又得到了各大 IT 企业的大力支持,Ajax 技术将有光明的前景。

参考文献:

[1] 方　俊. Ajax 引擎的设计和应用[J]. 电脑与信息技术,2006(3):25-29.

[2] 苑　璟,曹耀钦. 基于 PHP 技术的网络办公自动化系统[J]. 微机发展,2003,13(8):61-63.

[3] Flyinghail. Sajax 实例分析[EB/OL]. 2006-10-22. http://www.flyinghail.net/?p=6.

[4] Gehtland J. Ajax 修炼之道[M]. 徐　锋译. 北京:电子工业出版社,2006.

[5] 方　舟. PHP 技术发展迅猛[EB/OL]. 2006. http://soft.yesky.com/info/154/2556154.shtml.