

```

1  using System;
2  using System.Collections.Generic;
3
4  namespace TaiTanXing.Ctp.CtpNet
5  {
6      /// <summary>
7      /// CtpNet接口
8      /// </summary>
9      public interface ICtpNet
10     {
11         #region 属性
12
13         /// <summary>
14         /// 是否已登录交易服务器
15         /// </summary>
16         bool IsTradeServerLogined { get; }
17
18         /// <summary>
19         /// 是否已完成初始化
20         /// </summary>
21         bool IsInitialized { get; }
22
23         /// <summary>
24         /// 是否已清理组件
25         /// </summary>
26         bool IsReleased { get; }
27
28         /// <summary>
29         /// 上期所时间
30         /// </summary>
31         DateTime ShfeTime { get; }
32
33         /// <summary>
34         /// 郑商所时间
35         /// </summary>
36         DateTime CzceTime { get; }
37
38         /// <summary>
39         /// 大商所时间
40         /// </summary>
41         DateTime DceTime { get; }
42
43         /// <summary>
44         /// 中金所时间
45         /// </summary>
46         DateTime CffexTime { get; }
47
48         /// <summary>
49         /// 能交所时间
50         /// </summary>
51         DateTime IneTime { get; }
52
53         /// <summary>
54         /// 交易日
55         /// </summary>
56         DateTime TradingDay { get; }
57
58         /// <summary>
59         /// 资金帐户信息
60         /// </summary>
61         CtpNetTradingAccountField TradingAccount { get; }
62
63         /// <summary>
64         /// 是否已登录行情服务器
65         /// </summary>
66         bool IsMarketDataServerLogined { get; }
67
68         #endregion 属性
69
70         #region 函数
71

```

```

72     /// <summary>
73     /// 登录
74     /// </summary>
75     /// <param name="pLoginInfo">登录请求</param>
76     /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
77     CtpNetResponseCode Login(CtpNetReqUserLoginFieldEx pLoginInfo);
78
79     /// <summary>
80     /// 清理组件
81     /// </summary>
82     /// <remarks>如果不使用组件了,应当调用本方法以清理组件占用的系统资源。
83     /// 清理组件后如果想重新登录则需要重新创建一个组件实例</remarks>
84     void Release();
85
86     /// <summary>
87     /// 获取所有品种信息的字典,字典的键是品种代码
88     /// </summary>
89     Dictionary<string, CtpNetProductField> GetProducts();
90
91     /// <summary>
92     /// 获取所有合约信息的字典,字典的键是合约代码
93     /// </summary>
94     Dictionary<string, CtpNetInstrumentFieldEx> GetInstruments();
95
96     /// <summary>
97     /// 获取合约状态
98     /// </summary>
99     /// <param name="instrumentID">合约代码</param>
100    /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
101    CtpNetInstrumentStatusField GetInstrumentStatus(string instrumentID);
102
103    /// <summary>
104    /// 更新交易密码
105    /// </summary>
106    /// <param name="pRequest">请求参数</param>
107    /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
108    /// <remarks>请在更新交易密码响应事件中接收结果</remarks>
109    CtpNetResponseCode UpdateUserPassword(CtpNetUserPasswordUpdateField pRequest);
110
111    /// <summary>
112    /// 更新资金密码
113    /// </summary>
114    /// <param name="pRequest">请求参数</param>
115    /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
116    /// <remarks>请在更新资金密码响应事件中接收结果</remarks>
117    CtpNetResponseCode
118    UpdateTradingAccountPassword(CtpNetTradingAccountPasswordUpdateField
119    pRequest);
120
121    /// <summary>
122    /// 询价
123    /// </summary>
124    /// <param name="instrumentID">合约代码</param>
125    /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
126    CtpNetResponseCode InsertForQuote(string instrumentID);
127
128    /// <summary>
129    /// 执行宣告
130    /// </summary>
131    /// <param name="pInputExecOrder">执行宣告报单请求</param>
132    /// <returns>
133    /// 1. 大于等于零表示执行宣告报单请求发送成功,返回的是执行宣告报单序号;
134    /// 2. 小于零表示执行宣告报单请求发送失败,返回的是响应代码,请检查代码含义。
135    /// </returns>
136    int InsertExecOrder(CtpNetInputExecOrderFieldEx pInputExecOrder);
137
138    /// <summary>
139    /// 执行宣告操作
140    /// </summary>
141    /// <param name="field">执行宣告操作</param>
142    /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>

```

```

141 CtpNetResponseCode ExecOrderAction(CtpNetInputExecOrderActionField field);
142
143 /// <summary>
144 /// 撤销执行宣告
145 /// </summary>
146 /// <param name="field">执行宣告报单</param>
147 /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
148 CtpNetResponseCode CancelExecOrder(CtpNetExecOrderField field);
149
150 /// <summary>
151 /// 撤销执行宣告
152 /// </summary>
153 /// <param name="orderSn">执行宣告报单序号</param>
154 /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
155 CtpNetResponseCode CancelExecOrder(int orderSn);
156
157 /// <summary>
158 /// 获取执行宣告报单序号列表
159 /// </summary>
160 /// <param name="instrumentID">合约代码,空字符串为不指定</param>
161 /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
162 List<int> GetExecOrderSnList(string instrumentID = "");
163
164 /// <summary>
165 /// 获取执行宣告报单列表
166 /// </summary>
167 /// <param name="instrumentID">合约代码,空字符串为不指定</param>
168 /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
169 List<CtpNetExecOrderFieldEx> GetExecOrders(string instrumentID = "");
170
171 /// <summary>
172 /// 获取指定序号的执行宣告报单信息
173 /// </summary>
174 /// <param name="orderSn">执行宣告报单序号</param>
175 /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
176 CtpNetExecOrderFieldEx GetExecOrder(int orderSn);
177
178 /// <summary>
179 /// 报单
180 /// </summary>
181 /// <param name="pInputOrder">报单请求</param>
182 /// <returns>
183 /// 1. 大于等于零表示报单发送成功,返回的是报单序号;
184 /// 2. 小于零表示报单发送失败,返回的是响应代码,请检查代码含义.
185 /// </returns>
186 /// <remarks>SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持</remarks>
187 int InsertOrder(CtpNetInputOrderFieldEx pInputOrder);
188
189 /// <summary>
190 /// 报单
191 /// </summary>
192 /// <param name="instrumentID">合约代码</param>
193 /// <param name="volume">数量</param>
194 /// <param name="price">价格</param>
195 /// <param name="direction">买卖方向</param>
196 /// <param name="offsetFlag">开平标志</param>
197 /// <param name="orderPriceType">报单价格条件类型</param>
198 /// <param name="contingentCondition">触发条件类型</param>
199 /// <param name="stopPrice">条件价</param>
200 /// <param name="volumeCondition">成交量类型</param>
201 /// <param name="minVolume">最小成交量</param>
202 /// <param name="timeCondition">有效期类型</param>
203 /// <returns>
204 /// 1. 大于等于零表示报单发送成功,返回的是报单序号;
205 /// 2. 小于零表示报单发送失败,返回的是响应代码,请检查代码含义.
206 /// </returns>
207 /// <remarks>SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持</remarks>
208 int InsertOrder(
209     string instrumentID,
210     int volume,
211     double price,

```

```

212         CtpNetDirectionType direction,
213         CtpNetOffsetFlagType offsetFlag,
214         CtpNetOrderPriceType orderPriceType =
215             CtpNetOrderPriceType.LimitPrice,
216         CtpNetContingentConditionType contingentCondition =
217             CtpNetContingentConditionType.Immediately,
218         double stopPrice = 0.0,
219         CtpNetVolumeConditionType volumeCondition = CtpNetVolumeConditionType.AV,
220         int minVolume = 1,
221         CtpNetTimeConditionType timeCondition = CtpNetTimeConditionType.GFD);
222
223     /// <summary>
224     /// 多头开仓
225     /// </summary>
226     /// <param name="instrumentID">合约代码</param>
227     /// <param name="volume">数量</param>
228     /// <param name="price">价格,零为市价</param>
229     /// <returns>
230     /// 1. 大于等于零表示报单发送成功,返回的是报单序号;
231     /// 2. 小于零表示报单发送失败,返回的是响应代码,请检查代码含义.
232     /// </returns>
233     /// <remarks>
234     /// 1. SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持;
235     /// 2. 上期所合约的市价单转换为停板价的限价单然后立即撤单.
236     /// </remarks>
237     int OpenLong(string instrumentID, int volume, double price);
238
239     /// <summary>
240     /// 空头开仓
241     /// </summary>
242     /// <param name="instrumentID">合约代码</param>
243     /// <param name="volume">数量</param>
244     /// <param name="price">价格,零为市价</param>
245     /// <returns>
246     /// 1. 大于等于零表示报单发送成功,返回的是报单序号;
247     /// 2. 小于零表示报单发送失败,返回的是响应代码,请检查代码含义.
248     /// </returns>
249     /// <remarks>
250     /// 1. SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持;
251     /// 2. 上期所合约的市价单转换为停板价的限价单然后立即撤单.
252     /// </remarks>
253     int OpenShort(string instrumentID, int volume, double price);
254
255     /// <summary>
256     /// 多头平仓
257     /// </summary>
258     /// <param name="instrumentID">合约代码</param>
259     /// <param name="volume">数量</param>
260     /// <param name="price">价格,零为市价</param>
261     /// <returns>
262     /// 1. 大于等于零表示报单发送成功,返回的是报单序号;
263     /// 2. 等于零表示上期所合约平仓数量中包含今仓和昨仓,指令被分拆为平今和平昨两条指令发送;
264     /// 3. 小于零表示报单发送失败,返回的是响应代码,请检查代码含义.
265     /// </returns>
266     /// <remarks>
267     /// 1. SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持;
268     /// 2. 上期所合约的市价单转换为停板价的限价单然后立即撤单;
269     /// 3. 上期所合约不区分平今平昨,优先平今,如果想先平昨,请使用InsertOrder指令
270     /// </remarks>
271     int CloseLong(string instrumentID, int volume, double price);
272
273     /// <summary>
274     /// 多头平仓
275     /// </summary>
276     /// <param name="instrumentID">合约代码</param>
277     /// <param name="volume">数量</param>
278     /// <param name="price">价格,零为市价</param>
279     /// <param
name="closeTodayFirst">上期所有今仓和昨仓时,是否先平今,为否则先平昨</param>

```

```

278      /// <param
name="orderSnList">报单序号列表,如果某个序号小于一则表示该报单指令发送失败</pa
ram>
279      /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
280      /// <remarks>
281      /// 1.SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持;
282      /// 2.上期所合约的市价单转换为停板价的限价单然后立即撤单;
283      /// </remarks>
284      CtpNetResponseCode CloseLong(string instrumentID, int volume, double price,
bool closeTodayFirst, out List<int> orderSnList);
285
286      /// <summary>
287      /// 空头平仓
288      /// </summary>
289      /// <param name="instrumentID">合约代码</param>
290      /// <param name="volume">数量</param>
291      /// <param name="price">价格,零为市价</param>
292      /// <returns>
293      /// 1.大于等于零表示报单发送成功,返回的是报单序号;
294      ///
2.等于零表示上期所合约平仓数量中包含今仓和昨仓,指令被分拆为平今和平昨两条指令
发送;
295      /// 3.小于零表示报单发送失败,返回的是响应代码,请检查代码含义.
296      /// </returns>
297      /// <remarks>
298      /// 1.SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持;
299      /// 2.上期所合约的市价单转换为停板价的限价单然后立即撤单;
300      /// 3.上期所合约不区分平今平昨,优先平今,如果想先平昨,请使用InsertOrder指令
301      /// </remarks>
302      int CloseShort(string instrumentID, int volume, double price);
303
304      /// <summary>
305      /// 空头平仓
306      /// </summary>
307      /// <param name="instrumentID">合约代码</param>
308      /// <param name="volume">数量</param>
309      /// <param name="price">价格,零为市价</param>
310      /// <param
name="closeTodayFirst">上期所有今仓和昨仓时,是否先平今,为否则先平昨</param>
311      /// <param
name="orderSnList">报单序号列表,如果某个序号小于一则表示该报单指令发送失败</pa
ram>
312      /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
313      /// <remarks>
314      /// 1.SimNow模拟交易中大连,郑州,中金,不支持市价,但实盘支持;
315      /// 2.上期所合约的市价单转换为停板价的限价单然后立即撤单;
316      /// </remarks>
317      CtpNetResponseCode CloseShort(string instrumentID, int volume, double price,
bool closeTodayFirst, out List<int> orderSnList);
318
319      /// <summary>
320      /// 报单操作
321      /// </summary>
322      /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
323      CtpNetResponseCode OrderAction(CtpNetInputOrderActionField field);
324
325      /// <summary>
326      /// 撤单
327      /// </summary>
328      /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
329      CtpNetResponseCode CancelOrder(CtpNetOrderField field);
330
331      /// <summary>
332      /// 撤单
333      /// </summary>
334      /// <param name="orderSn">报单序号</param>
335      /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
336      CtpNetResponseCode CancelOrder(int orderSn);
337
338      /// <summary>
339      /// 批量撤单

```



```

340     /// </summary>
341     /// <param name="instrumentID">合约代码,空字符串为不指定</param>
342     /// <param name="direction">买卖方向,零为不指定</param>
343     /// <param
name="offsetFlag">开平标志,零为不指定,上期所合约使用Close平仓标志时不区分今仓
和昨仓</param>
344     /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
345     CtpNetResponseCode CancelOrderEx(
346         string instrumentID = "",
347         CtpNetDirectionType direction = 0,
348         CtpNetOffsetFlagType offsetFlag = 0);
349
350     /// <summary>
351     /// 报单是否处于可撤状态
352     /// </summary>
353     /// <param name="orderSn">报单序号</param>
354     bool IsOrderOpen(int orderSn);
355
356     /// <summary>
357     /// 获取指定条件的报单序号列表
358     /// </summary>
359     /// <param name="instrumentID">合约代码,空字符串为不指定</param>
360     /// <param name="direction">买卖方向,零为不指定</param>
361     /// <param
name="offsetFlag">开平标志,零为不指定,上期所合约使用Close平仓标志时不区分今仓
和昨仓</param>
362     /// <param name="orderStatus">报单状态,零为不指定,一为还在挂单</param>
363     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
364     List<int> GetOrderSnList(
365         string instrumentID = "",
366         CtpNetDirectionType direction = 0,
367         CtpNetOffsetFlagType offsetFlag = 0,
368         CtpNetOrderStatusType orderStatus = 0);
369
370     /// <summary>
371     /// 获取指定条件的报单列表
372     /// </summary>
373     /// <param name="instrumentID">合约代码,空字符串为不指定</param>
374     /// <param name="direction">买卖方向,零为不指定</param>
375     /// <param
name="offsetFlag">开平标志,零为不指定,上期所合约使用Close平仓标志时不区分今仓
和昨仓</param>
376     /// <param name="orderStatus">报单状态,零为不指定,一为还在挂单</param>
377     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
378     List<CtpNetOrderFieldEx> GetOrders(
379         string instrumentID = "",
380         CtpNetDirectionType direction = 0,
381         CtpNetOffsetFlagType offsetFlag = 0,
382         CtpNetOrderStatusType orderStatus = 0);
383
384     /// <summary>
385     /// 获取指定序号的报单信息
386     /// </summary>
387     /// <param name="orderSn">报单序号</param>
388     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
389     CtpNetOrderFieldEx GetOrder(int orderSn);
390
391     /// <summary>
392     /// 获取指定条件的成交记录列表
393     /// </summary>
394     /// <param name="instrumentID">合约代码,空字符串为不指定</param>
395     /// <param name="direction">买卖方向,零为不指定</param>
396     /// <param
name="offsetFlag">开平标志,零为不指定,上期所合约使用Close平仓标志时不区分今仓
和昨仓</param>
397     /// <param name="hedgeFlag">投机套保标志,零为不指定</param>
398     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
399     List<CtpNetTradeFieldEx> GetTrades(
400         string instrumentID = "",
401         CtpNetDirectionType direction = 0,
402         CtpNetOffsetFlagType offsetFlag = 0,

```

```

403         CtpNetHedgeFlagType hedgeFlag = 0);
404
405     /// <summary>
406     /// 获取指定条件的持仓信息列表
407     /// </summary>
408     /// <param name="instrumentID">合约代码,空字符串为不指定</param>
409     /// <param name="direction">买卖方向,零为不指定</param>
410     /// <param name="hedgeFlag">投机套保标志,零为不指定</param>
411     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
412     List<CtpNetInvestorPositionFieldEx> GetPositions(
413         string instrumentID = "",
414         CtpNetDirectionType direction = 0,
415         CtpNetHedgeFlagType hedgeFlag = 0);
416
417     /// <summary>
418     /// 获取指定合约的持仓信息
419     /// </summary>
420     /// <param name="instrumentID">合约代码</param>
421     /// <param name="direction">买卖方向</param>
422     /// <param name="hedgeFlag">投机套保标志,不指定则为投机</param>
423     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
424     CtpNetInvestorPositionFieldEx GetPosition(
425         string instrumentID,
426         CtpNetDirectionType direction,
427         CtpNetHedgeFlagType hedgeFlag = CtpNetHedgeFlagType.Speculation);
428
429     /// <summary>
430     /// 获取指定合约的持仓明细列表
431     /// </summary>
432     /// <param name="instrumentID">合约代码</param>
433     /// <param name="direction">买卖方向</param>
434     /// <param name="hedgeFlag">投机套保标志,不指定则为投机</param>
435     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
436     List<CtpNetInvestorPositionDetailFieldEx> GetPositionDetails(
437         string instrumentID,
438         CtpNetDirectionType direction,
439         CtpNetHedgeFlagType hedgeFlag = CtpNetHedgeFlagType.Speculation);
440
441     /// <summary>
442     /// 获取所有银行开销户信息
443     /// </summary>
444     List<CtpNetAccountRegisterFieldEx> GetBankAccountRegisters();
445
446     /// <summary>
447     /// 查询银行余额
448     /// </summary>
449     /// <param name="pRequest">查询请求参数</param>
450     /// <remarks>请在查询银行余额响应事件中接收结果</remarks>
451     /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
452     CtpNetResponseCode QueryBankAccountMoney(CtpNetReqQueryAccountField pRequest);
453
454     /// <summary>
455     /// 银行资金转期货
456     /// </summary>
457     /// <param name="pRequest">转账请求参数</param>
458     /// <remarks>请在银行资金转期货响应事件中接收结果</remarks>
459     /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
460     CtpNetResponseCode FromBankToFuture(CtpNetReqTransferField pRequest);
461
462     /// <summary>
463     /// 期货资金转银行
464     /// </summary>
465     /// <param name="pRequest">转账请求参数</param>
466     /// <remarks>请在期货资金转银行响应事件中接收结果</remarks>
467     /// <returns>响应代码(一个整数),NONE(零)表示成功,否则请检查代码含义</returns>
468     CtpNetResponseCode FromFutureToBank(CtpNetReqTransferField pRequest);
469
470     /// <summary>
471     /// 查询合约保证金率
472     /// </summary>
473     /// <param name="pRequest">查询请求参数</param>

```

```

474     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
475     /// <remarks>
476     /// 1.内部会缓存当前交易日成功查询的结果,跨越交易日重新登录后将清空缓存;
477     ///
478     2.获取失败则自动向期货公司查询,可以在查询响应事件中接收结果,或者稍后再重复查询
479     .
480     /// </remarks>
481     CtpNetInstrumentMarginRateField
482     QueryMarginRate(CtpNetQryInstrumentMarginRateField pRequest);
483
484     /// <summary>
485     /// 查询合约手续费率
486     /// </summary>
487     /// <param name="pRequest">查询请求参数</param>
488     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
489     /// <remarks>
490     /// 1.内部会缓存当前交易日成功查询的结果,跨越交易日重新登录后将清空缓存;
491     ///
492     2.获取失败则自动向期货公司查询,可以在查询响应事件中接收结果,或者稍后再重复查询
493     .
494     /// </remarks>
495     CtpNetInstrumentCommissionRateField
496     QueryCommissionRate(CtpNetQryInstrumentCommissionRateField pRequest);
497
498     /// <summary>
499     /// 查询合约报单手续费率
500     /// </summary>
501     /// <param name="pRequest">查询请求参数</param>
502     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
503     /// <remarks>
504     /// 1.内部会缓存当前交易日成功查询的结果,跨越交易日重新登录后将清空缓存;
505     ///
506     2.获取失败则自动向期货公司查询,可以在查询响应事件中接收结果,或者稍后再重复查询
507     .
508     /// </remarks>
509     CtpNetInstrumentOrderCommRateField
510     QueryOrderCommissionRate(CtpNetQryInstrumentOrderCommRateField pRequest);
511
512     /// <summary>
513     /// 查询期权合约手续费率
514     /// </summary>
515     /// <param name="pRequest">查询请求参数</param>
516     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
517     /// <remarks>
518     /// 1.内部会缓存当前交易日成功查询的结果,跨越交易日重新登录后将清空缓存;
519     ///
520     2.获取失败则自动向期货公司查询,可以在查询响应事件中接收结果,或者稍后再重复查询
521     .
522     /// </remarks>
523     CtpNetOptionInstrCommRateField
524     QueryOptionCommissionRate(CtpNetQryOptionInstrCommRateField pRequest);
525
526     /// <summary>
527     /// 查询期权合约手续费率
528     /// </summary>
529     /// <param name="pRequest">查询请求参数</param>
530     /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
531     /// <remarks>
532     /// 1.内部会缓存当前交易日成功查询的结果,跨越交易日重新登录后将清空缓存;
533     ///
534     2.获取失败则自动向期货公司查询,可以在查询响应事件中接收结果,或者稍后再重复查询
535     .
536     /// </remarks>
537     CtpNetOptionInstrTradeCostField
538     QueryOptionTradeCost(CtpNetQryOptionInstrTradeCostField pRequest);
539
540     /// <summary>
541     /// 朗读引擎名称
542     /// </summary>
543     List<string> GetSpeakerNames();

```



```

530      /// <summary>
531      /// 设置朗读参数
532      /// </summary>
533      /// <param name="speakerName">朗读引擎名称,空字符串表示第一个朗读引擎</param>
534      /// <param name="volume">朗读音量 [范围 0 ~ 100]</param>
535      /// <param name="rate">朗读频率 [范围 -10 ~ 10]</param>
536      ///
537      <returns>响应代码(一个整数), NONE(零)表示成功, 否则请检查代码含义</returns>
538      CtpNetResponseCode SetSpeakParameter(string speakerName = "", int volume =
539      100, int rate = 0);
540
541      /// <summary>
542      /// 语音朗读
543      /// </summary>
544      /// <param name="text">要朗读的文本</param>
545      /// <param name="cancleBefore">是否取消之前没有读完的语音,默认是,
546      /// 如果不取消且之前还在朗读则多段语音串行朗读</param>
547      ///
548      <returns>响应代码(一个整数), NONE(零)表示成功, 否则请检查代码含义</returns>
549      CtpNetResponseCode Speak(string text, bool cancleBefore = true);
550
551      /// <summary>
552      /// 获取行情
553      /// </summary>
554      /// <param name="instrumentID">合约代码</param>
555      /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
556      /// <remarks>如果查询失败且没有订阅行情则自动订阅</remarks>
557      CtpNetDepthMarketDataField GetMarketData(string instrumentID);
558
559      /// <summary>
560      /// 获取行情
561      /// </summary>
562      /// <param name="instrumentID">合约代码</param>
563      /// <param name="autoSubscribe">是否自动订阅行情</param>
564      /// <returns>成功返回具体的值,失败返回null(VB中的Nothing)</returns>
565      CtpNetDepthMarketDataField GetMarketData(string instrumentID, bool
566      autoSubscribe);
567
568      /// <summary>
569      /// 是否已订阅行情
570      /// </summary>
571      /// <param name="instrumentID">合约代码</param>
572      bool IsMarketDataSubscribed(string instrumentID);
573
574      /// <summary>
575      /// 订阅行情
576      /// </summary>
577      /// <param name="instrumentIDs">合约代码数组(区分大小写)</param>
578      CtpNetResponseCode SubscribeMarketData(params string[] instrumentIDs);
579
580      /// <summary>
581      /// 订阅行情
582      /// </summary>
583      /// <param name="instrumentIDs">合约代码数组(区分大小写)</param>
584      CtpNetResponseCode SubscribeMarketData(IEnumerable<string> instrumentIDs);
585
586      /// <summary>
587      /// 订阅行情
588      /// </summary>
589      /// <param name="instrumentID">合约代码</param>
590      CtpNetResponseCode SubscribeMarketData(string instrumentID);
591
592      /// <summary>
593      /// 退订行情
594      /// </summary>
595      /// <param name="instrumentIDs">合约代码数组(区分大小写)</param>
596      CtpNetResponseCode UnsubscribeMarketData(params string[] instrumentIDs);
597
598      /// <summary>
599      /// 退订行情
600      /// </summary>

```

```

597     /// <param name="instrumentIDs">合约代码数组(区分大小写)</param>
598     CtpNetResponseCode UnsubscribeMarketData(IEnumerable<string> instrumentIDs);
599
600     /// <summary>
601     /// 退订行情
602     /// </summary>
603     /// <param name="instrumentID">合约代码</param>
604     CtpNetResponseCode UnsubscribeMarketData(string instrumentID);
605
606     /// <summary>
607     /// (内部使用)
608     /// </summary>
609     void SetAssembly(System.Reflection.Assembly[] assm);
610
611     /// <summary>
612     /// (内部使用)
613     /// </summary>
614     void SetCurrentDirectory(string dir);
615
616     #endregion 函数
617
618     #region 事件
619
620     /// <summary>
621     /// 与行情服务器建立连接
622     /// </summary>
623     event Action OnMarketDataServerConnected;
624
625     /// <summary>
626     /// 与行情服务器断开连接
627     /// </summary>
628     event OnMarketDataServerDisconnectedCtpNetDelegate OnMarketDataServerDisconnected;
629
630     /// <summary>
631     /// 与交易服务器建立连接
632     /// </summary>
633     event Action OnTradeServerConnected;
634
635     /// <summary>
636     /// 与交易服务器断开连接
637     /// </summary>
638     event OnTradeServerDisconnectedCtpNetDelegate OnTradeServerDisconnected;
639
640     /// <summary>
641     /// 初始化响应
642     /// </summary>
643     event Action OnInitialized;
644
645     /// <summary>
646     /// 组件清理响应
647     /// </summary>
648     event OnMarketDataServerDisconnectedCtpNetDelegate OnReleased;
649
650     /// <summary>
651     /// 程序日志
652     /// </summary>
653     event OnLogCtpNetDelegate OnLog;
654
655     /// <summary>
656     /// 交易通知
657     /// </summary>
658     event OnNoticeCtpNetDelegate OnTradingNotice;
659
660     /// <summary>
661     /// 资金信息变化
662     /// </summary>
663     event OnTradingAccountCtpNetDelegate OnTradingAccount;
664
665     /// <summary>
666     /// 合约状态通知

```

```
667     /// </summary>
668     event OnInstrumentStatusCtpNetDelegate OnInstrumentStatus;
669
670     /// <summary>
671     /// 行情通知
672     /// </summary>
673     event OnMarketDataCtpNetDelegate OnMarketData;
674
675     /// <summary>
676     /// 报单通知
677     /// </summary>
678     event OnOrderCtpNetDelegate OnOrder;
679
680     /// <summary>
681     /// 成交通知
682     /// </summary>
683     event OnTradeCtpNetDelegate OnTrade;
684
685     /// <summary>
686     /// 持仓变化
687     /// </summary>
688     event OnPositionCtpNetDelegate OnPosition;
689
690     /// <summary>
691     /// 询价响应(询价失败)
692     /// </summary>
693     event OnInsertForQuoteCtpNetDelegate OnInsertForQuote;
694
695     /// <summary>
696     /// 执行宣告通知
697     /// </summary>
698     event OnExecOrderCtpNetDelegate OnExecOrder;
699
700     /// <summary>
701     /// 查询合约保证金率响应
702     /// </summary>
703     event OnQueryMarginRateCtpNetDelegate OnQueryMarginRate;
704
705     /// <summary>
706     /// 查询合约手续费率响应
707     /// </summary>
708     event OnQueryCommissionRateCtpNetDelegate OnQueryCommissionRate;
709
710     /// <summary>
711     /// 查询合约报单手续费率响应
712     /// </summary>
713     event OnQueryOrderCommissionRateCtpNetDelegate OnQueryOrderCommissionRate;
714
715     /// <summary>
716     /// 查询期权合约手续费率响应
717     /// </summary>
718     event OnQueryOptionCommissionRateCtpNetDelegate OnQueryOptionCommissionRate;
719
720     /// <summary>
721     /// 查询期权交易成本响应
722     /// </summary>
723     event OnQueryOptionTradeCostCtpNetDelegate OnQueryOptionTradeCost;
724
725     /// <summary>
726     /// 更新交易密码响应
727     /// </summary>
728     event OnUpdateUserPasswordCtpNetDelegate OnUpdateUserPassword;
729
730     /// <summary>
731     /// 更新资金密码响应
732     /// </summary>
733     event OnUpdateTradingAccountPasswordCtpNetDelegate
734     OnUpdateTradingAccountPassword;
735
736     /// <summary>
737     /// 查询银行余额响应
```

```
737         /// </summary>
738         event OnQueryBankAccountMoneyCtpNetDelegate OnQueryBankAccountMoney;
739
740         /// <summary>
741         /// 银行资金转期货响应
742         /// </summary>
743         event OnFromBankToFutureCtpNetDelegate OnFromBankToFuture;
744
745         /// <summary>
746         /// 期货资金转银行响应
747         /// </summary>
748         event OnFromFutureToBankCtpNetDelegate OnFromFutureToBank;
749
750         #endregion 事件
751     }
752 }
753
```