**webdav常用方法和概念总结**

参考：<http://blog.csdn.net/mahongming/article/details/1779033>

摘要：本文论述 WebDAV。作为对 HTTP 1.1 的扩展，它已经成为一种重要的 Web 通讯协议。 本文还描述了 WebDAV 的定义，以及它在客户端/服务器结构中的用途。  
  
简介  
随着对 Internet 标准和网络互操作性的日益关注，作为 HTTP 1.1 的扩展，WebDAV（Web 分布式创作和版本控制）已经成为重要的 Web 通讯协议（有关详细信息，请参阅 IETF RFC 2616）。 WebDAV 规范（有关详细信息，请参阅 IETF RFC 2518）在 1999 年 2 月由 Internet 工程任务组 (IETF) 发布，这中间有 Microsoft 的巨大贡献，以及许多第三方供应商（如 Netscape、Xerox、IBM 和 Novell）的支持。  
在 Microsoft，WebDAV 已经创建了许多不同领域的应用程序。 它允许通过 Web 向 Microsoft? Internet 信息服务 (IIS) 5.0 服务器进行丰富的协作式发布。 它是 Microsoft Office 2000 Web 文件夹后面的协议。此技术提供与 Microsoft Exchange 2000 Web 存储系统的 Web 接口，允许通过 Web 直接访问 Exchange 的面向对象的分层数据库。  
由于 WebDAV 与可扩展标记语言 (XML) 固有的集成，因此它不仅非常依赖 XML，而且已经成为通过 Web 传送 XML 数据的绝佳方法。 但是，在完全了解这些技术所带来的好处之前，一定要先了解什么是 WebDAV，以及它在客户端/服务器体系结构中的用途。  
协议问题的解决方案  
由于 Web 已经成为 Internet 的基础，因此 HTTP 1.1（超文本传送协议）被证实是用来传输数据的非常灵活的通用协议。但是，HTTP 存在一些明显的缺点，从而限制了它作为综合的 Internet 通讯协议而被采用：它非常适合用于查看的静态文档，但不能提供以足够复杂（以便向客户端提供丰富的创作功能）的方式来处理文档的方法。  
例如，当两个作者在未进行交流的情况下同时对一个文档进行更改时，就会出现“更新丢失”问题。 只有由最后一个作者完成、并将修改后的文档重新上载到服务器的修改才会保留下来，另一个作者进行的更改将丢失。  
IETF WebDAV 工作组的目标是，设计一个协议，它提供基于标准的论坛中任何分布式创作工具需要的功能。 当前的 WebDAV 规范 (IETF RFC 2518) 解决协作式创作工具的三个主要问题：  
改写保护。HTTP 1.1 无法确保客户端可以保护资源，并且可以在其他客户端同时编辑它们的情况下进行更改。 使用 WebDAV，可以通过多种方式来锁定资源，以便让其他客户端知道您对所讨论的资源感兴趣，或者防止其他客户端访问该资源。       
资源管理。HTTP 只能直接访问单个资源。 WebDAV 提供一种更有效地组织数据的方法。 WebDAV 引入了可包含资源的集合（类似于文件系统文件夹）概念。 通过 WebDAV 进行的资源管理包括如下功能：创建、移动、复制和删除集合，以及集合中的资源或文件。  
文档属性。不同类型的数据具有唯一的属性，这有助于描述数据。例如，在电子邮件中，这些属性可能是发件人的姓名和接收邮件的时间。 在协作文档中，这些属性可能是文档原始作者的姓名和最后一个编辑者的姓名。因为人们使用的文档类型各不相同，所以可能的属性类型列表也变得无限大。 XML 是 WebDAV 所需的一种可扩展通讯工具。  
WebDAV 请求的格式  
HTTP 1.1（请参阅 IETF RFC 2068）提供一组可供客户端与服务器通讯的方法，并指定响应（从服务器返回发出请求的客户端）的格式。 WebDAV 完全采用此规范中的所有方法，扩展其中的一些方法，并引入了其他可提供所描述功能的方法。 WebDAV 中使用的方法包括：  
•     Options、Head 和 Trace。 主要由应用程序用来发现和跟踪服务器支持和网络行为。  
Get。 检索文档。  
• Put 和 Post。 将文档提交到服务器。  
• Delete。 销毁资源或集合。  
•    Mkcol。 创建集合。  
•    PropFind 和 PropPatch。 针对资源和集合检索和设置属性。  
•    Copy 和 Move。 管理命名空间上下文中的集合和资源。  
•    Lock 和 Unlock。 改写保护。  
WebDAV 请求的一般结构遵循 HTTP 的格式并且由以下三个组件构成：  
•     方法。 声明由客户端执行的方法（上面描述的方法）。  
•     标头。 描述有关如何完成此任务的指令。  
•     主体（可选）。 定义用在该指令或其他指令中的数据，用以描述如何完成此方法。  
在主体组件中，XML 成为整个 WebDAV 结构中的关键元素。  
XML 对于 WebDAV 的含义  
WebDAV 旨在服务器上处理资源提供更多方法。 这些附加的方法通常要求大量信息与请求和响应相关联，以便显式定义客户端或服务器的意图。通过 HTTP 传递所有信息的方法只是标头在请求和响应方面的责任。 这会对传输强加一些限制。很难对请求中的多个资源应用标头信息，而且很难描述层次结构。  
因为 XML 本来就具有可扩展性，所以选择它来描述这些指令的通讯方式。 由于 XML 提供如下方法，所以它对于 WebDAV 操作至关重要：  
•     设置指令（描述如何处理数据）格式的方法。  
•     设置来自服务器的复杂响应格式的方法。  
•     传递有关所处理的集合和资源的自定义信息的方法。  
•     数据本身的灵活工具。  
在较高层，WebDAV 指令处理器实际上是一组逻辑，用来解释 WebDAV 方法，它后面跟随一个 XML 分析器，该分析器解释所传递的大部分信息。  
在 WebDAV 中使用 XML 如何将此技术转换为功能如此强大的工具？ 首先，XML 提供一种方法来将数据与作用于该数据的方法或者数据的表示方式分开。 这允许对数据进行直接和一致的提取。 对于这些提取的数据，WebDAV 通过现有网络结构所熟悉的信道，在网络结构中的所有层之间提供一种一致、统一的传输方法， 此技术允许在 Microsoft 产品和第三方应用程序之间进行更高级别的互操作。  
第二，XML 通过一种扩展方法来增强 WebDAV。 XML 允许客户在 WebDAV 服务器上描述和设置属性。 这些属性随后可用于对服务器上的资源进行索引、搜索和处理。 因为 XML 本来就具有可扩展性，所以这些属性的类型和用途是无限的。  
下例（使用 PropPatch 方法）向服务器提交要与该服务器上的每个资源相关联的数据（在本例中为 HTML 文档），然后基于它以前设置的自定义属性来搜索这些文档。 此示例使用成功完成这些任务所必需的原始 WebDAV 请求（通过连网传输的位），然后显示如何使用 MSXML XMLHTTPRequest 对象来创建类似的请求。  
  
用 WebDAV 和 XML 管理文档  
  
设想您需要方便地标识服务器上大量文档中每个文档的作者。 在非 WebDAV 世界中，为了查找由某个人创作的所有文档，您可以通过在这些文档中搜索文本来查找特定作者的姓名。这种搜索还将返回在其中偶然引用该特定作者的所有文档。 如果尝试填充一个包含所有这些文档以及每个文档作者的表，会怎么样？如果单独基于这样的原始文本搜索，这实际上是不可能的。  
  
使用在 XML 中编码的 WebDAV 请求，可以针对集合中的每个文档设置 Author 属性。 此属性随后可用于上面介绍的组织目的。  
  
使用 PropPatch 设置 Author 属性  
  
下面的 WebDAV 请求将针对 MyServer.com 服务器上 WebDavDocs 集合中的 Webdav-xml.htm 文档设置 Author 属性：  
  
PROPPATCH /WebDavDocs/webdav-xml.htm HTTP/1.1  
Host: myserver.com  
Content-Type: text/xml  
Content-Length: 138  
  
<?xml version="1.0">  
<d:propertyupdate xmlns:d="DAV:" xmlns:o="urn:schemas-microsoft-  
com:office:office">  
   <d:set>  
      <d:prop>  
         <o:Author>Sean Purcell</o:Author>  
      </d:prop>  
   </d:set>  
</d:propertyupdate>  
  
此请求的第一行指定客户希望制定的方法 (PropPatch)，并给出要设置其属性的文件的绝对 URL。 此方法的后面三行是标头，用来指定此方法将提交到的服务器，并告诉该服务器预期内容的类型和长度。  
  
XML 编码的主体确切告诉该服务器要设置的属性以及应当赋予它的值。 此 XML 文档中一个重要的发现是使用命名空间声明。 <d:propertyupdate> 元素中的第一个属性定义通过文档来使用 WebDAV 命名空间。 对于此文档中具有该前缀的所有元素，符合 WebDAV 的服务器将知道基于 “DAV:” 架构来应用行为。 在本例中，这些特定属性定义如何针对文档设置属性。  
  
第二个命名空间声明是用于 urn:schemas-microsoft-com:office:office 命名空间的。 在设计 XML 属性时一个最佳策略是彻底检查现有的命名空间，看其中是否存在有用的属性。但是，同样重要的是，要确保现有的属性按照最初打算防止属性冲突的方式来使用。 通过在我们的方案中使用现有的 Office 属性，它将允许其他能够识别此 Microsoft 自定义命名空间的客户端解释该属性。  
  
为了响应此请求，该服务器将发回一个响应，指出该属性已成功设置。  
  
HTTP/1.1 207 Multi-Status  
Server: Microsoft-IIS/5.0  
Date: Wed, 04 Aug 1999 21:52:58 GMT  
Content-Type: text/xml  
Content-Length: 310  
  
<?xml version="1.0"?>  
<a:multistatus xmlns:b="urn:schemas-microsoft-com:office:office"   
xmlns:a="DAV:">  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-xml.htm</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <b:Author/>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
</a:multistatus>  
  
在集合中按作者搜索所有文档  
  
向服务器上 /WebDavDocs 文件夹中的所有资源发出一个类似于上面的 PropPatch，以便该集合中的每个资源都与 Author 属性关联。 现在，为了解决该方案中概述的某个问题，下例检索了填充某个表所必需的信息，该表概述集合中每个文档的作者以及每个文档的名称。  
  
检索此信息的请求如下所示：  
  
PROPFIND /WebDavDocs/ HTTP/1.1  
Depth: 1,noroot  
Host: myserver.com  
Content-Type: text/xml  
Content-Length: 184  
  
<?xml version="1.0"?>  
<d:propfind xmlns:d="DAV:" xmlns:o="urn:schemas-microsoft-com:office:office">  
         <d:prop>  
            <d:displayname/>  
            <o:Author/>  
         </d:prop>  
</d:propfind>  
  
已在该请求中添加了另一个标头，那就是深度标头，它指定应当针对哪些资源应用此方法。 在本例中，值 “1,NOROOT” 指定此方法应当应用于指定 URL 的所有直接子级，而不应当应用于该 URL 本身。  
  
此 XML 请求的主体包含两个要检索的属性： 文档的名称（“DAV:” 命名空间中的属性）；使用 PropPatch 设置的 Author 属性（来自 “office:” 命名空间）。 服务器发送如下响应：  
  
HTTP/1.1 207 Multi-Status  
Server: Microsoft-IIS/5.0  
Date: Wed, 04 Aug 1999 22:38:42 GMT  
Content-Type: text/xml  
  
<?xml version="1.0"?>  
<a:multistatus xmlns:d="urn:schemas-microsoft-com:office:office"   
xmlns:a="DAV:">  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-xml.htm</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <a:displayname>webdav-xml.htm</a:displayname>  
            <d:Author>Sean Purcell</d:Author>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-http-        
requests.htm</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <a:displayname>webdav-http-requests.htm</a:displayname>  
            <d:Author>Sean Purcell</d:Author>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-implementation-  
plan.xls</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <a:displayname>webdav-implementation-plan.xls</a:displayname>  
            <d:Author>Adam Barr</d:Author>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-search.doc</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <a:displayname>dav-search.doc</a:displayname>  
            <d:Author>Laura Jennings</d:Author>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-info.txt</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <a:displayname>webdav-info.txt</a:displayname>  
            </a:prop>  
      </a:propstat>  
      <a:propstat>  
         <a:status>HTTP/1.1 404 Resource Not Found</a:status>  
         <a:prop>  
            <d:Author/>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
</a:multistatus>  
  
此 XML 文档中的每个 <DAV:response> 元素都代表该集合中的一个资源。 这很好地举例说明了 WebDAV 响应如何使用 XML 来表示层次结构以及对于不同资源的不同响应。 在该请求中，对于此集合中的所有项目（最后一个除外，未对其设置 Author 属性）已成功检索了 displayname 和 Author 属性。 最后一个项目的 Author 属性是在 <status> 元素中通讯的，生成的文本为 “HTTP/1.1 404 Resource Not Found”，即，找不到 Author 属性。  
  
此数据可用来填充应用程序中显示所请求数据的表：  
  
webdav-xml.htm  
      
  
Sean Purcell  
  
webdav-http-requests.htm  
      
  
Sean Purcell  
  
webdav-implementation-plan.xls  
      
  
Adam Barr  
  
webdav-search.doc  
      
  
Laura Jennings  
  
webdav-info.txt  
      
  
<author unknown>  
  
按作者搜索文档  
  
在该方案中遇到的最后一个问题是基于 Author 属性搜索大量文档。 IETF DAV 搜索和定位 (DASL) 组是一个已经形成的、旨在扩展 WebDAV 所提供功能的组。 这个组涉及到定义一个可用于搜索 WebDAV 资源的语法。因为这个组的工作还未完成，所以 Exchange 小组已经实现了一个 Search 方法，并将它作为 Exchange 2000 中使用 SQL 语法执行搜索的 WebDAV 服务器组件的一部分。 下例阐释了对集合中由 “Sean Purcell” 创作的所有文档的 WebDAV 搜索请求。  
  
SEARCH /WebDavDocs/ HTTP/1.1  
Host: myserver.com  
Content-Type: text/xml  
Content-Length: 295  
  
<?xml version="1.0"?>  
<g:searchrequest xmlns:g="DAV:">  
   <g:sql>SELECT "DAV:displayname" as prop1,  
      "urn:schemas-microsoft-com:office:office#Author" as prop2  
      FROM SCOPE('SHALLOW TRAVERSAL OF "."')  
      WHERE "prop2" = 'Sean Purcell'  
   </g:sql>  
</g:searchrequest>  
  
此请求的响应返回集合中由 Sean Purcell 创作的所有文档，而且对于其中的每个文档都分别返回 displayname 和 Author 属性，这两个属性由 <prop1> 和 <prop2> 进行标记。  
  
HTTP/1.1 207 Multi-Status  
Server: Microsoft-IIS/5.0  
Date: Wed, 04 Aug 1999 23:56:47 GMT  
Content-Type: text/xml  
  
<?xml version="1.0"?>  
<a:multistatus xmlns:b="urn:uuid:c2f41010-65b3-11d1-a29f-00aa00c14882/"   
xmlns:c="xml:" xmlns:a="DAV:">  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-xml.htm</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <prop1>webdav-xml.htm</prop1>  
            <prop2>Sean Purcell</prop2>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
   <a:response>  
      <a:href>http://myserver.com/WebDavDocs/webdav-http-  
requests.htm</a:href>  
      <a:propstat>  
         <a:status>HTTP/1.1 200 OK</a:status>  
         <a:prop>  
            <prop1>webdav-http-requests.htm</prop1>  
            <prop2>Sean Purcell</prop2>  
         </a:prop>  
      </a:propstat>  
   </a:response>  
</a:multistatus>  
  
使用 XMLHTTPRequest 对象来创建 WebDAV 请求  
  
上面显示的片断代表原始的 WebDAV 请求，它们完成一些非常有用的功能。 问题仍在于如何能以编程方式创建这些请求。 此问题的答案在于 XMLHTTPRequest 对象，该对象是 Msxml.dll 的一部分，它允许程序员创建自定义的 HTTP 请求并读取它们的响应，而不是使用 XMLDOMDocument 对象中内置的 Load 和 Save 方法。 因为 WebDAV 请求与 HTTP 请求的结构完全相同，所以该对象可用于创建任何 WebDAV 请求。  
  
有关 XMLHTTPRequest 对象的详细信息，请参考 XML SDK 文档。  
  
WebDAV 的未来发展  
  
开发人员已在努力扩展 WebDAV，让其实现最初的设想：使协议满足所有分布式创作工具的需要。 IETF 的 DASL 工作组和新组建的 Delta-V（Web 版本控制和配置管理）工作组都在朝这个方向努力。 Microsoft 积极参与这两个工作组。但是，为了提供完善的技术，Microsoft 正在这两个方向上取得自己的进展，这可通过以下事实来证明：向方法库中添加 Exchange 2000 WebDAV 实施所熟悉的 Search 方法。  
将不同的技术进行紧密组合，会同时满足 Microsoft 和第三方产品中许多客户端/服务器技术的要求。  
  
资源  
  
WebDAV.org，位于 http://www.webdav.org/。  
•       
Exchange 2000 官方网站，位于 http://www.microsoft.com/exchange/