

# 破冰号交易终端

## API 手册

(更新日期: 2012-12-22)

### 内容目录

1. 欢迎使用破冰号交易终端(API).....	2
2. 安装和卸载.....	2
3. 下单命令详细解释.....	2
4. 程序开发指南.....	3
4.1. Visual C++ (以 VC2010 学习版为例) .....	3
4.2. Matlab (以 Matlab R2012a 为例) .....	7

## 1. 欢迎使用破冰号交易终端(API)

破冰号交易终端(API)是底层用 C++实现的自动化下单的 API，使用该 API，用户可以直接用 C/C++/C#/Java/Matlab/R/等语言直接下单。

我们致力于为期货行业的投资者提供一个快捷的下单工具，但并不保证该软件能为所有的使用者带来盈利。

感谢您选择破冰号交易终端 API，希望您能够通过使用该系统找到乐趣，并能创造更多价值。

## 2. 安装和卸载

### 下载

用户可以登录破冰号交易终端下载页面，找到要下载的相关文件 ibt\_client.rar，直接点击下载。

下载地址：<http://code.google.com/p/icebreaker-trader/downloads/list>

### 安装

直接解压 ibt\_client.rar 即可（比如：d:\ibt\_client）。

### 卸载

直接删除 ibt\_client 目录即可。

ibt\_client 文件夹中文件说明：

- 1) ibt\_client.h <== 开发的头文件
- 2) ibt\_log4cplus.properties <== log4cplus 日志配置文件
- 3) libibt\_client\_ex.dll <== dll 文件
- 4) libibt\_client\_ex.lib <== lib 文件
- 5) libibt\_client\_test.exe <== API 测试程序
- 6) msvcrt100.dll <== 依赖的 VC 库文件
- 7) msvcp100.dll <== 依赖的 VC 库文件

## 3. 下单命令详细解释

```
int ibt_send_order(int Policy_ID, int BuyOrSell, int EntryOrExit, int Lot);
```

参数说明：

1. Policy\_ID - 策略 ID，即破冰号交易终端的配置的策略 ID
2. BuyOrSell - 买入或卖出，IBT\_Enum\_Buy(0)-买入，IBT\_Enum\_Sell(1)-卖出
3. EntryOrExit - 开仓或平仓，IBT\_Enum\_Entry(0)-开仓，IBT\_Enum\_Exit(1)-平仓，IBT\_Enum\_ExitToday(2)-平今
4. Lot - 交易手数

### 返回值:

- 0 - 成功
- 1 - 失败

### 特别说明:

1. 对于平仓, Lots = 0, 则平掉该策略的所有仓位。
2. 对于平仓, 无论你选择平昨还是平今, 都会平掉相应的 Lot 手数, 选择平今只是优先平今。比如, 你有 5 手昨仓, 3 手今仓, 如果 Lot=8, 则会平掉所有的仓位。如果你选择了平今, Lot=5, 则会平今仓=3, 平昨仓=2。
3. 对于下单指令, 没有商品名和价格, 这两项都在破冰号交易终端中配置。

### 举例:

1. 开多仓

```
int i_ret = ibt_send_order( Policy_ID, IBT_Enum_Buy, IBT_Enum_Entry, Lot);
```

2. 开空仓

```
int i_ret = ibt_send_order( Policy_ID, IBT_Enum_Sell, IBT_Enum_Entry, Lot);
```

3. 平多仓

```
int i_ret = ibt_send_order( Policy_ID, IBT_Enum_Sell, IBT_Enum_ExitToday, Lot);
```

4. 平空仓

```
int i_ret = ibt_send_order( Policy_ID, IBT_Enum_Buy, IBT_Enum_ExitToday, Lot);
```

## 4. 程序开发指南

### 4.1. Visual C++ (以VC2010 学习版为例)

1. 新建一个“Win32 控制台应用程序”项目 test\_ibt\_client

新建项目

最近的模板

已安装的模板

Visual C++

CLR

Win32

常规

排序依据: 默认值

Win32 控制台应用程序

Win32 项目

Visual C++

Visual C++

搜索 已安装的模板

类型: Visual C++

用于创建 Win32 控制台应用程序的项目

名称(N): test\_ibt\_client

位置(L): C:\e\svn\

解决方案名称(M): test\_ibt\_client

浏览(B)...

☒ 为解决方案创建目录(D)

确定

取消

Win32 应用程序向导 - test\_ibt\_client



应用程序设置

概述

应用程序设置

应用程序类型:

☐ Windows 应用程序(W)

☒ 控制台应用程序(C)

☐ DLL(D)

☐ 静态库(S)

附加选项:

☐ 空项目(E)

☐ 导出符号(X)

☒ 预编译头(P)

添加公共头文件以用于:

☐ ATL(A)

☐ MFC(M)

< 上一步

下一步 >

完成

取消

2. 把 d:\ibt\_client 目录加入到包含目录(include 路径)和库目录(lib 路径)中。



3. 复制下列代码到 `test_ibt_client.cpp` 文件

```
#include "stdafx.h"

#include <iostream>

#include "ibt_client.h"

#pragma comment(lib, "libibt_client_ex.lib")

using namespace std;

static int test_send_order();

int _tmain(int argc, _TCHAR* argv[])
{
    test_send_order();

    return 0;
}

int test_send_order()
{
    ibt_debug("test_send_order start...");

    int i_ret = 0;
    //发送开仓命令
```

```

int Policy_ID = 26;
// 买入开仓
/*int BuyOrSell = IBT_Enum_Buy;
int EntryOrExit = IBT_Enum_Entry;
int Lot = 1;*/

// 卖出平仓
int BuyOrSell = IBT_Enum_Sell;
int EntryOrExit = IBT_Enum_ExitToday;
int Lot = 0;

i_ret = ibt_send_order(Policy_ID,BuyOrSell,EntryOrExit,Lot);
if (0 != i_ret)
{
    ibt_error("破冰号客户端发送命令失败!");
}
else
{
    ibt_info("破冰号客户端发送命令成功!");
}

ibt_debug("test_send_order end.");

return i_ret;
}

```

#### 4. 编译 test\_ibt\_client 项目

#### 5. 运行

1) 把 **ibt\_log4cplus.properties**, **libibt\_client\_ex.dll** 两个文件复制到运行的目录下。

2) 启动“破冰号交易终端”并登录成功

3) 运行 test\_ibt\_client.exe

2012-12-22 18:09:49.405 INFO [2624] [Trader] IBT\_Client 发送命令. 策略 ID=26, BuyOrSell=1, EntryOrExit=2, Lot=0, SentTime=201

21222.180949, i\_ret=0

2012-12-22 18:09:49.436 INFO [2624] [ibt\_client] 破冰号客户端发送命令成功!

4) 在“破冰号交易终端”中交易日志中可以看到日志:

18:09:49.405 Task\_SendOrder --->>> 收到 API 报单请求。 策略 ID=26, 类型=卖平仓, 手数=0, 下单时间=20121222.180949

18:09:49.405 Task\_SendOrder --->>> 没有找到对应的策略 ID(26), 因此该交易指令无效!

## 4.2. Matlab (以 Matlab R2012a 为例)

1. 解压 ibt\_client.rar 文件到一个目录 (假如把该文件解压到 d:\ibt\_client)

2. 把 ibt\_log4cplus.properties 复制到 MATLAB 的 bin 目录下。

假如 MATLAB 安装在 d:\MATLAB\R2012a, 则复制到: d:\MATLAB\R2012a\bin 目录下

3. 设置编程环境

```
rehash toolboxcache
```

```
mex -setup
```

注: 此步骤只需要运行一次。

4. 加载动态库 (假如把该文件解压到一个目录, 比如 d:\ibt\_client)

```
loadlibrary('d:\ibt_client\libibt_client_ex.dll','d:\ibt_client\ibt_client.h');
```

5. 调用下单函数

```
[a1] = calllib('libibt_client_ex','ibt_send_order',2,0,0,1);
```

6. 卸载动态库

```
unloadlibrary('libibt_client_ex');
```

7. 查看动态库中所有函数

```
libfunctions libibt_client_ex -full
```

8. 调用其他函数

1) 显示启动路径

```
[a1] = calllib('libibt_client_ex','ibt_current_dir');
```

2) 打印调试信息

加载 log 配置文件, 这一步可以不用做。

```
calllib('libibt_client_ex','ibt_init_ex',  
        'd:\ibt_client\ibt_log4cplus.properties');
```

打印 DEBUG 信息

```
calllib('libibt_client_ex','ibt_debug','hell world');
```

打印 INFO 信息

```
calllib('libibt_client_ex','ibt_info','hell world');
```

打印 ERROR 信息

```
calllib('libibt_client_ex','ibt_error','hell world');
```

3) 查看日志

所有的日志都保存在在 MATLAB 的 bin 目录下的 ibt\_normal.log 文件里面

在 MATLAB 的 bin 目录下的 ibt\_trader.log 是交易日志文件, 如果你发了交易指令, 则可以在 ibt\_trader.log 中查看到。