PHP5 中 XML 解析的应用改进

李 昕[1,2],陈志刚[1]
(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:随着网络的普及,XML 在电子商务和数据交换中扮演了越来越重要的角色,而 PHP 则一直在动态网页的设计中得到广泛应用,两者的结合具有重要的意义。探讨了脚本语言 PHP 中对 XML 解析应用的支持,分析了 PHP 的 XML 解析器 Expat 的工作过程,同时提出直接利用扩展 DOM 类来完成 XML 文档操作。文中的实例结果也表现了 PHP 对 XML 应用支持的灵活性。

关键词:PHP;XML 解析;Expat;SAX;文档对象模型
中图分类号:TP312 **文献标识码**:A **文章编号**:1673-629X(2007)07-0057-03

Improvement of XML Parse Application in PHP5

LI Xin[1,2], CHEN Zhi-gang[1]
(1. College of Information Science and Engineering, Central South University, Changsha 410083, China;
2. Department of Information Management, Hunan Advanced Finance & Economy College, Changsha 410205, China)

Abstract: With the popularization of Internet, XML acts as a more and more important role on E-commerce and data exchange. However, PHP has been used widely in dynamic Web-page design in a long time. Their combination becomes significant now. Discussed the XML support in PHP, and analyzed the working of Expat-an XML parser in PHP. Then put forward the using of expand DOM class on XML-parse in PHP5. The example in this paper also shows the flexibility of PHP in XML application support.

Key words: PHP; XML parse; Expat; SAX; DOM

0 引 言

XML(eXtensible Markup Language,可扩展标记语言)是 W3C 为了适应 Internet 的发展,实现快速的电子商务和电子数据交换而推出的新型 Web 语言。它具有自描述性、数据结构高度规范、良好的扩展性,以及其数据与平台无关性等诸多优点,已在互联网世界被广泛接受和使用,成为网络数据交换的主要标准。而 PHP 作为一种得到广泛应用的 Web 设计的服务器端脚本语言[1],不但一直支持 XML,而且这种支持不断得到了加强。

早期的 PHP 版本就已经开始支持 XML,包含了一个基于 SAX 的接口用于解析 XML 文档;此后 PHP4 添加了 DOMXML 扩展模块和 XSLT,XML 得到了更好的支持[2]。在 PHP4 阶段后期,其它一些功能如 HTML 和 DTD 验证也被加到了 DOMXML 扩展中。但由于这些扩展始终处于不断修改中,它们不能以默认方式安装,也存在不少需要修复的问题,因此只有 SAX 扩展可被默认方式安装,其它一些扩展未得到广泛的使用。

在 PHP5 中,所有支持 XML 的部分几乎全部重新编写。PHP5 的所有 XML 扩展都是基于 GNOME 项目的 LIBXML2 库,允许不同的扩展模块之间互相操作,开发者可以在同一个底层的库上进行开发。除了继承 SAX 解析器,PHP5 还支持遵循 W3C 标准的 DOM 和基于 LIBXSLT 引擎的 XSLT,同时还加入了 PHP 独有的 SimpleXML,符合标准的 SOAP 扩展。

1 Expat 解析器功能分析

如何有效识别及提取 XML 文档中的数据,是对 XML 文档数据后期处理的前提。这就需对文档进行解析。XML 解析器可以让应用程序访问 XML 文档的结构和内容。Expat 是 PHP 脚本语言内置的 XML 解析器,同时也运用在 Mozilla,Apache 等其它项目中。

收稿日期:2006-10-11
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,硕士研究生,研究方向为计算机应用、电子商务、数据库技术;陈志刚,教授,博导,研究方向为分布式计算。

XML 解析器有两种基本类型：

a. 基于 DOM 的解析器：将整个 XML 文档转换成内存中的 DOM 树型结构，同时提供一个 API 来添加、编辑、移动或删除树中的任意一个元素；

b. 基于 SAX 的解析器：将 XML 文档视为一系列的事件。当一个指定事件发生时，解析器将调用开发者提供的相应函数来处理。

Expat 正是一种基于 SAX 的解析器。这些解析器都有一个 XML 文档的数据集中视图，它关注的是 XML 文档的数据部分，而不是其结构。这些解析器按照头到尾的顺序处理文档，并将类似于元素的开始、元素的结尾、特征数据的开始等等一系列事件通过回调(callback)函数报告给应用程序。不同于基于 DOM 的解析器，基于 SAX 的解析器并不需要描述被解析的 XML 文档的完整树型结构。它提供了更底层的访问，可以更好地利用资源和更快地访问。这种方式的明显优势就是不必将整个 XML 文档都放入内存[2]。

为了提高速度，Expat 同时是一个不检查有效性的解析器，它忽略任何与文档联系的 XML 模式文件如 DTD，但仍然要求被解析文档具有完整的 XML 格式，否则 Expat(和其他符合 XML 标准的解析器一样)将会随着出错信息而停止。正是由于不检查有效性，快速和轻巧的 Exapt 成为了非常适合互联网运行的解析器。

Expat 解析 XML 的基本过程包括五个步骤[3]：

(1)创建 XML 解析器的一个实例；

(2)定义处理触发事件的函数；

(3)定义实际意义的数据处理程序；

(4)打开 XML 文件，读取文件数据并解析数据；

(5)关闭文件释放 XML 解析器。

在解析文档前，基于 SAX(基于事件)的解析器通常要求注册回调函数，以用于特定的事件发生时进行调用。Expat 没有例外事件，表 1 列举了它定义的七个可能事件[4]。

表 1 Expat 的主要 XML 解析函数

对象	XML 解析函数	描述
元素	xml_set_element_handler()	元素的开始和结束
字符数据	xml_set_character_data_handler()	字符数据的开始
外部实体	xml_set_external_entity_ref_handler()	外部实体的出现
未解析外部实体	xml_set_unparsed_entity_decl_handler()	未解析的外部实体出现
处理指令	xml_set_processing_instruction_handler()	处理指令的出现
记法声明	xml_set_notation_decl_handler()	记法声明的出现
默认	xml_set_default_handler()	其它未指定处理函数的事件

所有的回调函数必须将解析器的实例作为其第一个参数(此外还有其它参数)。例如，Expat 将三个参数传递给开始元素的处理函数。在脚本范例中，其定义如下：

```
function start_element( $parser, $name, $attrs)
```

其中第一个参数是解析器实例标识，第二个参数是开始元素的名称，第三参数为包含元素所有属性和值的数组。用于产生 XML 解析器实例的函数是 xml_parser_create()。该实例将被用于以后的所有函数。开始解析 XML 文档后，Expat 在遇到任何一个开始元素时，都将调用已被定义好的 start_element()函数，并将参数传递过去。

2 使用 DOM 操纵 XML 文档

DOM(文档对象模型)是由 W3C 制定的一套访问 XML 文档树的标准。在 PHP4 可以使用 DOMXML 来对此进行操作，但 DOMXML 不符合 W3C 标准命名方法，而且还出现过内存泄漏问题。PHP5 中新的 DOM 扩展是基于 W3C 标准完成的，包含方法和属性名称。由于改用了新的标准，基于原来的 DOMXML 的代码将不能直接运行，需要将加载函数和保存函数进行修改，删除函数名中的下划线，以及对其它各处进行必要的调节，但主体逻辑部分可以保持不变[5]。

使用 DOM 来操纵 XML 对象，首先要创建一个 DOMDocument 对象，然后载入 XML 文件。

```
$dom = new DomDocument();
$dom->load("articles.xml");
```

XML 对象被加载到内存中成为 DOM 树后，PHP5 提供了很多方法来直接取得指定名称的元素，最便捷的就是使用 getElementsByTagName($tagname)。下列代码即可遍历所有 TagName 为“title”的元素。

```
$titles= $dom->getElementsByTagName("title");
foreach( $titles as $node) {
  print $node->textContent ."\n";
}
```

其中 textContent 属性虽不是 W3C 标准，但它可以快速方便地读取一个元素的所有文本结点。而遵循 W3C 的标准进行文本结点读取时，代码则应改为：

```
$node->firstChild->data;
```

getElementsByTagName()返回一个 DomNodeList 对象，而在 PHP4 中 get_elements_by_tagname()则返回一个数组，但该 DomNodeList 对象仍可以使用 foreach 语句进行遍历，也可以直接使用 $titles->item(0)来访问结点。

在 DOM 树中，另一个更复杂、更灵活的取得所有指定元素的办法是从根结点遍历。

```
foreach ( $ dom - >documentElement - >childNodes as  $ articles)
{
  if ( $ articles - >nodeType = =1  &&  $ articles - >nodeName
= ="item") {
    foreach ( $ articles - >childNodes as  $ item){
      if ( $ item - >nodeType = =1  &&  $ item - >nodeName
= ="title") {
        print  $ item - >textContent. " \ n"; //对所有 item -
>title 元素进行操作
      }
    }
  }
}
```

使用 DOM 处理 XML 对象除了能读取和查询,同样也可以进行增删和改写操作[4]。下面代码就是在 channels. xml 文件中添加了一个新元素。

```
$ item= $ dom - >createElement("item");
$ title= $ dom - >createElement("title");
$ titleText= $ dom - >createTextNode("new title text");
$ title - >appendChild( $ titleText);
$ item - >appendChild( $ title);
$ dom - > documentElement - > getElementsByTagName
("channel") - >item(0) - >appendChild( $ item);
```

这段代码首先创建了三级层次元素或结点(包括一个 item 元素,一个 title 元素和一个包含 item 标题的文本结点),然后将所有的结点按层次结构链接起来,把文本结点 titletext 加到 title 元素上,把 title 元素加到 item 元素上,最后把 item 元素插入到 channel 根元素上,在 DOM 树上添加一个结点的操作就完成了。

而从 DOM 中删除一个结点则更简单了:

```
$ dom - > documentElement - > RemoveChild( $ dom - >
documentElement - > getElementsByTagName ("channel") - >
item(0));
```

在 PHP5 中使用扩展 DOM 类的新特性,可以书写可读性更强的代码[5]。下面是用 DOMDocument 类重新编写的添加结点的代码例子:

```
class Articles extends DomDocument {
  ……
  function addArticle( $ title) {
    $ item= $ this - >createElement("item");
    $ titlespace= $ this - >createElement("title");
    $ titletext= $ this - >createTextNode( $ title);
    $ titlespace - >appendChild( $ titletext);
    $ item - >appendChild( $ titlespace);
    $ this - >documentElement - >appendChild( $ item);
  }
}
$ dom=new Articles();
$ dom - >load("articles. xml");
$ dom - >addArticle("XML in PHP5");
```

3　小　结

在对 XML 文档的解析上,采用 Expat 或扩展 DOM 类将分别获得来自 SAX 和 DOM 的优势。前者适于快速、顺序、大量地处理 XML 数据对象,主要是速度和内存耗用上的优势;后者适于对 XML 数据对象方便地进行各种复杂的加工操作,主要体现了操作便捷的优势。

PHP 对 XML 的支持遵循了 W3C 标准,功能强大,互用协作性强,已成为可授权使用的默认安装选项。PHP5 新加入的 SimpleXML 扩展提供了更加简单快速访问 XML 文档的方法,可节省很多的代码;PHP5 XML 扩展所使用的底层库 LIBXML2 库同时还支持了使用 DTD, RelaxNG 或 XML Schema 验证 XML 文档。

参考文献:

[1] 苑　璟,曹耀钦,王文海,等. 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.

[2] 聂　丹. XML 在脚本语言 PHP 中的应用[J]. 丹东纺专学报,2005(1):42 - 44.

[3] 刘小东. XML 技术上传文件[J]. 中国 ASP,2003(2):27 - 34.

[4] Ratschiller T. PHP 的 XML 分析函数[EB/OL]. 2006. http://www. ahaoz. com/Article/1/138/427/2005/2005111244856. html.

[5] Stocker C. PHP5 的 XML 新特性[EB/OL]. 2006. http://www. wengu. com/main/Article/website/language/php/200603/10659. shtml.

(上接第 56 页)

参考文献:

[1] He Ligang, Jarvis S A, Spooner D P, et al. Mapping DAG - based Applications to Multiclusters with Background Workload[M]. Warwick: Department of Computer Science, University of Warwick, 2005.

[2] 李敏强,寇纪松,林　丹,等. 遗传算法的基本理论与应用[M]. 北京:科学出版社,2002:64 - 86.

[3] 钟求喜,陈火旺. 基于遗传算法的任务分配与调度[J]. 计算机研究与发展,2000,37(10):1197 - 1203.

[4] 林剑柠. 基于遗传算法的网格资源调度算法[J]. 计算机研究与发展,2004,41(12):2195 - 2199.

[5] 张云锋,葛　玮. An improvement on Algorithm of Grid - Workflow Based on QoS[J]. Wuhan University Journal of Natural Sciences, 2004, 9(5):793 - 797.