# **Photon Server游戏服务器教程（二） Log日志文件配置与输出**

上一篇文章和大家介绍了[Photon Server游戏服务器的配置](http://gad.qq.com/article/detail/43978)，这篇我们就来学习下Log日志文件配置与输出。

**一、Log日志的打开**

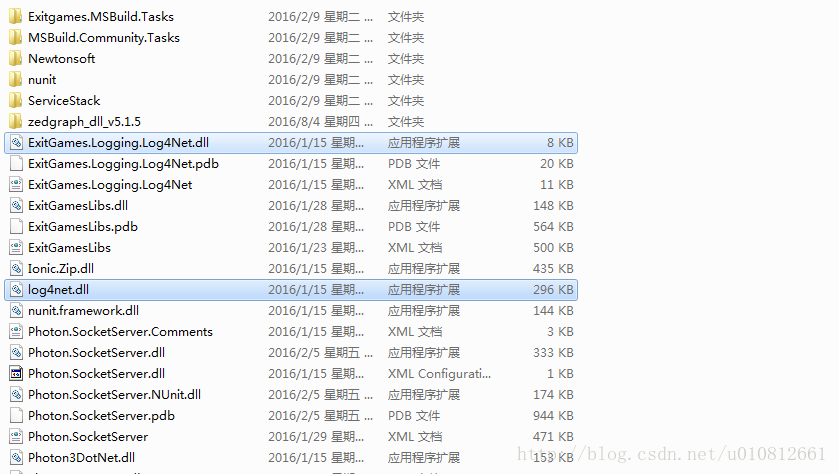
在Photon中点击Open Logs选项，打开BareTail软件，相当于一个记事本，用于Log日志的输出。

打开以后的界面

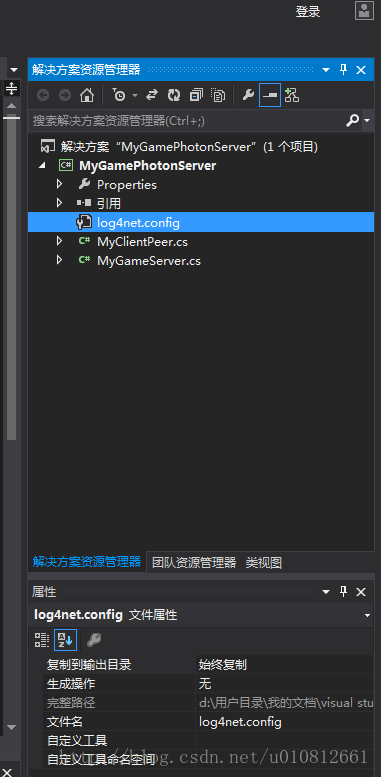


**二、Log日志的输出**

1.引用log4net.dll 和ExitGames.Logging.Log4Net.dll，位于lib文件夹下。



2.添加配置文件，由于有固定的格式，因此我们在src-server\Mmo\Photon.MmoDemo.Server文件夹下复制一份log4net.config文件放在我们项目的根目录下，修改属性为始终复制。



配置文件路径 Photon:ApplicationLogPath，修改为自己的程序名MyGame。

1

&lt;file type="log4net.Util.PatternString" value="%property｛Photon:ApplicationLogPath｝\\MyGame.Server.log" /&gt;

3.在主类MyGameServer中添加一个静态只读字段log并初始化

1

using System;

2

using System.Collections.Generic;

3

using System.Linq;

4

using System.Text;

5

using System.Threading.Tasks;

6

using Photon.SocketServer;

7

using ExitGames.Logging;

8

using ExitGames.Logging.Log4Net;

9

using System.IO;

10

using log4net.Config;

11

namespace MyGamePhotonServer

12

｛

13

//所有的server端 主类都要集成自applicationbase

14

public class MyGameServer:ApplicationBase

15

｛

16

public static readonly ILogger log = LogManager.GetCurrentClassLogger();

17

/// &lt;summary&gt;

18

/// 刚一个客户端请求连接的

19

/// &lt;/summary&gt;

20

/// &lt;param name="initRequest"&gt;&lt;/param&gt;

21

/// &lt;returns&gt;&lt;/returns&gt;

22

protected override PeerBase CreatePeer(InitRequest initRequest)

23

｛

24

log.Info("一个客户端连接过来了。。。。");

25

return new MyClientPeer(initRequest);

26

｝

27

/// &lt;summary&gt;

28

/// 初始化

29

/// &lt;/summary&gt;

30

protected override void Setup()

31

｛

32

// 日志的初始化

33

log4net.GlobalContext.Properties["Photon:ApplicationLogPath"] = Path.Combine(

34

Path.Combine(this.ApplicationRootPath, "bin\_Win64"), "log");

35

FileInfo configFileInfo = new FileInfo( Path.Combine( this.BinaryPath ,"log4net.config"));

36

if (configFileInfo.Exists)

37

｛

38

LogManager.SetLoggerFactory(Log4NetLoggerFactory.Instance);//让photon知道使用的是Log4NetLog插件

39

XmlConfigurator.ConfigureAndWatch(configFileInfo);//让log4net这个插件读取配置文件

40

｝

41

log.Info("初始化完成！");

42

｝

43

/// &lt;summary&gt;

44

/// server端关闭的时候

45

/// &lt;/summary&gt;

46

protected override void TearDown()

47

｛

48

log.Info("服务器应用关闭了");

49

｝

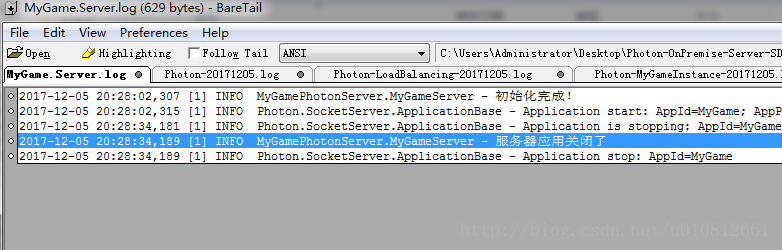
50

｝

51

｝

运行后的日志输出结果



来自：https://blog.csdn.net/u010812661/article/details/78717214