ISSN 1009-3044
Computer Knowledge and Technology 电脑知识与技术
Vol.6,No.28, October 2010, pp.7923-7925 ,7935

E-mail: info@cccc.net.cn
http://www.dnzs.net.cn
Tel:+86-551-5690963 5690964

WebSocket 在 Web 实时通信领域的研究

李代立,陈榕

(同济大学 基础软件工程中心,上海 200092)

摘要:分析目前几种 Web 实时事件响应方案的特点,着重研究基于 HTML5 WebSocket 的实时通信机制,并通过 WebSocket 与 AJAX 长轮询的实验对比,展示 WebSocket 在 Web 实时通信领域的应用开发方法和效率,为 Web 实时通信提供低延时、低网络吞吐量的解决方案。

关键词:WebSocket;Web 实时通信;AJAX

中图分类号:TP311 **文献标识码**:A **文章编号**:1009-3044(2010)28-7923-03

The Research of WebSocket Based on Web Real-time Communication

LI Dai-li, CHEN Rong

(System Software Engineering Centre of Tongji University, Shanghai 200092, China)

Abstract: Some schemes and their features are introduced on web real-time communications. More emphasis is put on studying real-time communication mechanism based on HTML5 Web Socket. Then a comparative experiment is made between WebSocket and AJAX to represent the simple method and high efficiency in developing real-time web applications by WebSocket . At last we provide a low latency and low network throughput solution.

Key words: WebSocket; Web real-time communication; AJAX

在 Web 应用高速发展的今天,Web 即时数据通信变得越加强烈。HTTP 协议[1]是 Web 实时通信的基础,浏览器与服务端建立连接浪费了大量时间和网络吞吐量,加重了服务端负担。HTML5 WebSocket 的实现使 Web 应用不需要每次都发起 HTTP 请求来建立与服务端的连接,而是仅在第一次请求连接后就建立起 TCP Socket 连接,从而基本做到通信的时时响应,提高了通信效率。

1 传统 Web 实时通信方案

1.1 HTTP 拉取方式

如图 1 所示,左侧为传统的 web 应用模型,可以通过设置"<meta http-equiv="refresh" c />"标签定时 Request 实现 HTTP Pull,右侧为 AJAX Web 应用模型,可以通过 XMLHttpRequest 定时取服务器端数据实现。

该方案健壮易用的特点使其经常被应用于 Web 系统中,例如,Adaptive TTR[3]机制允许服务器改变 TTR,客户端便能够依据数据的状态改变以不同频率来拉取数据。因此,这种动态 TTR 机制比静态 TTR 模式更好。但它并不能完全保证的数据精确度,会带来不必要的网络流量。

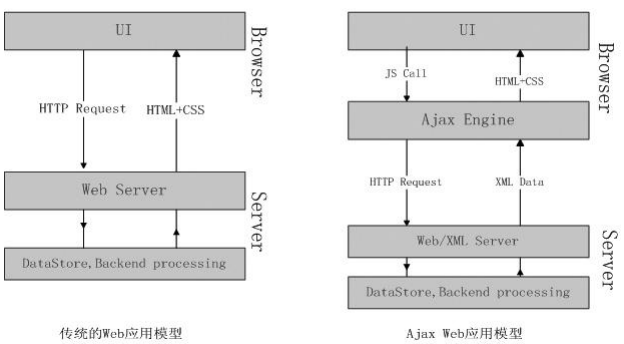

图 1 传统 Web 应用模型与 Ajax Web 应用模型

1.2 HTTP 流

HTTP 流是 1992 年网景公司推出的 push 机制,叫"动态文档"[4]。有两种形式:页面流和服务流。

页面流是指页面上不间断的 HTTP 连接响应。大部分 Web 服务器处理发送完响应之后立即退出。但这种情况下,连接通过很长的循环来保持打开状态, 服务器脚本利用事件注册或者其他技术来侦测数据状态的改变。当状态改变时,数据进行传输,但传输完并不会直接关闭连接。

服务流依赖于 XMLHttpRequest 对象,在 AJAX 技术下被称做 Reverse AJAX 或者 Comet[5]。服务流是 XMLHttpRequest 在后台的长连接,它在连接长度和频率上带来了灵活性。页面会一次性正常加载,数据流的连接可以在预定义的生命周期内。服务器会像页面流中一样不停的循环,浏览器读取最新响应以更新内容。

1.3 Comet 和 BAYEUX 协议

Comet 允许服务端发送消息到客户端而无需客户端显示的请求。由于 AJAX 应用缺少通信标准, 在 Cometd group [8] 草拟了一个叫

收稿日期:2010-07-18
基金项目:上海市高新技术产业化重点项目计划-基于第三代移动通讯网的移动互联网应用和服务平台。
作者简介:李代立(1986-),男,山东济宁市人,硕士研究生,研究方向为嵌入式操作系统,系统软件支撑技术;陈榕(1957-),男,北京人,博士生导师,教授,中心主任,科泰世纪首席科学家,研究方向为嵌入式系统,构件技术。

BAYEUX 的 Comet 协议。BAYEUX 的消息格式是基于 JavaScript 语言子集的数据交互格式的,该协议已经被许多 Web 服务端实现。

但 Comet 的绊脚石在于各浏览器对 XHR、iFrames 两种实现 Comet 所需数据块的支持程度不尽相同,没有统一的实现标准。另外,无论是从网络还是开发角度来看,Comet 管理两个连接的开销都很大。这些开销带来的直接影响就是 Comet 应用中的传输延时,限制了它们所提供的实时通信的精确性[6]。

2 WebSocket 实时通信

前面几种方法提供了 Web 实时通信数据,但其中包含许多不必要的头数据,最重要的是全双工连接需要的不仅仅是从服务器到客户端的下行连接。简言之,HTTP 不是为实时的,全双工通信设计的。

2.1 WebSocket 介绍

HTML5 规范中定义,WebSocket 在浏览器和服务器之间采用单 socket 全双工(或者叫双向)传输来"推送"和"拉取"信息。这不但可以避免 Comet 中存在的连接和可移植问题,还能够提供比 Ajax 轮询更高效的解决方案。目前,HTML5 WebSocket 是推动 web 全双工实时通信的主要机制[7]。

在服务器与客户端之间,WebSocket 的连接是通过 WebSocket 协议在第一次"握手"时建立的,同时它也基于底层的 TCP/IP 协议。WebSocket 的协议比较简单,客户端和普通的浏览器一样通过 80 或者 443 端口和服务器进行请求握手,服务器根据 http header 识别是否一个 WebSocket 请求,如果是,则将请求升级为一个 WebSocket 连接,握手成功后就进入双向长连接的数据传输阶段。WebSocket 的数据传输是基于帧的方式:0x00 表示数据开始,0xff 表示数据结束,数据以 utf-8 编码。不像具有双向能力的 Comet 或 Ajax,HTML5 WebSocket 使浏览器具有本地能力,不需要安装插件,不需要维持一个双向连接就可以实现高效的 web 通信。一旦建立连接,Web Socket 数据帧能够以双通道的模式发送和接收数据。连接之后,客户端与服务端数据交互是通过暴露在浏览器里的 WebSocket 接口中 onmessage 和 postMessage 方法来实现的。

Html5 WebSocket 在应用层实现了与 TCP Socket 相同的功能,具有以下点:

1) 无缝穿过防火墙和路由器
2) 支持跨域交流
3) 完美整合基于 cookie 的认证
4) 完美整合 HTTP 负载平衡
5) 兼容二进制数据

2.2 WebSocket 使用

HTML5 WebSocket 的使用非常简单:只需要创建一个新的 WebSocket 实例,提供一个代表访问端的 URL 对象即可,前缀 ws:// 和 wss://分别代表 WebSocket 和安全的 WebSocket 连接。

```
var myWebSocket = new WebSocket("ws://www.websocket.org");
```

在连接到访问端发送数据之前,实现一些事件侦听器来处理连接过程中的事件触发:

```
myWebSocket.onopen = function(evt) { alert("Connection open ..."); };
myWebSocket.onmessage = function(evt) { alert( "Received Message: " + evt.data); };
myWebSocket.onclose = function(evt) { alert("Connection closed."); };
```

向服务器发送消息,只需要调用 postMessage 来传递你要发送的消息即可。在发送完之后,调用 disconnect 函数:

```
myWebSocket.postMessage("Hello Web Socket! Goodbye Comet! ");
myWebSocket.disconnect();
```

3 实验设计

3.1 实验目的

设计两种简单的 Web 聊天应用程序,一种基于 AJAX 长轮询,另一种基于 WebSocket,根据实验数据,对比二者效率,从而体现 WebSocket 在实时事件响应方案中的优越性。

3.2 实验工具

客户端使用支持 WebSocket 的浏览器 Google Chrome 5.0.375.55 及其自带开发工具,服务器端使用实现了 WebSocket 的开源 servlet 容器 Jetty(7.0.1.v20091125)。

3.3 实验过程

1) 基于 AJAX 长轮询的 Web 聊天应用程序。

服务端主要实现 HttpServlet 的 doPost, doGet 方法来处理客户端的请求。客户端则通过 XMLHttpRequest 来响应服务端产生的消息。当建立好连接之后,发现每次通过 XMLHttpRequest 请求服务端数据时,都会产生以下头信息:

```
Request Headers
Request URL:http://localhost:8080/chat/chat
Request Method:POST
Status Code:200 OK
Content-Type:application/x-www-form-urlencoded
Origin:http://localhost:8080
```

Referer:http://localhost:8080/chat/

User-Agent:Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/533.4 (KHTML, like Gecko) Chrome/5.0.375.55 Safari/533.4

Response Headers

Content-Encoding:gzip

Content-Type:text/json;charset=UTF-8

Date:Thu, 03 Jun 2010 12:18:41 GMT

Server:Jetty(7.0.1.v20091125)

Transfer-Encoding:chunked

2) 基于 WebSocket 的 Web 聊天应用程序。

服务端主要实现 WebSocket 接口和 WebSocketServlet 虚类来完成响应客户端事件。客户端主要通过 WebSocket 对象来侦听事件触发,并通过 onmessage 来接收消息。

为了建立一个 Web Socket 连接,客户端和服务器在初始握手期间要从 HTTP 协议升级到 WebSocket 协议,通过浏览器开发工具,得到以下浏览器请求和响应的头信息:

Request Headers

Request URL:http://localhost:8080/ws/

Request Method:GET

Status Code:200 OK

Accept: application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5

User-Agent:Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/533.4 (KHTML, like Gecko) Chrome/5.0.375.55 Safari/533.4

Response Headers

Accept-Ranges:bytes

Content-Encoding:gzip

Content-Length:1357

Content-Type:text/html

Date:Thu, 03 Jun 2010 12:27:30 GMT

Last-Modified:Wed, 25 Nov 2009 05:31:32 GMT

Server:Jetty(7.0.1.v20091125)

建立好连接后,WebSocket 数据帧就可以在客户端和服务器之间以全双工模式传输,在同一时间任何方向,可以全双工发送文本和二进制帧,最小的帧只有 2 个字节。在文本帧中,每一帧始于 0x00 字节,止于 0xFF 字节,数据使用 UTF-8 编码。WebSocket 文本帧使用终结器,而二进制帧使用一个长度前缀。

3.4 实验结果及对比

3.4.1 网络吞吐量对比

HTTP 请求和响应头信息总开销 570 字节,且不包括任何数据。在该例子中,聊天消息数据平均只有 20 个字符。使用 WebSocket,每个消息都是一个 WebSocket 帧,开销只有 2 个字节。

使用三个不同的用例 (A:1000 客户端, 每秒轮询一次;B:10000 客户端,每秒轮询一次;C:100000 客户端,每秒轮询一次)观察 AJAX 轮询应用程序关联的 HTTP 头数据与 WebSocket 消息帧需要的网络吞吐量,得结果如图 2。随着客户端数量的增加,WebSocket 应用程序产生的网络吞吐量相比 AJAX 轮询来讲是微乎其微的。

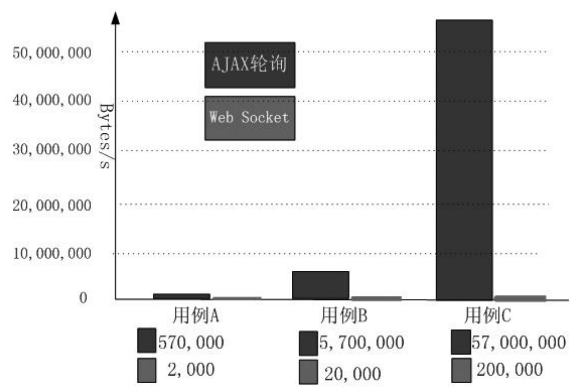

图 2 AJAX 轮询与 WebSocket 网络吞吐量对比

3.4.2 网络延迟对比

根据实验得出图 3 所示的网络延迟对比。在 AJAX 轮询方案下,消息从服务器传输到浏览器平均需要 50 毫秒, 因为当响应完成时,一个新请求又已发往服务器,这个新请求又需要 50 毫秒,在此期间服务器不能发送任何消息给浏览器,导致额外的服务器内存消耗。

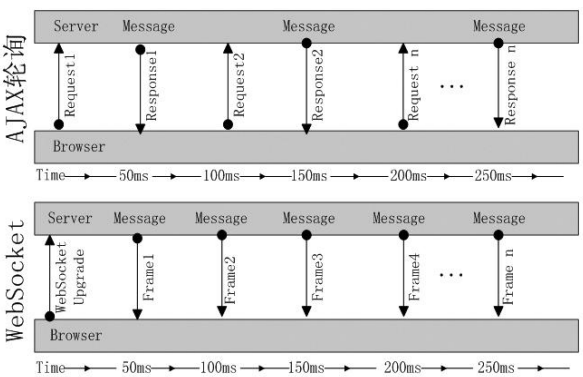

图 3 AJAX 轮询与 WebSocket 网络延迟对比

在 Web Socket 方案下,一旦连接升级到 WebSocket,消息的传输会更及时,从服务器传输到浏览器仍然需要 50 毫秒,但 WebSocket 连接保持打开,之后就再也不用向服务器发送请求了。因此,WebSocket 网络延时比 AJAX 轮询低很多。

4 结束语

尽管 Comet 和 AJAX 都可以提供桌面应用功能的终端用户体验, 而且传输延时也可以缩短到用户无法感知的程度, 但只有 WebSocket 才能真正为浏览器提供精确、高效的流事件,保证传输延时可以微乎其微,直至忽略不计。WebSocket 不仅通过单个 TCP/IP 连接提供完整的异步双工道流通信,更重要的是它能够支持浏览器和源服务的消息采用同样的格式,对于 Web 实时通信中的研究意义重大。

(下转第 7935 页)

9 基于 PPP 协议的远程拨号连接

个人远程通信一般需通过系统的“拨号连接”、MODEM(调制解调器,俗称“猫”)及电信部门的普通电话线路,连接到远方个人计算机、远方单位局域网或 Internet 网站(见图 9)。

图 9

程序界面(见图 10)及代码设计如下。

```
Private Sub Command1_Click()
   Shell "rasphone.exe " & Text1.Text, vbNormalFocus '用 Shell 加载 rasphone 命令
End Sub
```

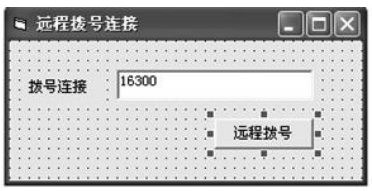

图 10

它的运行条件是,

a. 你的电脑中已经正确地安装有调制解调器(MODEM)及其设备驱动程序;

b. 正确地将 MODEM 通过 R14 双绞线连接到你的家庭电话线插座;

c. 在“网络连接”窗口中创建了一个“拨号网络”。

10 基于 Telnet 协议的远程主机登录

Telnet 是 DOS 时代的主要上网方式之一。它允许用户计算机以终端的方式连接并访问远方的 UNIX 网络主机 (具有独立注册 IP 地址和 Internet 域名的网络服务器),并在规定的权限内对主机进行浏览、下载等操作。现在仍然是网管们上网的一种辅助手段(见图 11)。

图 11

程序界面(见图 12)及代码设计如下。

```
Private Sub Command1_Click()
    Shell "telnet " & Text1.Text, vbNormalFocus
                                    '用 Shell 加载 telnet 命令
End Sub
```

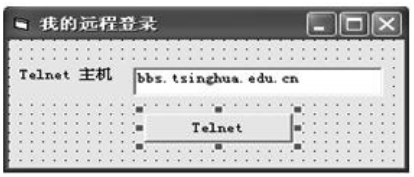

图 12

它的运行条件是你的电脑能够正常地上网。

11 小结

以上十个最常用网络应用的 DIY 最快解决方案, 主要采用了 WebBrowser、MAPI、Inet、和 Winsock 等网络控件及 rasphone.exe 和 telnet.exe 等网络命令。

通过选择最简洁的编程语言及添加专门的网络控件和调用专门的网络命令,来实现复杂的网络编程,而不陷入网络底层许多复杂技术细节的“泥潭”。这就是我们的编程秘诀。

参考文献:

[1] 郝春强.Virsual Basic 案例教程[M].北京:中科多媒体电子出版,2001.
[2] 萧秋水.Windows 网络编程之 VB 篇[M].北京:清华大学出版社,2001.
[3] 黄嘉辉. 互联网与 TCP/IP 进阶程序设计[M].北京:中国青年出版社,2002.
[4] 张树兵. Virsual Basic 6.0 中文版入门与提高[M].北京:清华大学出版社,1999.
[5] 李迎春,朱诗兵. Virsual Basic 6.0 网络编程[M].北京:北京希望电子出版社,2001.

(上接第 7925 页)

参考文献:

[1] HTTP-Hypertext Transfer Protocol[EB/OL].http://www.w3.org/Protocols/,2010.
[2] 周婷.基于 HTTP 长连接的“服务器推”技术.[EB/OL] http://www.ibm.com/developerworks/cn/web/wa-lo-comet/,2007.
[3] M. Bhide, P.Deolasee, A. Katkar, A. Panchbudhe, K. Ra-mamritham, and P. Shenoy. Adaptive push-pull: Disseminat-ing dynamic web data. IEEE Trans. Comput.,2002,51(6):652-668.
[4] Netscape. An exploration of dynamic documents[EB/OL]. http://wp.netscape.com/assist/netsites/pushpull.html, 1996.
[5] A. Russell. Comet: Low latency data for the browser[EB/OL].http://alex.dojotoolkit.org/?p=545.
[6] Dionysios G. Synodinos. HTML 5 Web Sockets vs. Comet and Ajax,2008.
[7] W3C.The WebSocket API [EB/OL].http://dev.w3.org/html5/websockets/,2010.
[8] CometD Bayeux Ajax Push [EB/OL].http://www.cometd.com /,2010.