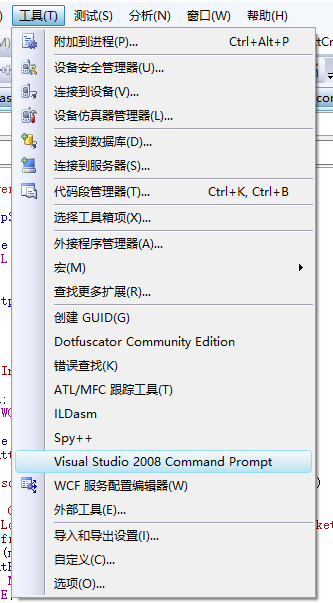
**类名：CXHttpServer**

**功能：基于libevent库实现一个可以接收并返回url报文的http轻量服务端**

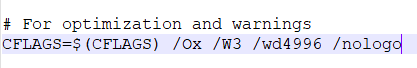
**编译使用：**

1. 从<https://libevent.org/>获取库工程，其中2.0.22版本适用于vs2008工具
2. Windows系统下解压.tar.gz并得到文件夹，使用vs2008命令行工具



1. 使用cd /d xxxxxxxx\libevent-2.0.22-stable 进入文件夹并找到makefile.nmake文件。

修改编译设置：



可以增加/MD、/MTD、/MDD、/MT

默认是relase版本，如果需要编译debug版本则还需要加入 /D\_DEBUG

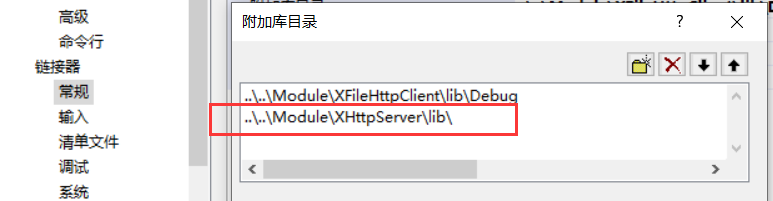
4.使用nmake /f makefile.nmake static\_libs指令编译库，得到3个lib文件

**工程基本设置：**

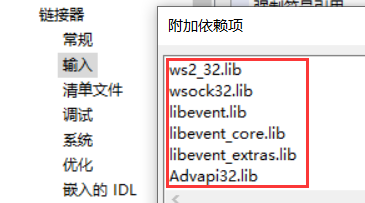
1. 设置包含路径（libevent内部include采用#include<>，所以需要设置额外的包含路径）



1. 设置lib路径



1. 增加libevent库所需要的lib



共六个：

ws2\_32.lib、wsock32.lib、libevent.lib、libevent\_core.lib、libevent\_extras.lib、Advapi32.lib

**主要函数：**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

初始化函数，设置Windows异步[套接字](https://baike.baidu.com/item/%E5%A5%97%E6%8E%A5%E5%AD%97/9637606)启动命令

使用libevent的初始化函数创建http服务

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BOOL InitHttpServer(std::string strIP, int nPort)

{

WSADATA wsaData;

WSAStartup(MAKEWORD(2, 2), &wsaData);

m\_pHttpEventBase = event\_base\_new();

m\_pEvHttp = evhttp\_new(m\_pHttpEventBase);

if(evhttp\_bind\_socket(m\_pEvHttp, strIP.c\_str(), nPort) == -1)

{

long nerr = GetLastError();

CLogPrint::LogString(XLOGLEVEL\_INFOR,\_T("evhttp\_bind\_socket: failed"));

event\_base\_free(m\_pHttpEventBase);

evhttp\_free(m\_pEvHttp);

m\_pHttpEventBase = NULL;

m\_pEvHttp = NULL;

return FALSE;

}

return TRUE;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

设置子路径对应的回调函数

strPath： url的子路径

cb：回调函数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BOOL CXHttpServer::SetHttpServerCallBack(const char\* strPath, void (\*cb)(struct evhttp\_request \*, void \*))

{

if(m\_pEvHttp == NULL)

{

CLogPrint::LogString(XLOGLEVEL\_INFOR,\_T("evHttp: failed"));

return FALSE;

}

long nRes = evhttp\_set\_cb(m\_pEvHttp, strPath, cb, NULL);

switch(nRes)

{

case 0:

break;

case -1:

CLogPrint::LogString(XLOGLEVEL\_INFOR,\_T("Callback existed already"));

return FALSE;

case -2:

CLogPrint::LogString(XLOGLEVEL\_INFOR,\_T("callback: failed"));

return FALSE;

default:

break;

}

return TRUE;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

启动服务器,成功启动则分发消息

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BOOL CXHttpServer::RunHttpServer ()

{

if(m\_pHttpEventBase == NULL)

{

CLogPrint::LogString(XLOGLEVEL\_INFOR,\_T("httpEventBase: failed"));

return FALSE;

}

event\_base\_dispatch(m\_pHttpEventBase);

return TRUE;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

关闭服务器,清空设置

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void CXHttpServer::ExitHttpServer ()

{

if(m\_pHttpEventBase)

{

event\_base\_free(m\_pHttpEventBase);

}

if(m\_pEvHttp)

{

evhttp\_free(m\_pEvHttp);

}

WSACleanup();

}