**mdk开源网络引擎**

# 开发背景

我有一个坏习惯——作为一个Coder就是不喜欢读代码

在我的意识里，一个好的代码库应该是足够稳定的，稳定到不需要用户为了解决问题而去读日志钻代码。

所以我总希望能有一个轻量，轻量，再轻量的开发库就好了，于是萌生了开发一个微量级网络引擎的想法，于是mdk(Micro-Development-Kit)微量级开发工具包，就是这套框架的名字

# mdk做了什么事情

编写网络程序，程序员开发最关心的其实只有3件事情，新连接到达，数据到达，连接断开。

mdk就是基于以上3个主要业务，为框架设计了OnConnect() OnMsg() OnClose()3个主要回调方法，用户的主要的工作就是实现这3个方法，所以我将这种开发模式命名为面向业务的开发。

用户不再关心底层各种繁杂的处理，比如线程模型，各种同步，内部对象生命周期，各种缓冲只要专心写业务就好

整个框架代码大约5000行，并且以后增加新代码时也会非常谨慎，力求保证“微量”的特质。

希望这款网络能给大家的工作带来更多便利。

水平有限，如果发现什么bug或问题，欢迎指出，我一定尽快答复

# 项目相关url

* 下载地址
* github下载：<https://github.com/huoyu820125/Micro-Development-Kit>
* google下载：<http://code.google.com/p/micro-development-kit/>
* 文档地址（wiki）：<https://github.com/huoyu820125/Micro-Development-Kit/wiki>
* OSChina博客：<http://my.oschina.net/u/732357/blog>
* 脑图（.mm文件）：使用freemind\_1.0.0软件打开

# 开发环境

* windows下，使用速度最快的经典VC++6.0，追求速度的朋友有服了
* linux下直接makefile编译

# 更新历史

**V1.50(2012.12.26)**  
1.解决gcc4.4.6编译问题，原因1.32版中新增的ShareMemory类用到memcpy系列字符串api，在gcc4.4.6下找不到头文件  
2.整理example代码，删除无用代码  
  
**V1.35(2012.12.24)**  
1.解决Bug 连接断开后，自动重连没有效果  
2.NetServer STNetServer类补上重连定时设置接口  
3.ConfigFile类写入顺序与读入顺序保持一至  
  
**V1.34 (2012.12.14)**

主要是为了方便windows下VS系列使用者编译，做了相关目录的修正

      包含头文件修正为相对路径，  
编译mdk lib库不再需要在VC下再设置头文件搜索目录但其它项目使用mdk.lib依旧需要设置mdk头文件目录，  
或包含时指定目录  
  
  
  
**V1.33(2012.12.12)**  
加强ConfigFile功能  
直接存取int char short int32 float double等基本类型  
  
  
**V1.32 (2012.12.10)**  
1.将代码中实现的通用api整理到一起mapi.\*  
2.补正VC工程，新增单线程版代码没有编译进lib  
3.相应更新文档  
4.增加工具类共享内存ShareMemory.\*  
5.配置文件类ConfigFile.\*更加傻瓜式，析构时自动保存修改  
  
  
**V1.30 (2012.12.04)**  
1.主要是为实现nosql需要单线程的io模型，于是新增一个单线程模型服务器引擎匡架的  
   基类为STNetServer与NetServer接口完全一样   
   事件响应返回host类型为STNetHost，与NetHost接口完全一样  
2. 1.25之后小更新  
     1.修正了Socket::Accept在windows下的失败bug  
     2.发送部分io吞吐能力忘记跟随recv一起扩大，不扩大也没关系，就是原来的低吞吐，已扩大  
  
  
**V1.25 (2012.11.30)**  
主要优化了io吞吐能力  
1.每次io吞吐量从256byte提升至8k，就是IOBufferBlock.h中的宏#define BUFBLOCK\_SIZE 8192，BUFBLOCK\_SIZE是一块连续内存的最大尺寸，所以一次recv最大可接收的数据受此宏定义限制。  
  
可根据需求自行修改，注意BUFBLOCK\_SIZE很大时，请适当调小IOBufferBlock::ms\_pMemoryPool初始化内存数量,目前是10000块，也就是8万k大小的内存池  
2.linux下每次io从4k提升至1M  
3.NetEngine::HeartMonitor()小优化

1. **V1.24 (2012.11.29)**      
   首先最近几次更新，要**感谢**来至OSChina.net的**Fenlog**积极提出的相关意见
2. 1.删除被注释掉的无用代码（所谓的僵尸代码）  
   2.消除class NetServer内存泄露，~NetServer()时没有释放NetEngine对象  
       还有一个地方的内存没有释放，就是以下红色的全局变量，而且是private的，程序退出就会释放没有关系的，如果实在不爽，可以自行修改为public或增加static释放方法，在程序退出的地方，主动释放一下  
   class IOBufferBlock  
   {  
           friend class IOBuffer;  
   //////////////////////////////////////////////////////////////////////////  
   //使用自己的内存管理  
   private:  
           //内存池，在这里为该类型的对象分配内存  
           static MemoryPool\* ms\_pMemoryPool;  
           //内存池线程安全锁  
           static Mutex\* ms\_pPoolMutex;  
     
     
     
   **V1.23 (2012.11.21更新)**  
   修正了接收数据的一切边界情况  
   1.nethost::recv第2个参数传递0  
       具体现象：由于没有读取数据，缓冲状态一直为可读，导致不停的触发OnMsg，修正后如果recv传递0则返回false，且不再触发onmsg，除非有 新数据到达修正：IOBuffer::ReadData（），第一行参数检查，0 > uLength 修正为0 >= uLength，读取0长度也返回不可读  
   2.nethost::recv刚好将缓冲中数据全部读出  
       具体现象：iobuffer中已经没有数据可读，但onmsg会再被触发1次，数据长度为0修正：NetConnect::IsReadAble()方法，状态为可读时，多检查一下iobuffer的长度，如果<=0则返回不可读  
     
   **V1.22r (2012.11.21更新)**  
     
   修正Bug:VC编译Release版本启动崩溃  
          Bug原因是Executor类中调用函数指针的汇编代码在release下不能正确定位函数地址，导致访问非法地址  
         修正后所有需要经过Executor远程调用的类成员函数（例如传递给Thread.Run()，ThreadPool::Accept()，Task::Accept()的成员函数），需将函数的调用规则申明为RemoteCall  
         如:void\* RemoteCall Fun(void\*)  
     
   **V1.21  （2012.11.08更新）**  
     
   框架优化1处，接口增加2个  
   1.对于框架的优化，保证OnConnect OnMsg OnClose完全序列化，这要感谢野火与Connor提出的宝贵建议  
     
   2.NetHost增加GetAddress和GetServerAddress方法  
   用于得连接双方IP 端口信息  
     
     
     
   **V1.20  便利性的优化**  
     
   1.修正1.10版本中编译Signal.cpp时，include atom.h文件名大小写不可识别问题  
     
   2.优化服务器引擎NetHost的使用删除了Hold()Free()方法，取而代之的是复制构造函数，=运算符，析构函数来自动化管理引用计数，  
   用起来更加方便，让用户完全不用关心对象使用中的生命周期问题  
     
   3.优化服务器网络事件通知顺序1.10版本中同一个连接上OnConnect与OnMsg是无序并发的优化后在同一个连接上，确保了OnConnect完成之前，不会有该连接的OnMsg发生  
     
   4.相应的修正example与文档  
     
   5.mdk类库，增加智能指针类SharedPtr  
     
   6.mdk类库，Thread在windows下stop时没有CloseHandle()。（前几天群里看到讨论CloseHandle()问题，帮我发现到一个不足点，于是补上呵呵）  
     
   **V1.10  github上发布以来第一个大版本**  
     
   1.各方面已经调整稳定，为将来写wiki有针对性，设置一个大版本  
     
   2.增加了服务器引擎接口文档.docx，方便喜欢看文档的朋友  
     
   3.修正NetServer.h NetHost.h方法注释，与增加了服务器引擎接口文档.docx完全一样，方便喜欢直接看代码的朋友  
     
   4.修正NetHost关于Group的方法为线程安全的  
     
   5.修正心跳检查机制默认为不检查，与NetServer::SetHeartTime()的注释匹配，以免用户被注释误导