**要**

昨天吃饭的时候，突然想起来一个好玩的事，如果能有个页面可以实时的监控网站或者其他类型的程序的日志，其实也不错。当然，网上也有很多成熟的类似的监控系统。就想着如果通过.net该如何实现？所以就在想，通过系统内部将消息推送到前端，.net中可以通过pull或者push的方式，pull通常的做法就是ajax方式不停的请求一个接口，这种方式，不太理想。然后就在想如何通过服务端想客户端推送消息。之前看到过SignalR方面的文章可以实现push的功能，signalr也是第一次接触，在这个网站<http://www.asp.net/signalr>看了一些文章。就自己动手做了这样的日志监控的demo。

**一个简单的例子**

先上一段代码，如下：

该类是一个监控日志文件是否变化的一个单例类。其中维护一个日志队列。

[复制代码](javascript:void(0);)

public class FileSystemWatcherSingle

{

public FileSystemWatcher wather;

private static FileSystemWatcherSingle \_instance;

private static readonly object \_objLock = new object();

public Queue<Log> MyQueue;

private string \_watherFolderPath = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "Log");

private FileSystemWatcherSingle()

{

if (MyQueue == null)

{

MyQueue = new Queue<Log>();

}

if (wather == null)

{

wather = new FileSystemWatcher();

if (!Directory.Exists(\_watherFolderPath))

{

Directory.CreateDirectory(\_watherFolderPath);

}

wather.Path = \_watherFolderPath;

wather.EnableRaisingEvents = true;

}

}

public static FileSystemWatcherSingle CreateInstance()

{

if (\_instance == null)

{

lock (\_objLock)

{

if (\_instance == null)

{

\_instance = new FileSystemWatcherSingle();

}

}

}

return \_instance;

}

}

[复制代码](javascript:void(0);)

写日志的操作

在写入文件之前，将当前的日志加入到日志队列里面。

[复制代码](javascript:void(0);)

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Web;

namespace Wolfy.LogMonitor.Models

{

public class LogHelper

{

public static void WriteLog(Log log)

{

string dir = Path.Combine(AppDomain.CurrentDomain.BaseDirectory, "Log");

if (!Directory.Exists(dir))

{

Directory.CreateDirectory(dir);

}

string \_watherFilePath = Path.Combine(dir, DateTime.Now.ToString("yyyy-MM-dd") + ".log");

if (!File.Exists(\_watherFilePath))

{

File.Create(\_watherFilePath);

}

FileSystemWatcherSingle wather = FileSystemWatcherSingle.CreateInstance();

wather.MyQueue.Enqueue(log);

File.AppendAllText(\_watherFilePath, string.Format("{0} {1}\r\n{2}", log.Type.ToString(), log.Dt, log.Content));

}

}

}

[复制代码](javascript:void(0);)

LogHub类

需要安装SignalR，在安装完成时，有个Readme文件，根据里面的提示，需要添加StartUp类，添加对应的owin程序集。

[复制代码](javascript:void(0);)

using Microsoft.Owin;

using Owin;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using Wolfy.LogMonitor.App\_Start;

[assembly: OwinStartup(typeof(Startup))]

namespace Wolfy.LogMonitor.App\_Start

{

public class Startup

{

public void Configuration(IAppBuilder app)

{

app.MapSignalR();

}

}

}

[复制代码](javascript:void(0);)

LogHub继承Hub类，并在这里想所有客户端推送消息。并在这里面面为文件监控类FileSystemWatcher注册创建和Change事件。

[复制代码](javascript:void(0);)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using Microsoft.AspNet.SignalR;

using Microsoft.AspNet.SignalR.Hubs;

using System.Timers;

using Newtonsoft.Json;

namespace Wolfy.LogMonitor.Models

{

[HubName("logHub")]

public class LogHub : Hub

{

private FileSystemWatcherSingle fileWather;

public void Send(string name, string message)

{

// Call the addNewMessageToPage method to update clients.

Clients.All.addNewMessageToPage(name, message);

}

public LogHub()

{

fileWather = FileSystemWatcherSingle.CreateInstance();

fileWather.wather.Changed += wather\_Changed;

fileWather.wather.Created += wather\_Created;

}

void wather\_Created(object sender, System.IO.FileSystemEventArgs e)

{

this.Send("system", "创建文件:" + e.Name);

}

void wather\_Changed(object sender, System.IO.FileSystemEventArgs e)

{

while (fileWather.MyQueue != null && fileWather.MyQueue.Count > 0)

{

Log log = fileWather.MyQueue.Dequeue();

if (log != null)

{

this.Send(log.Type.ToString(), JsonConvert.SerializeObject(log));

}

}

}

}

}

[复制代码](javascript:void(0);)

前端

home：log展示的页面。根据日志类型为该条日志显示不同的颜色。

[复制代码](javascript:void(0);)

@{

ViewBag.Title = "Home";

}

<div id="container">

</div>

<script>

$(function () {

// Reference the auto-generated proxy for the hub.

var log = $.connection.logHub;

console.log(log);

// Create a function that the hub can call back to display messages.

log.client.addNewMessageToPage = function (name, message) {

console.log(name);

console.log(message);

// Add the message to the page.

if (name === "Info") {

$('#container').append('<p style="color:green;">' + message + '</p>');

} else {

$('#container').append('<p style="color:red;">' + message + '</p>');

}

};

// Start the connection.

$.connection.hub.start().done();

});

</script>

[复制代码](javascript:void(0);)

LogController

Test页面是一个测试页面，通过不停的刷新写入随机类型的日志。

[复制代码](javascript:void(0);)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Wolfy.LogMonitor.Models;

namespace Wolfy.LogMonitor.Controllers

{

public class LogController : Controller

{

// GET: Log

public ActionResult Home()

{

return View();

}

public ActionResult Test()

{

LogType[] logtypes = { LogType.Info, LogType.Error };

Random r = new Random();

int index = r.Next(0, 2);

LogHelper.WriteLog(new Log { Content = "这是一次日志", Dt = DateTime.Now, Type = logtypes[index] });

return View();

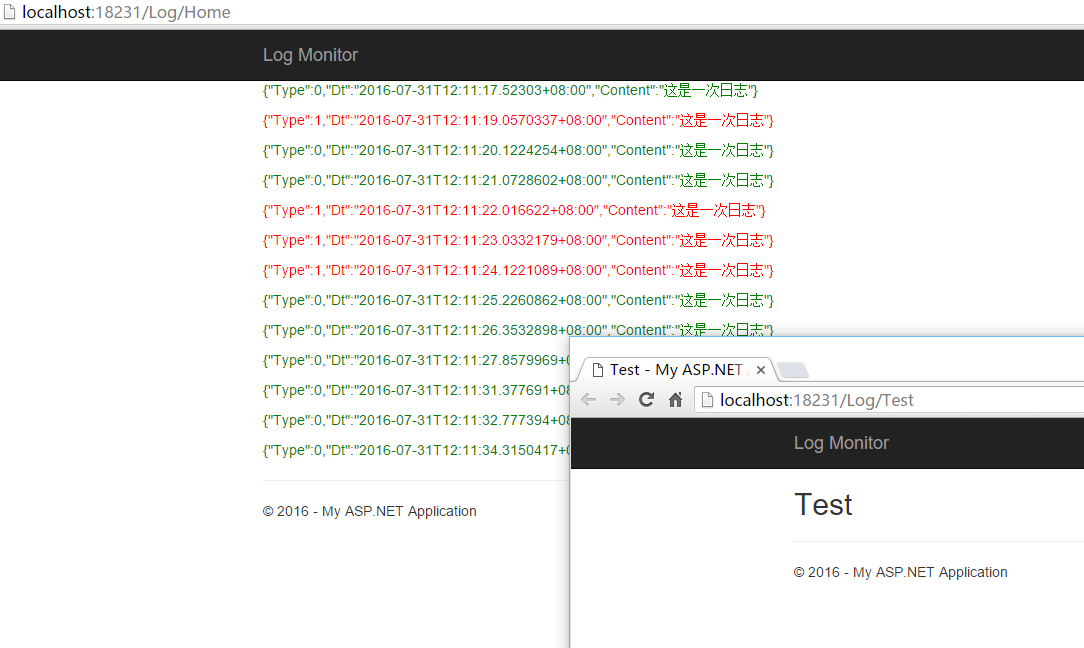
}

}

}

[复制代码](javascript:void(0);)

好了，到现在基本上大功告成。测试一下。



通过不停的刷新test页面，你会发现home页面上会相应的动态展示这次操作的日志。

**总结**

昨天突然有这个想法，今天就折腾了一上午，将这个想法用代码实现了一下。这里没有signalr的相关介绍，感兴趣的话，可以看下面的参考资料中的内容。

参考资料

http://www.asp.net/signalr

* 博客地址：<http://www.cnblogs.com/wolf-sun/>  
  博客版权:如果文中有不妥或者错误的地方还望高手的你指出，以免误人子弟。如果觉得本文对你有所帮助不如【推荐】一下！如果你有更好的建议，不如留言一起讨论，共同进步！ 再次感谢您耐心的读完本篇文章。

分类: [[ASP.NET MVC]](https://www.cnblogs.com/wolf-sun/category/600410.html)