Choice数据

Python接口使用手册

V2.4.2.0

电话：400-620-1818　　邮箱：Choiceinfo@eastmoney.com

接口官网：http://quantapi.eastmoney.com/

地址：上海市徐汇区龙田路190号东方财富大厦

升级公告

版本号	版本变动信息	发布时间
2.4.2.0	V2.4.1.1 -> V2.4.2.0	2020-07-03

1. 接口登录服务优化
2. csq、csqsnapshot、cst、cmc函数创业板改造
3. 交易状态字段增加枚举值14-无效查询，即证券品种不支持该字段查询
4. R语言支持R 4.0.0
5. 升级指标http://quantapi.eastmoney.com/About/NoticeIndex?version=2.4.2.0
6. 其他bug修复

目录

接口配置

接口配置

文件存放位置

在量化接口官网（http://quantapi.eastmoney.com/）下载压缩包EmQuantAPI_Python.zip,下载完成后解压。其中：

- installEmQuantAPI.py用于注册python接口；
- libs含接口的库文件，以及接口登录激活工具文件，通过手机号获取验证码，自动生成令牌用以登录；
- demo.py为示例脚本文件
- EmQuantAPI.py文件中有一个DemoCallback函数，为报价回调函数的实例，可参考使用；
- EmQuantAPI.py文件中有一个cstCallBack函数，为日内跳价回调函数的实例，可参考使用

EMQuantAPI Python接口配置的系统环境要求与方法

系统环境要求

- Windows 系统，支持32位和64位系统，以及Microsoft Visual C++ 2010 可再发行组件包（可在电脑"程序和功能"中查看是否安装，下载链接如下）

  32位：https://www.microsoft.com/zh-cn/download/confirmation.aspx?id=5555

  64位：https://www.microsoft.com/zh-CN/download/details.aspx?id=14632

- Centos, Ubuntu，支持32位和64位系统；
- Mac OS 系统，支持64位系统，以及gtk+3.0环境（激活工具使用）
- Python版本： 2.6.x, 2.7.x, 3.x

配置方法

- 在命令行运行 installEmQuantAPI.py 注册，若使用Anaconda编译器，请在编译器中运行installEmQuantAPI.py注册.
- 绑定手机号

  登录Choice量化接口网站主页（http://quantapi.eastmoney.com），点击右上角账户名-个人资料绑定手机号；或者登录Choice金融终端，进入用户中心-资料管理绑定手机号。

- 登录激活

根据所用系统环境，运行接口激活工具LoginActivator.exe（Windows）、loginactivator_mac（Mac）、loginactivator（Linux）/loginactivator_ubuntu（Ubuntu），输入绑定手机号获取验证码，激活成功后生成令牌文件userInfo，用户使用时无需输入用户名和密码，默认从令牌中获取登录权限。一个账号最多支持在十台设备上激活。

登录函数

```
start(options, logcallback, mainCallBack)
```

初始化登录函数，登录验证通过以后，即可正常使用接口函数获取数据

参数

参数名	简称	定义	输入/输出	描述
options	可选参数	字符串	输入	附加参数，可填附加字段。见**附注1**
logcallback	可选参数	c_LogCallback	输入	日志回调函数
mainCallBack	可选参数	c_DataCallback	输入	主回调函数

返回

类型	描述

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例(Python2.x)

```
from EmQuantAPI import *
loginresult = c.start( )
#loginresult为c.EmQuantData类型数据
print loginresult
```

范例(Python3.x)

```
from EmQuantAPI import *
loginresult = c.start( )
#loginresult为c.EmQuantData类型数据
print (loginresult)
```

附注1 登录函数可选参数列表:

中文名称	英文名称	取值范围	说明
服务器测速	TestLatency	0,1; 缺省值 0	取值0，不测速，连接默认服务器； 取值1，登录前服务器测速，并保存为默认
强制登录	ForceLogin	0,1; 缺省值 0	取值0，当线上已存在该账户时，不强制登录； 取值1，当线上已存在该账户时，强制登录，将前一位在线用户踢下线；
记录登录信息标记	RecordLoginInfo	0,1; 缺省值 1	取值0，不记录; 取值1，追加记录登录信息到logininfo.log文件，文件位于serverlist.json.e所在目录下

退出函数

```
stop ()
```

退出登录

无参数

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例(Python2.x)

```
from EmQuantAPI import *
loginresult = c.start()
print loginresult
#do something…
logoutresult = c.stop()
#logoutresult为c.EmQuantData类型数据
print logoutresult
```

范例(Python3.x)

```
from EmQuantAPI import *
loginresult = c.start()
print (loginresult)
#do something…
logoutresult = c.stop()
#logoutresult为c.EmQuantData类型数据
print (logoutresult)
```

数据函数

截面函数

```
css(codes,indicators,options=None,*arga,**argb)
```

获取股票，指数，基金，期货等各个证券品种或组合的基本资料，财务，估值等截面数据 （需授权）

参数名	简称	定义		描述

参数名	简称	定义		描述
codes	证券代码	字符串或者序列	输入	东财代码，支持多代码输入，以半角逗号分隔，**不支持跨品种证券输入**
indicators	指标简称	字符串或者序列	输入	指标名称，支持多指标输入，以半角逗号分隔，**最多不超过64个**，详细指标列表见指标手册
options	可选参数	字符串	输入	附加参数，可填附加字段，见**附注2**
arga	可选参数	可变参数	预留	
argb	可选参数	可变参数	预留	

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #**本地日期**
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
data = c.css("300059.SZ","open,close","tradedate=20190827")
if data.ErrorCode != 0:
    print "request css Error, ", data.ErrorMsg
else:
    for code in data.Codes:
        for i in range(0,len(data.Indicators)):
            print data.Data[code][i]
```

范例(Python3.x)

```
data = c.css("300059.SZ","open,close","tradedate=20190819")
if data.ErrorCode != 0:
    print("request css Error, ", data.ErrorMsg)
else:
    for code in data.Codes:
        for i in range(0,len(data.Indicators)):
            print(data.Data[code][i])
```

附注2 截面函数可选参数列表：

中文名称	英文名称	取值范围	说明
是否输出pandas格式	Ispandas	0--1 缺省值：0	非pandas格式--0 pandas格式--1 需要安装pandas包
pandas索引	RowIndex	1--2 缺省值：1	证券代码--1 日期--2；
空值替换	ShowBlank	整数	对返回数据中的空值的进行特殊处理 例如：ShowBlank=0，所有的空值都替换成0

（注：截面函数每分钟请求次数不能超过700次）

序列函数

```
csd(codes,indicators,startdate,enddate,options=None,*arga,**argb)
```

获取股票，指数，基金，期货等各个证券品种或组合的日频历史序列数据 （需授权）

参数

参数名	参数简称	定义	描述
codes	证券代码	字符串或者序列	东财代码，支持多代码输入，以半角逗号分隔，**不支持跨品种证券输入**
indicators	指标简称	字符串或者序列	指标名称，支持多指标输入，以半角逗号分隔，**最多不超过64个**，详细指标列表见指标手册
startdate	起始日期	字符串或者datetime	支持格式： YYYYMMDD，YYYY/MM/DD，YYYY/M/D，YYYY-MM-DD，YYYY-M-D
EndDate	截止日期	字符串或者datetime	支持格式： YYYYMMDD，YYYY/MM/DD，YYYY/M/D，YYYY-MM-DD，YYYY-M-D

参数名	参数简称	定义	描述
options	可选参数	字符串	附加参数，可填附加字段，见**附注3**
arga	可选参数	可变参数	预留
argb	可选参数	可变参数	预留

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
data = c.csd("300059.SZ,600425.SH","open,close","2016-07-01","2016-07-06", "")
if (data.ErrorCode != 0):
        print "request csd Error, ", data.ErrorMsg
else:
    for code in data.Codes:
        for i in range(0, len(data.Indicators)):
            for j in range(0, len(data.Dates)):
                print data.Data[code][i][j]
```

范例(Python3.x)

```
data = c.csd("300059.SZ,600425.SH","open,close","20160701","20160706","")
if(data.ErrorCode != 0):
    print("request csd Error, ", data.ErrorMsg)
else:
    for code in data.Codes:
        for i in range(0, len(data.Indicators)):
            for j in range(0, len(data.Dates)):
                print(data.Data[code][i][j])
```

附注3 序列函数可选参数列表:

中文名称	英文名称	取值范围	说明

中文名称	英文名称	取值范围	说明
是否输出pandas格式	Ispandas	0--1 缺省值：0	非pandas格式--0; pandas格式--1; 需要安装pandas包;
pandas索引	RowIndex	1--2 缺省值：1	证券代码--1; 日期--2;
日期周期	Period	1--4 缺省值：1	日期周期：日，周，月，年 分别对应：１，２，３，４
复权方式	AdjustFlag	1--3 缺省值：1	不复权--1 后复权--2 前复权--3
币种	CurType	1--4 缺省值：1	原始币种--1 人民币--2 美元--3 港币--4 (仅适用于港美股指标)
按日期排序	Order	1--2 缺省值：1	升序--1 降序--2
市场类型	Market	见说明 缺省值： “CNSESH”	CNSESH 上海证券交易所 CNSESZ 深圳证券交易所 HKSE00 香港证券交易所 USSE00 美国证券交易所 USSEND 美国纳斯达克市场 USSENY 纽约证券交易所 CNFEBC 渤海商品交易所 CNFEDC 大连商品交易所 CNFESF 上海期货交易所 CNFEZC 郑州商品交易所 INE000 上海国际能源交易中心 CNGCSH 上海黄金交易所 HKME00 香港商品交易所 0 自然日 CNSH00 沪股通交易日 CNSHHK 沪港股通交易日 CNSZ00 深股通交易日 CNSZHK 深港股通交易日 NYMEX0 纽约商业期货交易所 USFENY 纽约商品交易所 CME000 芝加哥商业交易所 LDMETL 伦敦金属交易所 LDEXCH 伦敦证券交易所 SGSE00 新加坡交易所

中文名称	英文名称	取值范围	说明
空值替换	ShowBlank	整数	对返回数据中的空值的进行特殊处理，例如：ShowBlank=0，所有的空值都替换成0

（注：序列函数每分钟请求次数不能超过700次）

历史分钟

```
cmc(codes,indicators,startdate,enddate,options=None,*arga,**argb)
```

获取最近30个自然日沪深股票的历史分钟K线序列数据（需授权）

参数

参数名	参数简称	定义	描述
codes	证券代码	字符串或者序列	东财代码，**只支持单个代码**
indicators	指标简称	字符串或者序列	指标名称，支持多指标输入，以半角逗号分隔，详细指标列表见指标手册
startdate	起始日期	字符串或者datetime	支持格式： YYYYMMDD，YYYY/MM/DD，YYYY/M/D，YYYYMM-DD，YYYY-M-D，YYYYMMDDHHMMSS
EndDate	截止日期	字符串或者datetime	支持格式： YYYYMMDD，YYYY/MM/DD，YYYY/M/D，YYYYMM-DD，YYYY-M-D，YYYYMMDDHHMMSS
options	可选参数	字符串	附加参数，可填附加字段，见**附注4**
arga	可选参数	可变参数	
argb	可选参数	可变参数	

返回

类型	描述

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = list() #数据结果

范例(Python2.x)

```
from datetime import timedelta, datetime
#请求300059.SZ以今天为截止日的最近7个自然日的分钟数据
data = c.cmc("300059.SZ", "OPEN,CLOSE,HIGH",
(datetime.today()+timedelta(-6)).strftime("%Y-%m-
%d"),datetime.today().strftime("%Y-%m-%d"), "IsHistory=1")

#输出示例
for i in range(0,len(data.Indicators)):
    for j in range(0, len(data.Dates)):
        print "indicator=%s, value=%s" % (data.Indicators[i],
str(data.Data[i][j]))
```

范例(Python3.x)

```
from datetime import timedelta, datetime
#请求300059.SZ以今天为截止日的最近7个自然日的分钟数据
data = c.cmc("300059.SZ","OPEN,CLOSE,HIGH", (datetime.today() +
timedelta(-6)).strftime("%Y-%m-%d"),datetime.today().strftime("%Y-%m-%d"),
"IsHistory=1")

#请求300059.SZ在2019-08-19 9:40到2019-08-19 9:55的分钟数据
#仅支持获取最近30个自然日
data = c.cmc("300059.SZ",
"Date,Time,High,Open,Low,Close,Volume,Amount,Change,PctChange","20190819094000
","20190819095500", "IsHistory=1")

#输出示例
for i in range(0,len(data.Indicators)):
    for j in range(0, len(data.Dates)):
        print("indicator=%s, value=%s" % (data.Indicators[i],str(data.Data[i]
[j])))
```

附注4 历史分钟K线函数函数可选参数列表：

中文名称	英文名称	取值范围	说明
是否输出pandas格式	Ispandas	0，1 缺省值：0	非pandas格式--0 pandas格式--1 需要安装pandas包
pandas索引	RowIndex	1，2 缺省值：1	证券代码--1 日期和时间--2
日期周期	Period	正整数 缺省值：1	单位，分钟，取值为1至 240
复权方式	AdjustFlag	1，2，3 缺省值：1	不复权--1 后复权--2 前复权--3
是否包含历史数据	IsHistory	0，1 缺省值：1	历史分钟--1 当日分钟--0
复权基期	BaseDate	取值>=截止日期 缺省值：当天日期	支持YYYYMMDD,YYYY-MM-DD,YYYY/MM/DD， 例：20160101

（注：交易所盘后会推送修正数据，当日分钟最后一笔数据根据最新修正数据变动）

报价订阅

```
csq(codes,indicators,options=None,fncallback=None,userparams=None,*arga,**argb
)
```

提供各个证券品种的报价数据订阅（需授权）

参数

参数名	简称	定义	描述
codes	证券代码	字符串或者序列	东财代码，支持多代码输入，以半角逗号分隔
indicators	指标简称	字符串或者序列	指标名称，支持多指标输入，以半角逗号分隔，详细指标列表见指标手册，其中沪深股票的行情报价、资金流向的指标需在csq语句中分开使用
options	可选参数	字符串	附加参数，可填附加字段，见**附注5**

参数名	简称	定义	描述
fncallback	回调函数	c_DataCallback	报价订阅回调
userparams	字符串	用户参数，回调时原样返回	

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
import time as _time
def csqCallback(quantdata):
    print "csqCallback,", str(quantdata)

data = c.csq("300059.SZ,002716.SZ,600834.SH,600616.SH",
"TIME,OPEN,HIGH,LOW,NOW,AMOUNT", "Pushtype=0", csqCallback)
if data.ErrorCode != 0:
    print "request csq Error, ", data.ErrorMsg
else:
    print u"csq输出结果======分隔线======"
    _time.sleep(2)
    text = raw_input("press any key to cancel csq \r\n")
    # 取消订阅
    data = c.csqcancel(data.SerialID)
```

范例(Python3.x)

```
import time as _time
def csqCallback(quantdata):
    print ("csqCallback,", str(quantdata))

data = c.csq('300059.SZ',
'Time,now,Roundlot','Pushtype=0,alltick=1',csqCallback)
if(data.ErrorCode != 0):
    print("request csq Error, ", data.ErrorMsg)
else:
    print("csq输出结果======分隔线======")
    _time.sleep(2)
    text = input("press any key to cancel csq \r\n")
    #取消订阅
    data = c.csqcancel(data.SerialID)
```

附注5 报价订阅函数可选参数列表：

中文名称	英文名称	取值范围	说明
推送方式	Pushtype	取值范围 0,2 缺省值： 0	取值0，增量推送 取值2，增量推送（数据补齐）
待处理队列堆积警告数量	WarnSize	默认值 500	当待处理队列达到设置数量时会有一条日志输出，若队列数一直大于设置数量，则每5秒输出一次日志。本参数在初次调用csq时生效。
是否包含全部tick	AllTick	默认不使用	推荐使用，当该参数有效时，返回的数据包含单只股票单次推送的所有数据。该参数无效时，返回的数据仅包含单只股票单次推送的最后一个tick的数据。

注：关于推送方式的说明

- 增量推送指的是当订阅代码的指标数据有变动时，变动指标推送对应数值，无变动时，推送none；
- 增量推送（数据补齐）指的是当订阅代码的指标数据有变动时，变动指标推送对应数值，无变动的指标客户端补齐为上一条的数据；
- AllTick为1时，quantdata.Data的Value数据结构为订阅数据的list组成的list；AllTick不为1时，quantdata.Data的Value数据结构为订阅数据组成的list。

快照函数

```
csqsnapshot(codes,indicators,options="")
```

提供各个证券品种的报价数据快照（需授权）

输入

参数	简称	定义	说明
codes	证券代码	字符串或者序列	东财代码，支持多代码输入，以半角逗号分隔
indicators	指标简称	字符串或者序列	指标名称，支持多指标输入，以半角逗号分隔，详细指标列表见指标手册或官网命令生成，其中沪深股票的行情报价、资金流向的指标需分开使用，与csq共用一套指标
options	可选参数	字符串	附加参数，可填附加字段，见**附注6**

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
data = c.csqsnapshot("000001.SZ",
    "PRECLOSE,OPEN,HIGH,LOW,NOW,AMOUNT", "IsPandsa=0")
if not isinstance(data, c.EmQuantData):
    print(data)
else:
    if data.ErrorCode != 0:
        print "request csqsnapshot Error, ", data.ErrorMsg
    else:
        print u"csqsnapshot输出结果======分割线======"
        for key, value in data.Data.items():
            print key, ">>> ",
```

```
            for v in value:
                print v,
            print ""
```

范例(Python3.x)

```
data = c.csqsnapshot("000001.SZ", "PRECLOSE,OPEN,HIGH,LOW,NOW,AMOUNT",
"Ispandas=0")

if not isinstance(data, c.EmQuantData):
    print(data)
else:
    if(data.ErrorCode != 0):
        print("request csqsnapshot Error, ",data.ErrorMsg)
    else:
        print("csqsnapshot输出结果======分割线======")
        for key,value in data.Data.items():
            print(key, ">>> ", end="")
            for v in value:
                print(v, " ", end="")
            print()
```

附注6 快照函数可选参数列表:

中文名称	英文名称	取值范围	说明
是否输出pandas格式	Ispandas	0--1，缺省值：0	非pandas格式--0; pandas格式--1; 需要安装pandas包

（注：快照函数请求频次不能超过每3秒/次，行情报价和资金流向限频互不影响）

日内跳价

```
cst(codes,indicators,startdatetime,enddatetime,options =
None,fncallback=None,userparams=None)
```

提供沪深股票的当日跳价（需授权）

输入

参数名	简称	定义	描述

参数名	简称	定义	描述
codes	证券代码	字符串或者序列	东财代码，支持多代码输入，以半角逗号分隔，不支持跨品种证券输入
indicators	指标简称	字符串或者序列	指标名称，支持多指标输入，以半角逗号分隔，详细指标列表见指标手册
startdatetime	开始时间	字符串	支持格式 YYYYMMDDHHMMSS或HHMMSS
enddatetime	结束时间	字符串	支持格式 YYYYMMDDHHMMSS或HHMMSS
options	可选参数	字符串	预留
fncallback	回调函数	c_DataCallback	日内跳价回调
userparams	字符串	用户参数，回调时原样返回	

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
import time as _time
def cstCallBack(quantdata):
    for i in range(0, len(quantdata.Codes)):
        length = len(quantdata.Dates)
        for it in quantdata.Data.keys():
            print it
            for k in range(0, length):
                for j in range(0, len(quantdata.Indicators)):
                    print quantdata.Data[it][j * length + k], " ",
                print ""
```

```
data = c.cst('300059.SZ', 'TIME,NOW', '100000', '101000', "", cstCallBack)
if data.ErrorCode != 0:
    print "request cst Error, ", data.ErrorMsg
else:
    print u"cst输出结果======分割线======"
    _time.sleep(2)
    raw_input("press any key to quit cst \r\n")
```

范例(Python3.x)

```
import time as _time
def cstCallBack(quantdata):
    for i in range(0, len(quantdata.Codes)):
        length = len(quantdata.Dates)
        for it in quantdata.Data.keys():
            print(it)
            for k in range(0, length):
                for j in range(0, len(quantdata.Indicators)):
                    print(quantdata.Data[it][j * length + k], " ",end = "")
                print()
data = c.cst('600000.SH', 'TIME,NOW', '093000', '094000','',cstCallBack)
if(data.ErrorCode != 0):
    print("request cst Error, ", data.ErrorMsg)
else:
    print("cst输出结果======分割线======")
    _time.sleep(2)
    input("press any key to quit cst \r\n")
```

专题报表

```
ctr(ctrName,indicators="",options="")
```

提供专题报表数据

输入

参数	简称	定义	描述
ctrName	报表名称	字符串	东财报表名称，详细枚举见指标手册

参数	简称	定义	描述
indicators	报表字段简称	字符串或者序列	报表字段简称，支持多字段输入，以半角逗号分隔，传空或匹配不到时展示报表全部字段，字段枚举详见指标手册
options	报表参数	字符串	报表参数明细，详见指标手册

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
data = c.ctr("INDEXCOMPOSITION", "","IndexCode=000001.SH,EndDate=2017-01-13")

if data.ErrorCode != 0:
    print "request ctr Error, ", data.ErrorMsg
else:
    for key,value in data.Data.items():
        for v in value:
            print v,
        print ""
```

范例(Python3.x)

```
data = c.ctr("INDEXCOMPOSITION", "", "IndexCode=000300.SH,EndDate=2017-01-13")
if(data.ErrorCode != 0):
    print("request ctr Error, ", data.ErrorMsg)
else:
    print("ctr输出结果======分割线======")
    for key,value in data.Data.items():
        for v in value:
            print(v, " ", end="")
        print()
```

宏观数据

```
edb(edbids, options)
```

获取宏观指标数据

输入

参数名	简称	定义	描述
edbids	宏观指标id	字符串	宏观指标id，支持多代码输入，最多不超过100个，以半角逗号分隔，宏观指标列表见量化接口官网-命令生成-宏观数据
options	可选参数	字符串	附加参数，可填""，可填附加字段，见**附注7**

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
data = c.edb("EMM00087117", "IsPublishDate=1,RowIndex=1,Ispandas=1")
if not isinstance(data, c.EmQuantData):
    print(data)
else:
    if (data.ErrorCode != 0):
        print "request edb Error, ", data.ErrorMsg
    else:
        print "edbid           date          ",
        for ind in data.Indicators:
            print ind, "    ",
```

```
        print ""
        for code in data.Codes:
            for j in range(0, len(data.Dates)):
                print code, "    ", data.Dates[j], "    ",
                for i in range(0, len(data.Indicators)):
                    print data.Data[code][i][j], "    ",
                print ""
```

范例(Python3.x)

```
data = c.edb("EMM00087117","IsPublishDate=1,RowIndex=1,Ispandas=1")
if not isinstance(data, c.EmQuantData):
    print(data)
else:
    if(data.ErrorCode != 0):
        print("request edb Error, ", data.ErrorMsg)
    else:
        print("edbid           date          ",end="")
        for ind in data.Indicators:
            print(ind, end="    ")
        print("")
        for code in data.Codes:
            for j in range(0, len(data.Dates)):
                print(code, "    ", data.Dates[j], end="    ")
                for i in range(0, len(data.Indicators)):
                    print(data.Data[code][i][j], end="    ")
                print("")
```

附注7 宏观数据函数可选参数列表：

中文名称	英文名称	取值范围	说明
是否输出pandas格式	Ispandas	取值范围：0，1 缺省值：0	非pandas格式--0； pandas格式--1； 需要安装pandas包
pandas索引	RowIndex	取值范围：1，2 缺省值：0	宏观指标ID--1； 日期--2；
起始日期	StartDate	支持格式： YYYYMMDD，YYYY/MM/DD， YYYY/M/D，YYYY-MM-DD，YYYY-M-D	若StartDate不传，从第一条数据开始返回； 若EndDate不传，返回至最新一条数据； 若都不传则输出全部数据.
截止日期	EndDate	同上	同上

中文名称	英文名称	取值范围	说明
最新一条数据	IsLatest	取值范围：0，1 缺省值：0	取值0，选定日期范围内数据 取值1，最新一条数据
请求指标真实发布日期	IsPublishDate	取值范围：0，1 缺省值：0	取值0，不请求publishdate 取值1，请求publishdate 备注：只有部分EDB指标有发布日期

资讯函数

```
cfn(codes,content,mode,options="")
```

提供多个证券品种的公告、新闻等历史资讯和多个板块的历史资讯查询

输入

参数	简称	定义	描述
codes	证券代码或板块代码	字符串、列表、元组	东财代码，支持多代码，以半角逗号分开。证券代码和板块代码不能混用。板块代码需先调用资讯板块查询函数获取。
content	请求内容类型	字符串	companynews-公司资讯 industrynews-行业资讯 report-公告 regularreport-定期公告 tradeinfo-重大事项（交易信息） **content为前面几个的时候codes必须为证券代码** sectornews-板块资讯 **content为sectornews时codes必须为板块代码** content支持前面五个混合，以半角逗号分隔。sectornews和其他不能混合请求

mode	请求模式	枚举int	eCfnMode_StartToEnd = 1 # starttime和endtime中间的所有资讯
			eCfnMode_EndCount = 2 # 提取endtime的近count条数据

options	可选参数	字符串	附加参数，可填""，可填附加字段，详见下表

options可选参数

中文名称	英文名称	说明

中文名称	英文名称	说明
开始时间	starttime	模式一必传，模式二无意义。YYYYMMDDHHMMSS或者YYYYMMDD
结束时间	endtime	YYYYMMDDHHMMSS或者YYYYMMDD。为空则为当前时间
资讯条数	count	模式二必传，模式一无意义，在模式二返回以endtime为基准的近count条资讯

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

EmQuantData.Data中指标字段说明

输出字段	字段中文
datetime	展示时间（公告只展示日期，盘后公告展示日期是下一个交易日）
eitime	生产时间（公告只提供2017/01/01之后的数据并且只含日期）
code	证券代码或板块代码
content	请求类型
title	资讯标题
infoCode	资讯编码
medianname	来源
url	链接

范例(Python2.x)

```
data = c.cfn("300059.SZ,600519.SH,300024.SZ", "companynews,industrynews",
eCfnMode_EndCount,
             "starttime=20190501010000,endtime=20190725,count=10")
print u"cfn输出结果======分隔线======"
if (not isinstance(data, c.EmQuantData)):
    print data
else:
    if (data.ErrorCode != 0):
        print "request cfn Error, ", data.ErrorMsg
    else:
        for code in data.Data:
            total = len(data.Data[code])
            for k in range(0, len(data.Data[code])):
                print data.Data[code][k]
```

范例(Python3.x)

```
data =
c.cfn("300059.SZ,600519.SH,300024.SZ","companynews,industrynews",eCfnMode_EndC
ount,"starttime=20190501010000,endtime=20190725,count=10")
print("cfn输出结果======分隔线======")
if (not isinstance(data, c.EmQuantData)):
    print (data)
else:
    if (data.ErrorCode != 0):
        print("request cfn Error, ", data.ErrorMsg)
    else:
        for code in data.Data:
            total = len(data.Data[code])
            for k in range(0, len(data.Data[code])):
                print(data.Data[code][k])
```

资讯订阅

```
cnq(codes, content, options="", fncallback)
```

订阅多个证券品种的公告、新闻等资讯和多个板块的资讯

输入

参数	简称	定义	描述

参数	简称	定义	描述
codes	证券代码或板块代码	字符串、列表、元组	东财代码，支持多代码，以半角逗号分开。证券代码和板块代码不能混用。板块代码需先调用资讯板块查询函数获取。
content	请求内容类型	字符串	companynews-公司资讯 industrynews-行业资讯 report-公告 regularreport-定期公告 tradeinfo-重大事项（交易信息） **content为前面几个的时候codes必须为证券代码** sectornews-板块资讯 **content为sectornews时codes必须为板块代码** content支持前面五个混合，以半角逗号分隔。sectornews和其他不能混合请求
options	可选参数	字符串	""，预留，暂无可选参数
fncallback	回调函数		资讯订阅回调，可以使用该函数对返回的数据进行处理

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
def cnqcallback(quantdata):
    print(str(quantdata.Data))
data = c.cnq("S888005002API", "sectornews", "", cnqCallback)
if data.ErrorCode != 0:
    print "request cnq Error, ", data.ErrorMsg
else:
    print u"cnq输出结果======分隔线======"
    _time.sleep(60)
    text = raw_input("press any key to cancel cnq \r\n")
    #取消订阅
    data = c.cnqcancel(data.SerialID)
```

范例(Python3.x)

```
def cnqcallback(quantdata):
    print(str(quantdata.Data))
data = c.cnq('300059.SZ,600030.SH','companynews,industrynews','',cnqcallback)
if data.ErrorCode != 0:
    print "request cnq Error, ", data.ErrorMsg
else:
    print u"cnq输出结果======分隔线======"
    _time.sleep(60)
    text = raw_input("press any key to cancel cnq \r\n")
    #取消订阅
    data = c.cnqcancel(data.SerialID)
```

功能函数

条件选股

```
cps(cpsCodes,cpsIndicators,cpsConditions,cpsOptions)
```

条件选股函数

输入

参数	简称	定义	描述

参数	简称	定义	描述
cpsCodes	板块代码或证券代码	字符串、列表、元组	控制选股范围，CPS函数只能选取沪深的板块和证券代码，取值格式有两种： 1. 板块代码，以B_开头，如 "B_001004"，常见板块代码见**附注8**; 2. 东财代码，多个代码间用半角逗号隔开， 如"000001.SZ,000002.SZ,600000.SH"
cpsIndicators	条件参数	字符串	定义条件表达式使用的参数，多个参数之间用英文分号隔开,内部各参数用半角逗号隔开，具体指标和英文简称见接口官网命令生成-功能函数-条件选股，如： s1,open,2016/12/31,1;s2,close,2017/02/25,1;s3,NAME;s4, LISTDATE
cpsConditions	条件表达式	字符串	条件表达式，各表达式用 and 连接，表达式支持的操作符：ANY，CONTAINALL，ISNULL，ISNOTNULL，比较运算符, 算术运算符，逻辑运算符(必须小写)如and、or、not 等，具体操作符释义详见**附注9**; 条件参数引用格式： [参数名1]，例如： [s1] >10 and [s2] > [s1] and not CONTAINANY ([s3],重工,银行)；若选择的条件是日期，需加d( )，例如：[s4]>d(2017/7/21)，若选日期区间，需用多项日期表达式，用and连接，例如：[s4]>d(2013-09-30) and [s4]<d(2014-07-10)
cpsOptions	附加参数	字符串	其他附加条件，如排序、取前N条选股结果等，具体使用规则见**附注10**

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = list() #数据结果

范例(Python2.x)

```
data = c.cps("B_001004","s1,OPEN,2017/2/27,1; s2,NAME","
[s1]>0","orderby=rd([s1]),top=max([s1],100)")
if data.ErrorCode != 0:
    print "request cps Error, ", data.ErrorMsg
else:
    for it in data.Data:
        print it
```

范例(Python3.x)

```
data = c.cps("B_001004","s1,OPEN,2017/2/27,1; s2,NAME", "[s1]>0",
"orderby=rd([s1]),top=max([s1],100)")
print("cps输出结果======分割线======")
for it in data.Data:
    print(it)
```

附注8 常见板块代码：

所属领域	板块名称	板块代码
沪深股票	全部A股	001004
沪深股票	上证A股	001005
沪深股票	深证A股	001006
沪深股票	深证B股	001013
沪深股票	上证B股	001012
沪深股票	全部B股	001011
沪深股票	创业板	001010
沪深股票	中小板	001009
沪深股票	深证主板	001008
沪深股票	深证主板A股	001007
沪深股票	风险警示股票	001023
沪深股票	风险警示股票（深交所）	001025
沪深股票	风险警示股票（上交所）	001024
沪深股票	已发行待上市股票	001020
沪深股票	正在发行的股票	001019
沪深股票	*ST	001018
沪深股票	ST	001017
沪深股票	全部A股(非金融石油石化)	001044
沪深股票	可转债标的	001046
沪深股票	融资融券标的	001045
沪深股票	深股通	001041
沪深股票	沪深股通	001047

所属领域	板块名称	板块代码
沪深股票	深证主板B股	001033
沪深股票	中小板(含ST,ST*)	001032
沪深股票	深证主板A股(含ST,ST*)	001031
沪深股票	沪股通	001038
沪深股票	中证500成份	009006062
沪深股票	中证1000成份	009007552
沪深股票	上证50指数成份	009007063
沪深股票	上证180指数成份	009007060
沪深股票	创业板综成份	009007145
沪深股票	创业板指成份	009007144
沪深股票	中小板综成份	009007125
沪深股票	中小板指成份	009007124
沪深股票	上证综合指数成份	009007104
沪深股票	沪深300成份	009006195
沪深股票	深证综合指数成份	009007251
沪深股票	MSCI中国（概念类）	007230
沪深股票	预盈预增	007054
沪深股票	预亏预减	007053

附注9 操作符列表：

分类	操作符	描述	详细	举例
算术运算符	+ - * /	加 减 乘 除		
比较运算符	> = < >= <= <>	大于 等于 小于 大于或等于 小于或等于 不等于		
逻辑运算符	and or not	与 或 非		

分类	操作符	描述	详细	举例
	ANY CONTAINANY	包含任意一个		ANY(s[1] , 中国 , 美国) 表示当变量s1中包含“中国”或“美国”则成立
	CONTAINALL	包含所有值		CONTAINALL (s[1] , 中国 , 美国) 表示当变量s1中包含“中国”且包含“美国”则成立
	MAX	取最大的N个值	用在Top表达式中，对选股结果取TOP	top=max([s1],100)
	MIN	取最小的N个值	用在在Top表达式中，对选股结果取最小的N行	top=min([s2],100)
	ISNULL ISNOTNULL	等于空值 不等于空值	条件选股取空值或不取空值，多项输入用and连接	isnull([s1])and isnotnull([s2])

附注10 排序表达式和Top表达式使用规则：

字段说明	取值格式	取值示例
排序表达式：对返回的结果进行排序	格式：orderby=[rd|ra]，（rd为降序，ra为升序） 支持变量引用，引用格式为方括号+变量名 多个排序字段间，以=> 符号分隔 支持的操作符：rd,ra,算术运算符，“=>”分隔符	orderby=rd([s1]*2) => ra([s2])
Top表达式：对返回结果按指定排序提取前N行	格式：top=max(排序字段表达式,行数)或top=min(排序字段表达式,行数) 支持变量引用，引用格式为方括号+变量名 支持的操作符and,or,算术运算符	top = max([s1],100) and min([s2],100)
板块成分日期：选择的板块成分的日期	格式：sectordate=板块历史成分的日期，若sectordate不传则默认取最新的一天。	sectordate=2018-07-18

宏观指标查询

```
edbquery(edbids,indicators="",options="")
```

获取宏观指标id详情信息

输入

参数名	简称	定义	描述
edbids	宏观指标id	字符串	宏观指标id，支持多代码输入，最多不超过100个，以半角逗号分隔
indicators	详情字段简称	字符串	详情字段简称，支持多字段输入，以半角逗号分隔，传空或匹配不到时则输出全部字段，详细字段列表见**附注11**
options	可选参数	字符串	附加参数，可填NULL，可填附加字段

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
data = c.edbquery("EMM00058124,EMM00087117,EMG00147350")
if(data.ErrorCode != 0):
    print "request edbquery Error, ", data.ErrorMsg
else:
    print "edbid           ",
    for ind in data.Indicators:
        print ind, "   ",
    print ""
    for code in data.Codes:
        for j in range(0,len(data.Dates)):
            print code, "    ", "   ",
            for i in range(0,len(data.Indicators)):
                print data.Data[code][j],   "   ",
            print ""
```

范例(Python3.x)

```
data = c.edbquery("EMM00058124,EMM00087117,EMG00147350")
if(data.ErrorCode != 0):
    print("request edbquery Error, ", data.ErrorMsg)
```

```
else:
    print("edbid           ",end="")
    for ind in data.Indicators:
        print(ind, end="   ")
    print("")
    for code in data.Codes:
        for j in range(0,   len(data.Dates)):
            print(code, "    ", end="   ")
            for i in range(0, len(data.Indicators)):
                print(data.Data[code][j], end="   ")
            print("")
```

附注11 宏观指标信息查询函数支持字段列表：

字段简称	中文简称	备注
ID	指标ID	
Name	指标名称	
Unit	单位	
Source	来源	
Region	国家/地区	
Frequency	日期频率	1 日 2 周 3 旬 4 半月 5 月 6 季 7 半年 8 年 9 不定期
Startdate	起始日期	
Enddate	截止日期	
Updatetime	更新时间	

资讯板块查询

```
cfnquery()
```

获取资讯函数和资讯订阅函数支持的板块信息

返回

类型	描述

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

EmQuantData.Data字段说明

输出字段	字段中文
seccode	板块代码
secname	板块名称
psecname	母板块中文名称

范例(Python2.x)

```
data = c.cfnquery("")
print u"cfnquery输出结果======分隔线======"
if(not isinstance(data, c.EmQuantData)):
    print data
else:
    if(data.ErrorCode != 0):
        print "request cfnquery Error, ", data.ErrorMsg
    else:
        for code in data.Codes:
            for i in range(0, len(data.Indicators)):
                    print data.Data[code][i]
```

范例(Python3.x)

```
data = c.cfnquery("")
print("cfnquery输出结果======分隔线======")
if (not isinstance(data, c.EmQuantData)):
    print(data)
else:
    if (data.ErrorCode != 0):
        print("request cfnquery Error, ", data.ErrorMsg)
    else:
        for code in data.Codes:
            for i in range(0, len(data.Indicators)):
                print(data.Data[code][i])
```

取消报价订阅

```
csqcancel(serialID);
```

取消特定或所有的报价订阅

输入

参数名	简称	定义	描述
serialID	流水号	数字	传入特定流水号，取消对应的报价订阅； 传入0，取消所有的报价订阅

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例

```
data = c.csqcancel(0)
```

取消资讯订阅

```
cnqcancel(serialID);
```

取消特定或所有的资讯订阅

输入

参数名	简称	定义	描述
serialID	流水号	数字	传入特定流水号，取消对应的资讯订阅； 传入0，取消所有的资讯订阅

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例

```
data = c.cnqcancel(0)
```

板块函数

```
sector(pukeycode,enddate,ptions=None,*arga,**argb)
```

获取Choice金融终端指定系统板块证券代码成分列表，**目前只支持沪深股票、上交所期权的历史成分查询，其他板块只能获取最新成分**

输入

参数	简称	定义	描述
pukeycode	板块代码	字符串	通过 Choice 量化接口网站命令生成（http://quantapi.eastmoney.com/Cmd/Sector?from=web）获取
enddate	截止日期	字符串或者datetime	支持格式： YYYYMMDD，YYYY/MM/DD，YYYY/M/D，YYYY-MM-DD，YYYY-M-D
options	可选参数	字符串	附加参数，可填NULL，可填附加字段

参数	简称	定义	描述
arga	可选参数	可变参数	预留
argb	可选参数	可变参数	预留

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = list() #数据结果

范例(Python2.x)

```
data = c.sector("001004", "2016-04-26")
if data.ErrorCode != 0:
    print "request sector Error,   ", data.ErrorMsg
else:
    for code in data.Data:
        print code
```

范例(Python3.x)

```
data = c.sector("001004", "2016-04-26")
if data.ErrorCode != 0:
    print("request sector Error,   ", data.ErrorMsg)
else:
    for code in data.Data:
        print(code)
```

交易日历

```
tradedates(startdate,enddate,options=None,*arga,**argb)
```

获取指定交易市场，指定时间区间的日期序列

输入

参数名	简称	定义		描述
startdate	起始日期	字符串或者datetime	支持格式： YYYYMMDD， YYYY/MM/DD YYYY/M/D ， YYYY-MM-DD ， YYYY-M-D	
enddate	截止日期	字符串或者datetime	支持格式： YYYYMMDD， YYYY/MM/DD ，YYYY/M/D，YYYY-MM-DD，YYYY-M-D	不建议使用未来交易日
options	可选参数	字符串	In	附加参数，可填NULL，可填附加字段，见**附注12**
arga	可选参数	可变参数		预留
argb	可选参数	可变参数		预留

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = list() #数据结果

范例(Python2.x)

```
data = c.tradedates("2016-07-01", "2016-07-12")
if data.ErrorCode != 0:
    print "request tradedates Error, ", data.ErrorMsg
else:
    print u"tradedate输出结果======分隔线======"
    for item in data.Data:
        print item
```

范例(Python3.x)

```
data = c.tradedates("2016-07-01", "2016-07-12")
if(data.ErrorCode != 0):
    print("request tradedates Error, ", data.ErrorMsg)
else:
    print("tradedate输出结果======分隔线======")
    for item in data.Data:
        print(item)
```

附注12 交易日函数可选参数列表:

中文名称	英文名称	取值范围	说明
日期周期	Period	1--4，缺省值：1	日期周期：日，周，月，年 分别对应：1，2，3，4
按日期排序	Order	1--2，缺省值：1	升序--1； 降序--2

中文名称	英文名称	取值范围	说明
市场类型	Market	见说明，缺省值：“CNSESH”	CNSESH 上海证券交易所 CNSESZ 深圳证券交易所 HKSE00 香港证券交易所 USSE00 美国证券交易所 USSEND 美国纳斯达克市场 USSENY 纽约证券交易所 CNFEBC 渤海商品交易所 CNFEDC 大连商品交易所 CNFESF 上海期货交易所 CNFEZC 郑州商品交易所 INE000 上海国际能源交易中心 CNGCSH 上海黄金交易所 HKME00 香港商品交易所 CNSH00 沪股通交易日 CNSHHK 沪港股通交易日 CNSZ00 深股通交易日 CNSZHK 深港股通交易日 NYMEX0 纽约商业期货交易所 USFENY 纽约商品交易所 CME000 芝加哥商业交易所 LDMETL 伦敦金属交易所 LDEXCH 伦敦证券交易所 SGSE00 新加坡交易所

交易日偏移

```
getdate(tradedate,offday=0,options=None,*arga,**argb)
```

获取指定市场交易日历推算第N天交易日

输入

参数名	简称	定义	描述
tradedate	交易日期	字符串或者datetime	支持格式： YYYYMMDD，YYYY/MM/DD，YYYY/M/D，YYYY-MM-DD，YYYY-M-D
offday	偏移天数	数字	N=0时，返回交易日当天； N>0时，交易日往后取最近第N个交易日的日期，若交易日期为最新交易日并N>0,则返回最新交易日； N<0,时，交易日往前取最近第N个交易日的日期。

参数名	简称	定义	描述
options	可选参数	字符串	附加参数，可填附加字段，见附**注13**
arga	可选参数	可变参数	预留
argb	可选参数	可变参数	预留

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = list() #数据结果

范例(Python2.x)

```
data = c.getdate("20160426",-3,"Market=CNSESH")
if data.ErrorCode != 0:
    print "request getdate Error,   ", data.ErrorMsg
else:
    for tradedate in data.Data:
        print tradedate
```

范例(Python3.x)

```
data = c.getdate("20160426",-3,"Market=CNSESH")
if data.ErrorCode != 0:
    print("request getdate Error,   ", data.ErrorMsg)
else:
    print(data.Data)
```

附注13 偏移N天函数可选参数列表：

中文名称	英文名称	取值范围	说明

中文名称	英文名称	取值范围	说明
市场类型	Market	见说明，缺省值：“CNSESH”	CNSESH 上海证券交易所 CNSESZ 深圳证券交易所 HKSE00 香港证券交易所 USSE00 美国证券交易所 USSEND 美国纳斯达克市场 USSENY 纽约证券交易所 CNFEBC 渤海商品交易所 CNFEDC 大连商品交易所 CNFESF 上海期货交易所 CNFEZC 郑州商品交易所 INE000 上海国际能源交易中心 CNGCSH 上海黄金交易所 HKME00 香港商品交易所 CNSH00 沪股通交易日 CNSHHK 沪港股通交易日 CNSZ00 深股通交易日 CNSZHK 深港股通交易日 NYMEX0 纽约商业期货交易所 USFENY 纽约商品交易所 CME000 芝加哥商业交易所 LDMETL 伦敦金属交易所 LDEXCH 伦敦证券交易所 SGSE00 新加坡交易所

区间交易日数

```
tradedatesnum(startdate,enddate,options)
```

获取指定交易市场，指定时间区间的交易日个数

输入

参数名	简称	定义	描述
startdate	起始日期	字符串或者datetime	支持格式： YYYYMMDD， YYYY/MM/DD YYYY/M/D ， YYYY-MM-DD ， YYYY-M-D
enddate	截止日期	字符串或者datetime	支持格式： YYYYMMDD， YYYY/MM/DD YYYY/M/D ， YYYY-MM-DD ， YYYY-M-D
options	可选参数	字符串	附加参数，可填附加字段，见**附注13**

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例(Python2.x)

```
data = c.tradedatesnum("2018-01-01", "2018-09-15")
if data.ErrorCode != 0:
    print "request tradedatesnum Error, ", data.ErrorMsg
else:
    print "tradedatesnum======分割线======"
    print data.Data
```

范例(Python3.x)

```
data = c.tradedatesnum("2018-01-01", "2018-09-15")
if data.ErrorCode != 0:
    print("request tradedatesnum Error, ", data.ErrorMsg)
else:
    print("tradedatesnum======分割线======")
    print(data.Data)
```

设置代理函数

```
setproxy(type,proxyip,port,verify,usr,pwd)
```

设置代理

输入

参数	简称	定义	描述
type	代理类型	数字（整数）	ePT_NONE：不使用代理 ePT_HTTP：HTTP代理 ePT_HTTPS：HTTPS代理 ePT_SOCK4：SOCK4代理 ePT_SOCK5:SOCK5代理

参数	简称	定义	描述
proxyip	代理服务器地址	字符串	代理服务器IP
port	代理服务器端口	数字（整数）	代理服务器Port
verify	是否验证账户名和密码	布尔值	True：验证代理服务器账户名和密码 False：不验证
usr	账户名	字符串	代理服务器账户名
pwd	密码	字符串	代理服务器密码

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例(Python3.x)

```
data = c.setproxy(ePT_HTTP,"120.0.0.0",8080,True,"choice","password")
if data.ErrorCode != 0:
    print("setproxy   failed, ", data.ErrorMsg)
```

人工激活函数

```
manualactivate(uname, password, options, logcallback)
```

人工激活登录。本函数可独立使用，无需调用start。本函数适用于无界面运行环境（如远程linux）或无法运行LoginActivator程序的情况，激活成功后将通过邮件获得的激活文件"userInfo"放到"ServerList.json.e"同级目录，再调用start登录

输入

参数	简称	定义	描述
uname	用户名	字符串	输入用户名
password	密码	字符串	输入密码
options	附加参数	字符串	邮箱地址必传，格式"email=xx@xx.com"

参数	简称	定义	描述
logcallback	附加参数	字符串	日志回调函数，也可传None

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例(Python3.x)

```
data = c.manualactivate("usr", "pwd","email=xxx@163.com")
if data.ErrorCode != 0:
    print("manualactivate   failed, ", data.ErrorMsg)
```

组合函数

新建组合

```
pcreate(combinCode,combinName,initialFound,remark,options="")
```

新建组合

输入

参数名	参数简称	定义	描述
combinCode	组合代码	字符串	组合代码，英文和数字，**最大10位，单个账户最多支持30个**
combinName	组合名称	字符串	组合名称
initialFound	初始资金	数字	初始资金，上限99999999999
remark	组合说明	字符串	组合说明
options	附加参数	字符串	附加参数，可填空字串，可填附加字段，见**附注14**

返回

类型	描述

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = list() #数据结果

范例(Python2.x)

```
data = c.pcreate("quant001.PF", "组合牛股", 100000000, "这是一个牛股的组合")
if data.ErrorCode != 0 :
    print "request pcreate Error, ", data.ErrorMsg
else:
    print "create succeed"
```

范例(Python3.x)

```
data = c.pcreate("quant001.PF", "组合牛股", 100000000, "这是一个牛股的组合")
if(data.ErrorCode != 0):
    print("request pcreate Error, ", data.ErrorMsg)
else:
    print("create succeed")
```

附注14 新建组合函数可选参数列表:

中文名称	英文名称	取值范围	说明
组合类型	combintype	1-4 默认值 1	1 成长型 2 指数型 3 平衡型 4 稳健型
创建公司	createcompany	字符串	默认留空

中文名称	英文名称	取值范围	说明
业绩基准	criteria	1-16 默认3	1 上证指数 2 深证成指 3 沪深300指数 4 上证A股指数 5 上证180指数 6 上证50指数 7 深证100指数 8 中小板指 9 中小板综 10 创业板指 11 深证综指 12 三板做市指数 13 基金指数 14 中证500指数 15 中证100指数 16 中证1000指数

组合资金调配

```
pctransfer(combinCode,transferdirect,date,opCash,remark,options="")
```

提供组合出入金调配

参数

参数名	参数简称	定义	描述
combinCode	组合代码	字符串	组合代码，英文和数字，最大10位，单个账户最多支持30个
transferdirect	资金调配方向	字符串	in：增加资金 out：减少资金
date	调配日期	字符串或者datetime	交易日期格式：YYYYMMDD,YYYY/MM/DD,YYYY/M/D,YYYY-MM-DD,YYYY-M-D
opCash	调配资金量	数字	增加或减少的资金量，上限100000000000
remark	说明	字符串	说明
options	附加参数	字符串	附加参数，可填空字串，可填附加字段，见**附注15**

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data =dict() #数据结果

范例(Python2.x)

```
data = c.pctransfer("quant001.PF", "in", "2019-08-29", 100000, "追加资金")
if data.ErrorCode != 0 :
    print "request pctransfer Error, ", data.ErrorMsg
else:
    print "pctransfer succeed"
```

范例(Python3.x)

```
data = c.pctransfer("quant001.PF", "in", "2019-08-29", 100000, "追加资金")
if(data.ErrorCode != 0):
    print("request pctransfer Error, ", data.ErrorMsg)
else:
    print("pctransfer succeed")
```

附注15 组合资金调配函数可选参数列表:

中文名称	英文名称	取值范围	说明
调配方式	transfertype	1-2	1、单个组合增加或减少 2、组合间调配
组合2 ID	combincode2	字符串	若transfertype为2 则必传
调配时间	time	24小时	格式：HHMMSS，仅在当天有效

组合查询

```
pquery(options="")
```

提供组合账户信息相关数据

输入

参数名	简称	定义	描述
options	附加参数	字符串	附加参数，详见**附注16**

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果，详见**附注17**

范例(Python2.x)

```
data = c.pquery()
if (data.ErrorCode != 0):
    print("request pquery Error, ", data.ErrorMsg)
else:
    print "[key]:",
    for index in range(0, len(data.Indicators)):
        print "\t", data.Indicators[index],
    print ""
    for k, v in data.Data.items():
        print k, ": ",
        for vv in v:
            print "\t", vv,
        print ""
```

范例(Python3.x)

```
data = c.pquery()
if(data.ErrorCode != 0):
    print("request pquery Error, ", data.ErrorMsg)
else:
    print("[key]:",end="")
    for index in range(0, len(data.Indicators)):
        print("\t",   data.Indicators[index],end="")
    print("")
    for k,v in data.Data.items():
        print(k,": ",   end="")
        for vv in v:
            print("\t", vv,   end="")
        print("")
```

附注16 组合查询函数附加参数明细

中文名称	英文名称	取值范围	说明
组合类型	combinetype	0-我管理的组合 1-我关注的组合	默认值：0

附注17 返回数据具体参数明细

参数名	简称	备注
COMBINCODE	组合代码	
GROUPNAME	组合名称	
STARTCAST	初始资金	
RESTFOUND	剩余资金	
CREATEDATE	创建日期	
MODIFYDATE	最近调整日期	
GROUPTYPE	组合类型	1：成长型 2：指数型 3：平衡型 4：稳健型
MONEYTYPE	基准货币	1：人民币 2：美元 3：港币

参数名	简称	备注
criteria	组合业绩基准	1 上证指数 2 深证成指 3 沪深300指数 4 上证A股指数 5 上证180指数 6 上证50指数 7 深证100指数 8 中小板指 9 中小板综 10 创业板指 11 深证综指 12 三板做市指数 13 基金指数 14 中证500指数 15 中证100指数 16 中证1000指数
CREATECOMPANY	创建公司	
REMARK	组合说明	
LEVEL	组合等级	1：普通组合 2：POP组合
FOLLOWEDID	关注组合ID	仅在请求关注组合数据时，返回对应ID

批量下单

```
porder(combincode,orderdict,remark,options="")
```

组合批量下单

输入

参数名	简称	定义	描述
combincode	组合代码	字符串	组合代码
orderdict	下单参数	字典	具体下单信息，见**附注18**
remark	备注信息	字符串	备注信息
options	附加参数	字符串	附加参数，见**附注19**

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例(Python2.x)

```
orderdict = {'code':['300059.SZ','600000.SH'],
             'volume':[1000,200],
             'price':[13.11,12.12],
             'date':['2017-08-14','2017-08-24'],
             'time':['14:22:18','14:22:52'],
             'optype':[eOT_buy,eOT_buy],
             'cost':[0,3],
             'rate':[0,2],
             'destvolume':[0,0],
             'weight':[0.1,0.1]}

data = c.porder("quant001", orderdict, "this is a test")
if(data.ErrorCode != 0):
    print "porder Error, ", data.ErrorMsg
else:
    print "order succeed"
```

范例(Python3.x)

```
orderdict = {'code': ['300059.SZ', '600000.SH'],
             'volume': [1000, 200],
             'price': [13.11, 12.12],
             'date': ['2017-08-14', '2017-08-24'],
             'time': ['14:22:18', '14:22:52'],
             'optype': [eOT_buy, eOT_buy],
             'cost': [0, 3],
             'rate': [0, 2],
             'destvolume': [0, 0],
             'weight': [0.1, 0.1]}
data = c.porder("quant001", orderdict, "this is a test")
if(data.ErrorCode != 0):
    print("porder Error, ", data.ErrorMsg)
else:
    print("order succeed")
```

附注18：orderdict参数定义列表

序号	参数名	简称	定义	是否必填参数	说明

序号	参数名	简称	定义	是否必填参数	说明
1	code	代码	字符串	是	东财代码，格式为'300059.SZ'
2	volume / destvolume / weight	数量或金额/ 目标数量/ 目标权重	数字	是	股票、场内基金数量（正负表示买入、卖出）， 场外基金申购定投金额、赎回份额 option参数： OrderMode=0时表示交易数量或金额 OrderMode=1时表示持仓目标数量 OrderMode=2时表示持仓目标权重
3	price	价格	数字	是	交易价格
4	date	日期	字符串	是	交易日期格式为"YYYYMMDD,YYYY/MM/DD,YYYY-MM-DD"
5	time	时间	字符串	否	交易时间，格式为“hhmmss，hh:mm:ss”（只影响当日交易，历史交易后台默认150000录入计算）
6	optype	操作	数字	否	1买入，2卖出，3申购，4赎回 股票交易对应1、2， 场外基金交易对应3、4
7	cost	费用	数字	否	场外基金申购、赎回费用/费率，二选一填写，另一个位置填写0。 费用为0，则读取费率； 费率为0，则读取费用； 两者都为0，表示0费用、费率； 两者都不为0，默认读取第一个费用。 单位：费用 元；费率 % **适用于对场外基金、股票品种**
8	rate	费率	数字	否	**仅适用于对场外基金品种**

附注19：组合交易附加参数options列表

中文名称	英文名称	取值范围	说明
补入现金方式	autoAddCash	0--2，缺省值：0	0：不补充 1：先扣除可用现金，不足再补充 2：全部外部补充本次批量买入操作所需现金
下单模式	OrderMode	0--2，缺省值：0	0：按数量交易 volume传入交易数量 1：调仓至目标数量，destvolume传目标数量 2：调仓至目标权重，weight传目标权重，总权重相加不能超过1 备注：1 和 2 不支持逆回购

组合报表查询

```
preport(combinCode,indicator,options="")
```

查询组合报表信息

输入

参数名	参数简称	定义	描述
combinCode	组合代码	字符串	组合代码，支持单次查询单个组合的单个报表
indicator	报表名称	字符串	报表名称，hold(持仓查询)， record(交易记录查询)，Contri（业绩贡献-已清仓股票），stagePerf（周期回报-阶段回报），profAna（盈亏分析-区间分析），RiskAna（风险分析），VarAna（VAR分析），GDaily（组合日报），TDaily（交易日报），ctransferrecord(组合资金调配报表)
options	附加参数	字符串	附加参数，可填空字符串，可填附加字段，见指标手册

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息
	self.Codes = list() #东财代码
	self.Indicators = list() #指标简称
	self.Dates = list() #日期序列
	self.RequestID = 0 #请求ID
	self.SerialID = 0 #返回的订阅号
	self.Data = dict() #数据结果

范例(Python2.x)

```
data =
c.preport("quant001.PF","record","startdate=2017/07/12,enddate=2018/01/15")
if(data.ErrorCode != 0):
    print "request preport Error, ", data.ErrorMsg
else:
    for ind in data.Indicators:
        print ind, "   ",
    print ""
    for k in data.Data:
        for it in data.Data[k]:
            print it, "   ",
        print ""
```

范例(Python3.x)

```
data = c.preport("quant001",
"record","startdate=2017/07/12,enddate=2018/01/15")
if(data.ErrorCode != 0):
    print("request preport Error, ", data.ErrorMsg)
else:
    for ind in data.Indicators:
        print(ind, end="   ")
    print("")
    for k in data.Data:
        for it in data.Data[k]:
            print(it, end="   ")
        print("")
```

删除组合

```
pdelete(combinCode,options="")
```

删除组合

输入

参数名	参数简称	定义	描述
combinCode	组合代码	字符串	组合代码
options	附加参数	字符串	附加参数，可不填

返回

类型	描述

EmQuantData结构体	class EmQuantData:
	def **init**(self):
	self.ErrorCode = 0 #错误码
	self.ErrorMsg = 'success' #错误信息

范例(Python2.x)

```
data = c.pdelete("quant001.PF")
if(data.ErrorCode != 0):
    print "request pdelete Error, ", data.ErrorMsg
else:
    print "delete succeed"
```

范例(Python3.x)

```
data = c.pdelete("quant001.PF")
if(data.ErrorCode != 0):
    print("request pdelete Error, ", data.ErrorMsg)
else:
    print("delete succeed")
```

错误类型

错误类型

错误标识	错误描述	错误代码
EQERR_SECUSS	成功	0
EQERR_NO_LOGIN	用户未登陆	10001001
EQERR_USERNAMEORPASSWORD_ERR	用户名或密码错误	10001002
EQERR_NO_ACCESS	用户无API权限	10001003
EQERR_ACCESS_EXPIRE	用户API权限过期	10001004
EQERR_GETUSERINFO_FAIL	获取用户信息失败	10001005
EQERR_DLLVESION_EXPIRE	DLL版本号过期	10001006

错误标识	错误描述	错误代码
EQERR_NO_LV2_ACCESS	用户无API_LV2权限	10001007
EQERR_LV2_ACCESS_EXPIRE	用户API_LV2权限过期	10001008
EQERR_LOGIN_COUNT_LIMIT	账号登陆数达到上限	10001009
EQERR_LOGIN_FAIL	用户登陆失败	10001010
EQERR_LOGIN_DISCONNECT	用户登陆掉线	10001011
EQERR_ACCESS_INSUFFICIENCE	用户权限不足	10001012
EQERR_IS_LOGIN	用户正在登录	10001013
EQERR_NEED_ACTIVATE	需要登录激活	10001014
EQERR_LOGIN_SERVICE_ERR	登录服务异常	10001015
EQERR_IS_MANUAL_ACTIVATE	正在人工激活	10001016
EQERR_NOTNEED_MANUAL_ACTIVATE	无需人工激活	10001017
EQERR_MANUAL_ACTIVATE_FAIL	人工激活失败	10001018
EQERR_DIFFRENT_DEVICE	激活设备与登录设备不一致	10001019
EQERR_USERINFO_EXPIRED	userInfo已失效需重新激活	10001020
EQERR_QUOTE_LOGIN_FAIL	行情服务登录验证失败	10001021
EQERR_QUOTE_FLOW_FAIL	行情服务流量验证失败	10001022
EQERR_INFOQUERY_LOGIN_FAIL	资讯查询服务登录验证失败	10001023
EQERR_INFOSUB_LOGIN_FAIL	资讯订阅服务登录验证失败	10001024
EQERR_INFO_FLOW_FAIL	资讯服务流量验证失败	10001025
EQERR_GET_TRADE_FAIL	获取交易日失败	10000001
EQERR_INIT_OBTAIN_CLASS_FAIL	初始化主类失败	10000002
EQERR_NEW_MEM_FAIL	申请内存失败	10000003
EQERR_PARSE_DATA_ERR	解析数据错误	10000004
EQERR_UNGZIP_DATA_FAIL	gzip解压失败	10000005
EQERR_UNKNOWN_ERR	未知错误	10000006
EQERR_FUNCTION_INTERNAL_ERR	函数内部错误	10000007
EQERR_OUTOF_BOUNDS	数组越界	10000008
EQERR_NO_DATA	无数据	10000009

错误标识	错误描述	错误代码
EQERR_SYSTEM_ERROR	系统级别错误	10000010
EQERR_SERVERLIST_ERROR	服务器列表错误	10000011
EQERR_OPERATION_FAILURE	操作失败	10000012
EQERR_SERVICE_ERROR	服务错误	10000013
EQERR_GETSERVERLIST_FAIL	获取服务器列表失败	10000014
EQERR_SERVICE_TIMEOUT	服务超时	10000015
EQERR_FREQUENCY_OVER	请求频次过高	10000016
EQERR_OVERSEAS_IP_RESTRICTED	海外IP受限	10000017
EQERR_POP_GROUP_NOT_SUPPORT	POP组合不支持此操作	10000018
EQERR_SOCKET_ERR	网络错误	10002001
EQERR_CONNECT_FAIL	网络连接失败	10002002
EQERR_CONNECT_TIMEOUT	网络连接超时	10002003
EQERR_RECVCONNECTION_CLOSED	网络接收时连接断开	10002004
EQERR_SENDSOCK_FAIL	网络发送失败	10002005
EQERR_SENDSOCK_TIMEOUT	网络发送超时	10002006
EQERR_RECVSOCK_FAIL	网络接收错误	10002007
EQERR_RECVSOCK_TIMEOUT	网络接收超时	10002008
EQERR_QUOTE_RECONNECT_FAIL	行情服务器连续重连失败	10002009
EQERR_HTTP_FAIL	http访问失败	10002010
EQERR_WAIT_NET_RES_TIMEOUT	等待网络响应超时	10002011
EQERR_QUOTE_RECONNECT	行情服务器重连	10002012
EQERR_INFO_RECONNECT	资讯服务器重连	10002013
EQERR_INFO_RECONNECT_FAIL	资讯服务器连续重连失败	10002014
EQERR_INPARAM_EMPTY	传入参数为空	10003001
EQERR_OUTPARAM_EMPTY	传出参数为空	10003002
EQERR_PARAM_ERR	参数错误	10003003
EQERR_START_DATE_ERR	起始日期格式不正确	10003004
EQERR_END_DATE_ERR	截止日期格式不正确	10003005

错误标识	错误描述	错误代码
EQERR_START_BIGTHAN_END	起始日期大于截至日期	10003006
EQERR_DATE_ERR	日期格式不正确	10003007
EQERR_CODE_INVALIED	无效的证券代码	10003008
EQERR_CODE_REPEAT	证券代码重复	10003009
EQERR_INDICATOR_INVALIED	无效的指标	10003010
EQERR_USERNAME_EMPTY	用户名为空	10003011
EQERR_PASSWORD_EMPTY	密码为空	10003012
EQERR_TO_UPPER_LIMIT	订阅数或股票总数达到上限	10003013
EQERR_MIXED_INDICATOR	不支持的混合指标	10003014
EQERR_INDICATOR_TO_UPPER_LIMIT	单次订阅指标达到上限	10003015
EQERR_BEYOND_DATE_SUPPORT	超出日期支持范围	10003016
EQERR_BASE_LESS_THAN_END	复权基期小于截止日期	10003017
EQERR_MIXED_CODES_MARKET	不支持的混合证券品种	10003018
EQERR_NO_SUPPORT_CODES_MARKET	不支持的证券代码品种	10003019
EQERR_ORDER_TO_UPPER_LIMIT	交易条数超过上限	10003020
EQERR_NO_SUPPORT_ORDERINFO	不支持的交易信息	10003021
EQERR_INDICATOR_REPEAT	指标重复	10003022
EQERR_INFOBKCODE_INVALIED	资讯板块代码错误	10003023
EQERR_INFOSIZE_TOOLARGE	资讯数据量过大	10003024
EQERR_INFO_SEARCH_NODATA	资讯查询不到数据	10003025
EQERR_INFOBKCODE_REPEAT	资讯板块代码重复	10003026