微软.NET框架下提取在线Web数据的方法

Method to Extract Online Web Data under Frame of Microsoft's .Net

耿建勇（中国科学院研究生院　100039）
鲁士文（中国科学院计算技术研究所　100080）

摘　要：本文通过一个在线Web数据提取的实例，介绍了在微软的.NET框架下，使用SgmlReader将HTML文件转换为XHTML，并使用XML的XPath语言和XSLT转换技术提取在线Web数据的方法，文中还给出了核心程序的部分代码。

关键词：.NET Web数据 提取

1 引言

互联网是当今世界最大最丰富的信息源，世界各地的很多机构和个人在互联网上发布了大量的信息，互联网上的信息大多是用HTML标记语言描述的，这种结构能够方便地呈现信息，但却很难用程序的方法自动收集信息。本文通过一个应用实例介绍一种在微软的.NET框架下，使用ASP.NET和XML技术自动提取在线Web数据的方法。

2 在线Web数据提取方法

目前的Web网页主要使用HTML标记语言，HTML语言比较擅长网页的布局和外观设置，但缺乏对网页信息内容的表达能力，HTML语言的语法要求也很不严谨，使用程序直接从HTML页面提取数据是非常困难的。随着XML技术的发展，W3C（World Wide Web Consortium）推出了XHTML标准，XHTML是XML的一种应用，它兼容了HTML擅长格式排版方面的特点，又能使用XML的编程方法处理文件中的数据。本文介绍的在线Web数据提取方法就是先将现有的Web页通过转换工具从HTML转换成XHTML，再使用XML的编程方法提取数据。将HTML文件转换为XHTML文件的工具有很多，本文介绍使用的是由Chris Lovett在http://www.gotdotnet.com网站上发布的基于微软.NET框架的SgmlReader。

HTML转换为XHTML后，在数据提取上具有一些优势，HTML文件和转换以后的XHTML文件中的很多内容都是网页格式和排版方面的信息，需要提取的数据通常只集中在文档的一小部分并包含在<table>标记内，比较有利于将注意力集中在数据上，忽略其他无用的信息。数据提取的关键是确定数据在文件中的位置，简单办法就是查看转换以后的XHTML文件，寻找数据位置和包含这些数据的<table>节点的特点[1]。

根据数据位置的特点结合XML的XPath和XSLT技术就能提取我们需要的数据并将其转换为单独的XML文件。XPath是W3C制定的对XML文件进行数据查询的规范，它是一种路径语言，通过XPath表达式说明到达要查询节点的路径。XSLT是W3C推出的转换XML文件的规范，XSLT文件中包含一组模板，XSLT处理器根据文件中的XPath表达式在XML文档中寻找满足条件的节点与文件中的模板进行匹配，它的特点是只处理满足条件的节点，对不满足条件的节点，不进行任何处理。使用这两种技术，就能在XHTML文件中提取我们需要的数据，忽略所有无用的信息。

这种方法的具体实现步骤如下：

（1）确定数据源并使用SgmlReader将其转换为XHTML文件；

（2）确定数据在XHTML文件中的位置并设计提取数据使用的XSLT文件；

（3）使用XSLT转换技术提取数据并将其保存为XML文件；

（4）将每次转换的结果合并。其流程如图1所示。

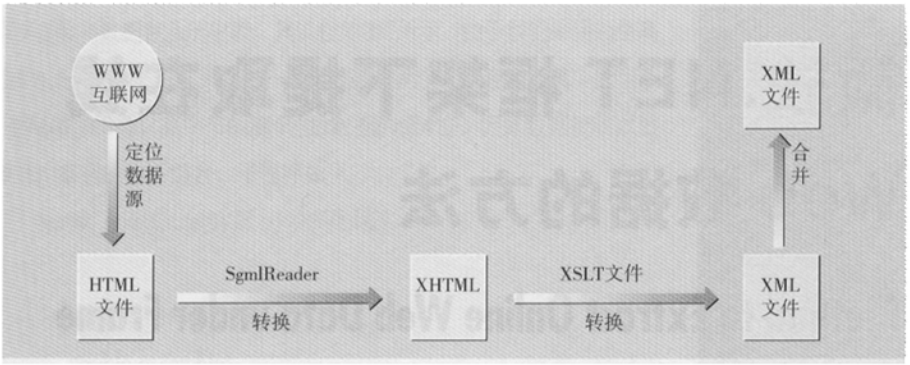

图 1

这种方法对数据源有一定的要求，一般需要数据源能够在网络上长期稳定地存在、能够通过浏览器访问、数据真实并经常更新、页面结构相对稳定等。

3 Web 数据提取的实例

下面通过一个具体的实例介绍这种方法实现的详细步骤，我们将设计一个程序提取在中国银行网站上公布的每日外汇牌价，其网址为：http://www.bank-of-china.com/info/qpindex.shtml，页面显示如图2所示。

图 2

由于该页是框架网页，其包含汇率信息的最终网页地址是：http://www.bank-of-china.com/info/whjrpj.html，我们将提取汇率公布的日期、美元、日元和欧元三种货币的现钞买入价和卖出价等数据，并将最终的结果合并成一个XML文件。程序实现的步骤如下：

3.1 用 SgmlReader 将 HTML 转换为 XHTML

SgmlReader是在微软.NET框架下开发的用来解析SGML文件的工具，也能用来解析HTML文件并将其转换为XHTML文件。SgmlReader是从XmlReader派生出来的，使用它解析HTML文件的用法与使用XmlTextReader解析XML文件是一样的2。在使用SgmlReader之前，必须将其DLL格式封装的文件放入应用程序的bin目录中，并在程序中导入其名字空间Sgml，本例使用VB.NET和ASP.NET开发，需在文件中加入语句<%@Import Namespace="Sgml"%>。SgmlReader还提供了命令行执行的EXE文件的封装格式。SgmlReader输入的文件格式可以是HTML文件或其他SGML文件，文件可以位于本地硬盘上或远程服务器上，还可以是文件流的形式；其输出为XML文件，默认编码格式是UTF-8。关于SgmlReader使用的具体细节和参数请参考SgmlReader的说明文件。

本例中使用SgmlReader将中国银行公布外汇牌价的网页转换为XHTML文件并将结果读入XpathDocument类，以便使用XSLT技术转换。我们直接调用中国银行远程服务器上的网页作为SgmlReader的输入项，其程序代码和说明如55页左上角所示。

3.2 确定数据的位置并设计提取数据使用的 XSLT 文件

使用SgmlReader完成转换XHTML文件后，需要分析数据在文件中的位置，转换以后的XHTML文件在IE中的显示如图3（55页中间所示）所示。

可以看到，文件的根元素是<html>，我们需要提取的时间数据位于/html/body/b/b的节点内，所有的汇率信息都位于/html/body/b/b/table表内，每种货币的信息位于一个<tr>节点内，其中货币名称位于<tr>节点的第一个<td>节点内，我们需提取的现钞买入价和卖出价分别位于<tr>节点的第三和第四个<td>节点的<p>节点内。

明确数据在XHTML文件中的位置后，就能使用XPath和XSLT技术设计提取数据的XSLT文件。其中时间信息在文件中是唯一的，可以直接使用XPath表达式定位，由于在<b>节点内除时间信息以外还包括<table>表格，需使用XPath的substring函数将时间单独提取出来，

```
dim reader as SgmlReader=new SgmlReader()//定义SgmlReader类
reader.DocType= "HTML"//指定SgmlReader类解析的文件类型为HTML
//指定SgmlReader类解析的HTML文件所在的URI地址
reader.Href="http://www.bank-of-china.com/info/whjrpj.html"

dim sw as stringwriter=new stringwriter()//定义StringWriter类
//定义XmlTextWriter类，并将结果写入StringWriter类
dim writer as xmltextwriter=new xmltextwriter(sw)
writer.Formatting=Formatting.Indented//定义缩进格式写入
While (reader.read())//使用Read方法读取HTML文件的节点
//判断HTML的节点是否为空，如不是则将节点写入XmlTextWriter类
if (reader.NodeType<>xmlnodetype.whitespace) then
writer.writenode(reader,true)
End if
End while
//将StringWriter类中的内容转换为字符串并导入XPathDocument类
dim doc as xpathdocument=new xpathdocument(new stringreader(sw.ToString()))
```

```
- <html>
 + <head>
 - <body bgcolor="#ffffff" topmargin="0" leftmargin="0" onload="if(parent.frames.length=="0")
     top.location.href="index.html";">
   - <b>
     <font color="#0000ff">日期：</font>
     - <b>
       2003/06/27
     - <table border="#0000AA">
       + <tr bgcolor="#e0e0e0">
       + <tr bgcolor="#c0c0c0">
       - <tr bgcolor="#e0e0e0">
           <td height="20">美元</td>
         - <td height="20">
             <p align="right">826.5100 nbsp;</p>
           </td>
         - <td height="20">
             <p align="right">821.5400 nbsp;</p>
           </td>
         - <td height="20">
             <p align="right">828.9900 nbsp;</p>
           </td>
         - <td height="20">
             <p align="right">827.7500 nbsp;</p>
```

```
//使用 for-each 遍历<table>元素的每一个<tr>节点
  <xsl: for-each select="//tr">
   <xsl: choose>// 设置条件判断是否是我们需要的货币种类
     <xsl：when test="td[1]"'美元' ">//如果第一个<td>标记内容是美元
      <美元>
       <现钞买入价>
         //<提取美元的现钞买入价
        <xsl:value-of select="normalize-space(substring-before(td[3]/p,'n'))" />
       </现钞买入价>
       <卖出价>
         //<提取美元的卖出价
        <xsl: value-of select="normalize-space(substring-before(td[4] /p, 'n' ))" />
       </卖出价>
      </美元>
     </xsl: when>
    </xsl: choose>
   </xsl: for-each>
```

表达式为substring(body/b/b,1,10)，即提取<b>节点内的第1至10个字符。汇率信息我们只提取美元、日元和欧元三种，由于各种货币的汇率信息都位于<tr>节点内，币种的标志在<tr>节点的第一个<td>节点内，使用XPath表达式直接定位比较困难，需要借助XSLT提供的循环元素for-each和条件检查元素choose、when，具体思路如下：使用for-each元素遍历<table>节点下的每一个<tr>节点，使用choose、when元素设定条件判断其第一个<td>节点内容是否是美元、日元或欧元，如是则提取其第三和第四个<td>节点的<p>节点的内容。由于<P>节点内还包含我们不需要的字符“nbsp”，可以使用XPath语言的substring函数截取字符n前面的内容并使用normalize-space函数去除所有空格，下面以美元为例说明其实现过程（本页左下角所示）。

//使用for-each遍历<table>元素的每一个<tr>节点

3.3 使用 XSLT 技术进行转换并创建 XML 文件

设计完成XSLT文件后，可以进行实际的数据提取操作，微软的.NET框架提供了XslTransform类用来执行XSLT转换操作，XslTransform类先用load方法装入XSLT文件，然后调用Transform方法实施转换，使用的格式如下：Xsl.Transform(XPathDocument类名称, XsltArgumentList, 目标文档路径或流)，其中的XsltArgumentList类用来传递XML文件的参数，本例没有参数，值设为nothing3。转换的过程和程序代码如下：

3.4 合并转换结果

完成XSLT转换后，每天汇率公布的日期、美元、日元和欧元的现钞买入价和卖出价等信息被提取形成一个单独的XML文件，最后需将每天提取的信息合并成一个文件，基本思路如下：建立一个汇总的XML文件，将每天形成的XML文件的根节点作为一个子节点添加到汇总文件。这个过程的完成需要使用XML的

```
Dim xslpath as String=Server.MapPath("xmltrans.xsl")//定义XSLT文件路径
Dim xsldoc as XslTransform=new XslTransform()//定义XslTransform类
xsldoc.Load(xslpath)//导入XSLT文件
//定义文件流在硬盘上创建XML文件
Dim myOut As FileStream=New FileStream(Server.MapPath("extract.xml"),FileMode.
Create)
xsldoc.Transform(doc,nothing,myout)//实施转换
myout.close
```

DOM编程接口技术，DOM是W3C制定的访问和修改XML文件数据的接口标准，它将XML文件读入内存，转换为树型结构，文件中的元素转换为节点，通过对节点的访问、添加与修改实现对数据的相应操作3。本例需要用到XmlDocument类的一些属性和方法，如表1所示：

表1

类名称	属性或方法		使用说明
XmlDocument	DocumentElement	属性	表示XML文档的根节点
	Load	方法	从指定位置装载XML文档
	importnode	方法	从其他XML文件导入XML节点
	appendchild	方法	在当前节点的尾部添加新的子节点
	save	方法	将XML文档保存到指定位置

其实现过程和程序代码如下：

```
dim finaldoc as xmldocument=new xmldocument()//定义XmlDocument类存放汇总文件数据
//使用try-catch结构当汇总的XML文件不存在时通过字符串创建该文件
try
finaldoc.load(server.mappath("finalextract.xml"))
catch
dim hntq as string="<?xml version='1.0' encoding='UTF-8'?><汇率提取></汇率提取>"
finaldoc.loadXml(hntq)
end try

dim newhn as xmldocument=new xmldocument()//定义XmlDocument类
newhn.load(Server.MapPath("extract.xml"))//读入提取每天汇率数据的XML文件
dim root as xmlnode=newhn.documentelement//定义包含每天汇率数据的XML文件的根节点
dim finalroot as xmlnode=finaldoc.documentelement//定义汇总文件的根节点
//将包含当天汇率数据的XML文件的根节点转变为一个汇总XML文件的节点ins
dim ins as xmlnode=finaldoc.importnode(root,True)
finalroot.appendchild(ins)//将ins节点作为子节点导入汇总文件
finaldoc.save(server.mappath("finalextract.xml"))//保存汇总文件
```

3.5 提取 Web 数据的结果

图3显示的是利用上述方法，从6月25日至27日连续三天提取的中国银行公布的美元、日元和欧元三种货币汇率的结果。

```
<?xml version="1.0" encoding="UTF-8" ?>
- <汇率提取>
  + <汇率 提取时间="2003/06/25">
  - <汇率 提取时间="2003/06/26">
top.location.href="index.html";">
    - <美元>
    - <日元>
    - <欧元>
    </汇率>
  - <汇率 提取时间="2003/06/26">
    - <美元>
        <现钞买入价>821.5400</现钞买入价>
        <卖出价>828.9900</卖出价>
        </美元>
      - <日元>
          <现钞买入价>6.8630</现钞买入价>
          <卖出价>6.9496</卖出价>
        </日元>
      - <欧元>
          <现钞买入价>936.2100</现钞买入价>
          <卖出价>947.0800</卖出价>
        </欧元>
        </汇率>
        </汇率提取>
```

图 3

4 结束语

本文介绍了在微软的.NET框架下实现的一种提取在线Web数据的方法，这种方法简单、易行、高效，虽然对数据源有一定的要求，但目前在互联网上符合这些要求的数据源还是很多的。通过选取合适的数据源，正确分析数据在数据源文件中的位置，使用这种方法就能快速建立一个Web数据的提取系统。

参考文献

1 [美] Jared Jackson, Jussi Myllymaki. 基于Web的数据挖掘[EB/OL]. http://www-900.ibm.com

2 [美] Dan Wahlin. Parse HTML Pages to Extract Data[EB/OL]. http://www.ftponline.com

3 [美] Dan Wahlin. 基于XML的ASP.NET开发[M],王宝良译, 清华大学出版社，2002。