

# DataAPI 文档

发布 0.7.0

东兴投资

2016 年 03 月 24 日

<b>1</b>	<b>更新历史</b>	<b>1</b>
<b>2</b>	<b>介绍</b>	<b>4</b>
2.1	目的	4
2.2	为 DataAPI 添砖加瓦	4
<b>3</b>	<b>安装</b>	<b>5</b>
3.1	依赖	5
3.2	下载	5
3.3	安装	5
3.4	运行	6
<b>4</b>	<b>api</b>	<b>7</b>
4.1	基本使用	7
4.2	函数列表	8
<b>5</b>	<b>证券基本信息</b>	<b>21</b>
5.1	基本检索	21
5.2	获取期货基本信息	21
5.3	获取股票基本信息	24
5.4	函数列表	25
<b>6</b>	<b>证券行情</b>	<b>27</b>
6.1	如何使用	27
6.2	函数列表	28
<b>7</b>	<b>股票</b>	<b>30</b>
7.1	获取基本行情	30

7.2	分批获取数据 . . . . .	32
7.3	使用缓存加速获取数据 . . . . .	33
7.4	将 instrumentID 转为 column index . . . . .	34
7.5	函数列表 . . . . .	36
<b>8</b>	<b>指数</b>	<b>39</b>
8.1	行情信息 . . . . .	39
8.2	成分股信息 . . . . .	39
8.3	函数列表 . . . . .	40
<b>9</b>	<b>指数期货</b>	<b>43</b>
9.1	函数列表 . . . . .	43
<b>10</b>	<b>商品期货</b>	<b>46</b>
10.1	获取大宗商品库存数据 . . . . .	46
10.2	函数列表 . . . . .	47
<b>11</b>	<b>量化基金研究库</b>	<b>48</b>
11.1	获取在池基金信息 . . . . .	48
11.2	获取特定基金的信息 . . . . .	48
11.3	获取基金净值表现 . . . . .	49
11.4	获取基金风险收益指标 . . . . .	50
11.5	获取基金的风格分析 . . . . .	51
11.6	函数列表 . . . . .	52
<b>12</b>	<b>主题与情感信息</b>	<b>55</b>
12.1	获取主题列表 . . . . .	55
12.2	获取主题相关个股 . . . . .	55
12.3	获取主题热度 . . . . .	56
12.4	获取热门股票热门股 . . . . .	56
12.5	函数列表 . . . . .	57
<b>13</b>	<b>市场快照</b>	<b>60</b>
13.1	行业资金流向快照 . . . . .	60
13.2	函数列表 . . . . .	60
<b>14</b>	<b>面向开发者</b>	<b>62</b>
14.1	为自定义的数据 api 开启缓存 . . . . .	62
14.2	中文编码处理 . . . . .	63
14.3	函数列表 . . . . .	64

---

## 更新历史

---

### 2016 年 3 月 21 日

- [新功能]: 增加对冲基金风格分析的 api。
- [更新]: 将对冲基金数据库从本地 sqlite 改为 mysql。

### 2016 年 2 月 16 日

- [新功能]: 增加获取获取商品库存的数据 api。
- [新功能]: 增加获取指数期货连续合约行情的 api。
- [更新]: 使得 sanpshot 中函数的报错信息更为友好。

### 2016 年 2 月 4 日

- [修正]: 取消期货 1 分钟线的测试, 现阶段期货 1 分钟数据数据库中已经停止更新。

### 2016 年 1 月 21 日

- [新功能]: 增加获取交易所交易基金价格的 api。

### 2016 年 1 月 14 日

- [新功能]: 增加获取行业资金流向的 api, 来源: [通联数据 \(https://open.datayes.com\)](https://open.datayes.com)。
- [更新]: 在所有的行情 api 中增加 securityID 列来唯一标识某个证券。

### 2016 年 1 月 11 日

- [更新]: 现在在 index 的行情数据中, 同时支持输入指数的原始代码或者 TDB 转换后的代码 (使用 TDB 转换代码时, api 会打印警告信息提示)。
- [更新]: 股票 EOD 价格数据切换至 WindDB.ASHAREEODPRICES。
- [更新]: 指数 EOD 价格数据切换至 WindDB.AINDEXEODPRICES。

### 2016 年 1 月 7 日

- [新功能]: 增加关于主题热门股的 api, 来源: 通联数据 (<https://open.datayes.com>)。

#### 2016 年 1 月 6 日

- [新功能]: 增加指数成分股的 api。

#### 2015 年 12 月 31 日

- [新功能]: 增加获取股票信息的 api。
- [更新]: 股票数据源切换至 patch 表中。

#### 2015 年 12 月 17 日

- [新功能]: 增加获取期货合约信息的 api。
- [修正]: 在某些情况下, 获取复权数据的时候, closePrice 会变成非常接近于 0 的错误。

#### 2015 年 12 月 15 日

- [修正]: 取消行情数据中填充 nan 数据的操作, 需要确定更合乎逻辑的操作流程。

#### 2015 年 12 月 14 日

- [新功能]: 在行情 api 中增加参数 instrumentIDasCol, 将该参数设为 True 可以将 instrumentID 转为 column index。
- [修正]: 将缓存机制使用的 hdf 文件储存格式从 table 调整为 fixed。

#### 2015 年 12 月 11 日

- [新功能]: 增加获取混合品种行情数据的 api。

#### 2015 年 12 月 10 日

- [更新]: 将对 tables 的依赖改为可选并更新相应文档。

#### 2015 年 11 月 23 日

- [新功能]: 完善复权的获取方式, 用户可以在 baseDate 指定 start (后复权) 或者 end (前复权)。
- [修正]: 修改 chunksize 不为 0 时的行为, 保证输出的数据仍然为按照时间排序的。

#### 2015 年 11 月 20 日

- [修正]: 有的时候无法正确识别缓存数据的 bug。

#### 2015 年 11 月 19 日

- [新功能]: 现在的行情 API 可以启用本地缓存提高读取速度。默认情况下, 缓存机制不开启。用户可以在参数中使用 forceUpdate=False 来开启缓存机制。
- [修正]: 取消在 windows 环境下 multiprocessing 模块的使用。该模块会在某些特定情况下引发 RuntimeError ;

- [修正]: python3 下整除问题的修复。

**2015 年 11 月 17 日**

- [修正]: 修改 `_GetBarData` 实现, 保证使用 `chunksize` 参数时的行为与不使用时候的一致性。

**2015 年 11 月 16 日**

- [新功能]: 股票行情数据接口支持获取复权数据;

---

## 介绍

---

### 2.1 目的

提供基于 Python 的数据库访问，主要目的：

1. 将用户从繁琐的拼写 sql 语句的工作中解放出来，更专注于数据具体的内容；
2. 增加抽象层，避免用户对数据库的直接访问。方便未来 DBA 对于数据库的更新、维护以及解耦。

### 2.2 为 DataAPI 添砖加瓦

数据 API 是一个需求驱动型的项目，单个的 api 往往即对应一个典型的数据库查询情景，例如这条查询语句：

```
select instrumentID, tradingDate, closePrice, openPrice from EQY_EOD
  where instrumentID='600000' and tradingDate > '2015-01-01' and tradingDate < '2015-10-01'
```

等价于如下的 api 函数调用：

```
api.GetEquityBarEOD('600000', '2015-01-01', '2015-10-01', ['closePrice', 'openPrice'])
```

所以，最简单的为 DataAPI 做出贡献的办法是：

---

**小技巧：**把你最最最常用 SQL 语句发给我们!!（不要忘记附上简单的查询含义，这样方便我们进行 api 命名）

---

## 3.1 依赖

推荐使用 [Anaconda](https://www.continuum.io/downloads) (<https://www.continuum.io/downloads>) 发行版，降低使用 pip 安装 numpy 以及 pandas 可能遇到的问题：

- decorator
- enum34 (仅 python2)
- numpy
- pandas
- pymssql
- sqlalchemy

可选依赖（用于缓存机制）：

- tables

## 3.2 下载

DataAPI 的源代码可以在以下 svn 地址获取：

```
svn co https://10.63.6.72/svn/IT/midOffice/MarketData/DataCenter/DataAPI/trunk
```

## 3.3 安装

在签出 svn 中最新代码之后，运行：



```
cd DataAPI  
python setup.py install
```

## 3.4 运行

运行 python:

```
In [1]: import DataAPI  
  
In [2]: DataAPI.__version__  
Out[2]: '0.7.0'
```

4.1 基本使用

该模块负责导入希望用户可以直接访问的数据获取函数。一般来说，用户只需要在使用的时候专注于这个模块的函数，而无须关注其他的实现细节。本质上说，该模块的作用是将分散于各处的数据获取函数以统一的方式展示给用户使用。

典型的一个使用 DataAPI 情景如下：

```
In [1]: from DataAPI import api

In [2]: prices = api.GetEquityBarMin5('000001', '2015-10-09', '2015-10-20', ['closePrice'])

In [3]: prices.tail()
Out[3]:
```

	closePrice	instrumentID	tradingDate	tradingTime	\
timeStamp					
2015-10-20 14:35:00	11.26	000001	2015-10-20	14:35:00.0000000	
2015-10-20 14:40:00	11.27	000001	2015-10-20	14:40:00.0000000	
2015-10-20 14:45:00	11.29	000001	2015-10-20	14:45:00.0000000	
2015-10-20 14:50:00	11.28	000001	2015-10-20	14:50:00.0000000	
2015-10-20 14:55:00	11.32	000001	2015-10-20	14:55:00.0000000	

  

	securityID
timeStamp	
2015-10-20 14:35:00	000001.xshe
2015-10-20 14:40:00	000001.xshe
2015-10-20 14:45:00	000001.xshe
2015-10-20 14:50:00	000001.xshe

2015-10-20 14:55:00 000001.xshe

关于每个证券类型具体的 API，请参看：[股票](#)、[指数](#)、[期货](#) 等。

## 4.2 函数列表

Created on 2015-11-12

@author: cheng.li, weijun.shen

**class** `api.BAR_TYPE`

An enumeration.

**class** `api.ASSET_TYPE`

An enumeration.

`api.GetEquityBarMin1(instrumentIDList, startDate, endDate, field='*', chunksize=None, base-`

`Date=None, forceUpdate=True, instrumentIDasCol=False)`

获取股票 1 分钟线数据

### 参数

- **instrumentIDList** – 证券名或者列表，例如：‘600000’ 或者 [‘600000’, ‘000001’]
- **startDate** – 开始日期，格式：‘YYYY-MM-DD’
- **endDate** – 结束日期，格式：‘YYYY-MM-DD’
- **field** – 需要获取的字段类型，例如：[‘closePrice’]，不填的话，默认获取所有字段；可用的 field 包括：[productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **baseDate** – 获取复权数据时的默认基准日，可填：日期，‘start’，‘end’，以及 None，默认为 None：不复权 ‘YYYY-MM-DD’：以固定日期为复权基准日。若该日不是交易日，向前调整至最近的交易日；‘start’：以 startDate 起始日期为基准日。若该日不是交易日，向前调整至最近的交易日；‘end’：以 endDate 结束日期为基准日。若该日不是交易日，向前调整至最近的交易日；None：不复权
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 `pandas.DataFrame`

`api.GetEquityBarMin5(instrumentIDList, startDate, endDate, field='*', chunksize=None, baseDate=None, forceUpdate=True, instrumentIDasCol=False)`  
 获取股票 5 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '600000' 或者 ['600000', '000001']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **baseDate** – 获取复权数据时的默认基准日, 可填: 日期, 'start', 'end', 以及 None, 默认为 None: 不复权 'YYYY-MM-DD': 以固定日期为复权基准日。若该日不是交易日, 向前调整至最近的交易日; 'start': 以 startDate 起始日期为基准日。若该日不是交易日, 向前调整至最近的交易日; 'end': 以 endDate 结束日期为基准日。若该日不是交易日, 向前调整至最近的交易日; None: 不复权
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

`api.GetEquityBarEOD(instrumentIDList, startDate, endDate, field='*', chunksize=None, baseDate=None, forceUpdate=True, instrumentIDasCol=False)`  
 获取股票日线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '600000' 或者 ['600000', '000001']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **baseDate** – 获取复权数据时的默认基准日, 可填: 日期, 'start', 'end', 以及 None, 默认为 None: 不复权。'YYYY-MM-DD': 以固定日期为复权基准日。若

该日不是交易日，向前调整至最近的交易日；‘start’：以 startDate 起始日期为基准日。若该日不是交易日，向前调整至最近的交易日；‘end’：以 endDate 结束日期为基准日。若该日不是交易日，向前调整至最近的交易日；None：不复权

- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetIndexBarMin1(instrumentIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, instrumentIDasCol=False)
```

获取指数 1 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表，例如：‘000300’ 或者 [‘000300’]
- **startDate** – 开始日期，格式：‘YYYY-MM-DD’
- **endDate** – 结束日期，格式：‘YYYY-MM-DD’
- **field** – 需要获取的字段类型，例如：[‘closePrice’]，不填的话，默认获取所有字段；可用的 field 包括：[productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetIndexBarMin5(instrumentIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, instrumentIDasCol=False)
```

获取指数 5 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表，例如：‘000300’ 或者 [‘000300’]
- **startDate** – 开始日期，格式：‘YYYY-MM-DD’
- **endDate** – 结束日期，格式：‘YYYY-MM-DD’
- **field** – 需要获取的字段类型，例如：[‘closePrice’]，不填的话，默认获取所有字段；可用的 field 包括：[productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据

- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

`api.GetIndexBarEOD(instrumentIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, instrumentIDasCol=False)`  
获取指数日线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '000300' 或者 ['000300']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

`api.GetIndexConstitutionInfo(instrumentIDList, refDate=None, field='*', forceUpdate=True)`  
获取指定日的指数成分股信息

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '000300' 或者 ['000300']
- **refDate** – 基准日, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['inDate'], 不填的话, 默认获取所有字段; 可用的 field 包括: [instrumentID, windCode, conInstrumentID, conWindCode, inDate, outDate]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

`api.GetFutureBarMin1(instrumentIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, instrumentIDasCol=False)`  
获取期货 1 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: 'IF1506' 或者 ['IF1502', 'IH1506']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover, matchItems]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetFutureBarMin5(instrumentIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, instrumentIDasCol=False)
```

获取期货 5 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: 'IF1506' 或者 ['IF1502', 'IH1506']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover, matchItems]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetFutureBarMin5Continuing(productIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, productIDasCol=False)
```

获取期货连续合约 5 分钟线数据

#### 参数

- **productIDList** – 证券名或者列表, 例如: 'IF' 或者 ['IF', 'IH']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'

- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover, matchItems]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **productIDasCol** – 联合使用 field 以及 productIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetFutureBarEOD(instrumentIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, instrumentIDasCol=False)
```

获取期货日线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: 'IF1506' 或者 ['IF1502', 'IH1506']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetFutureBarEODContinuing(productIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, productIDasCol=False)
```

获取期货连续合约日线数据

#### 参数

- **productIDList** – 证券名或者列表, 例如: 'IF' 或者 ['IF', 'IH']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'



- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **productIDasCol** – 联合使用 field 以及 productIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetCommodityStocksInfo(productIDList, startDate, endDate, wareHouse='*', field='*', forceUpdate=True)
```

获取商品期货的每日仓单库存信息

#### 参数

- **productIDList** – 证券名或者列表, 例如: 'IF' 或者 ['IF', 'IH']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **wareHouse** – 所在仓库, 例如: [u' 江苏: 中储无锡']。默认为 '\*', 获取全部仓库的信息
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, tradingDate, wareHouse, stocks]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

```
api.GetHedgeFundInfo(instruments=None, instrumentName=None, field='*', firstInvestType=None, forceUpdate=True)
```

获取对冲基金基础信息

#### 参数

- **instruments** – 基金代码列表, 例如: 'XT1515141.XT'。默认为 None, 获取所有的基金信息
- **field** – 需要获取的字段类型, 例如: ['firstInvestType'], 不填的话, 默认获取所有字段; 可用的 field 包括: [instrumentID, fullName, firstInvestType, investScope, maturityDate, advisory]
- **firstInvestType** – 需要获取的基金所属的投资类型列表, 例如: [u' 期货型']
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 `pandas.DataFrame`

```
api.GetHedgeFundPool(field='*', refDate=None, forceUpdate=True)
```

获取指定日期下, 基金备选池的信息

#### 参数

- **field** – 需要获取的字段类型, 例如: `['logRetAcc']`, 不填的话, 默认获取所有字段; 可用的 field 包括: `[tradingDate, instrumentID, navUnit, navAcc, logRetUnit, logRetAcc]`
- **refDate** – 指定日期, 将查询范围限制于当日在基金备选池中的基金, 格式为: `YYYY-MM-DD`; 不填的话, 默认查询最新的备选池信息
- **forceUpdate** – 当为 `True` 时强制刷新数据, 不使用缓存。默认为 `True`

返回 `pandas.DataFrame`

```
api.GetHedgeFundBarWeek(instruments=None, startDate='2003-01-01', endDate='2015-12-31',
                        field='*', forceUpdate=True, instrumentIDasCol=False)
获取对冲基金历史表现信息 (周)
```

#### 参数

- **instruments** – 基金代码列表, 例如: `'XT1515141.XT'`。默认为 `None`, 获取所有基金历史表现
- **field** – 需要获取的字段类型, 例如: `['logRetAcc']`, 不填的话, 默认获取所有字段; 可用的 field 包括: `[tradingDate, instrumentID, navUnit, navAcc, logRetUnit, logRetAcc]`
- **forceUpdate** – 当为 `True` 时强制刷新数据, 不使用缓存。默认为 `True`
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 `pandas.DataFrame`

```
api.GetHedgeFundBarMonth(instruments=None, startDate='2003-01-01', endDate='2015-12-31',
                        field='*', forceUpdate=True, instrumentIDasCol=False)
获取对冲基金历史表现信息 (月)
```

#### 参数

- **instruments** – 基金代码列表, 例如: `'XT1515141.XT'`。默认为 `None`, 获取所有基金历史表现
- **field** – 需要获取的字段类型, 例如: `['logRetAcc']`, 不填的话, 默认获取所有字段; 可用的 field 包括: `[tradingDate, instrumentID, navUnit, navAcc, logRetUnit, logRetAcc]`

- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
api.GetHedgeFundStylePerf(styleName=None, startDate='2012-01-01', endDate='2015-12-31', instrumentIDasCol=False, forceUpdate=True)
```

获取特定风格的基金类型的历史表现

#### 参数

- **styleName** – 基金风格类型, 可选的风格有: ['u' 债券型', 'u' 多空仓型', 'u' 宏观策略', 'u' 市场中性', 'u' 管理期货', 'u' 股票型', 'u' 货币型']
- **startDate** – 历史表现起始日期
- **endDate** – 历史表现结束日期
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

```
api.GetHedgeFundPerfComparison(instruments=None, startDate='2015-01-01', endDate='2015-12-31', forceUpdate=True)
```

获取指定基金的历史表现风险收益指标以及打分结果。

#### 参数

- **instruments** – 基金 wind 代码列表, 可以为空。若为空, 则获取所有在池基金的结果。
- **startDate** – 历史表现起始日
- **endDate** – 历史表现结束日
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

```
api.GetHedgeFundStyleAnalysis(instruments=None, navTable=Empty DataFrame Columns: [] Index: [], startDate='2012-01-01', endDate='2015-01-01', forceUpdate=True)
```

获取根据基金历史表现得到的风格分析的结果

#### 参数

- **instruments** – 基金 wind 代码列表, 可以为空。若为空, 则使用用户输入的 navTable

- **navTable** –  $N * (M+1)$  的 `pandas.DataFrame`, 表示  $M$  个基金历史上  $N$  个时间点的净值
- **startDate** – 历史表现起始日, 在使用 `navTable` 时候会被忽略

:param **endDate**: 历史表现结束日, 在使用 `navTable` 时候会被忽略: param **forceUpdate**: 当为 `True` 时强制刷新数据, 不使用缓存。默认为 `True` :return: `pandas.DataFrame`

`api.GetGeneralBarData(instrumentIDList, startDate, endDate, bType, field='*', chunksize=None, baseDate=None, forceUpdate=True, instrumentIDasCol=False)`  
获取一般证券的行情数据

### 参数

- **instrumentIDList** – 证券列表, 例如: ['600000.xshg', 'ifl512', '000300.zicn']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **bType** – bar 的类型, 例如: `BAR_TYPE.MIN1`
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 `chunksize` 大小的数据
- **baseDate** – 获取股票复权数据时的默认基准日, 可填: 日期, 'start', 'end', 以及 `None`, 默认为 `None`: 不复权。'YYYY-MM-DD': 以固定日期为复权基准日。若该日不是交易日, 向前调整至最近的交易日; 'start': 以 `startDate` 起始日期为基准日。若该日不是交易日, 向前调整至最近的交易日; 'end': 以 `endDate` 结束日期为基准日。若该日不是交易日, 向前调整至最近的交易日; `None`: 不复权
- **forceUpdate** – 当为 `True` 时强制刷新数据, 不使用缓存。默认为 `True`
- **instrumentIDasCol** – 联合使用 `field` 以及 `instrumentIDs` 作为 `column` 的名字

返回 `pandas.DataFrame`

`api.GetFutureInstrumentInfo(instrumentDescList=None, field='*', refdate=None, forceUpdate=True)`  
获取期货基本信息数据

### 参数

- **instrumentDescList** – 证券名或者列表, 例如: 'ZN1502' 或者 ['ZN1502', 'A1502']; 不分大小写, 例如: 'a1502' 同 'A1502'; 支持模糊查询, 例如: 'if\*' 返

回所有以'IF'开头的证券，'15'返回'IF1501','ZN1502'等证券名称中包含'15'的证券及其检索信息；不填的话，默认获取所有证券；

- **field** – 需要获取的字段类型，例如：['market', 'cnName']，不填的话，默认获取所有字段；可用的 field 包括：[instrumentID, windCode, market, enName, cnName]
- **refdate** – 指定日期，将查询范围限制于当日依然处于交易状态的期货合约，例如：instrumentDescList = 'IF\*', refdate= '2015-11-11' 时，返回 IF1511, IF1512, IF1603, IF1606 四份期货的信息，非交易日时返回空值；不填的话，默认不限制具体某交易日；
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True

返回 pandas.DataFrame

```
api.GetEquityInstrumentInfo(instrumentIDList=None, boardName=None, field='*', refDate=None,
                             forceUpdate=True)
```

获取股票基本信息数据

#### 参数

- **instrumentIDList** – 证券名称或者列表，例如：'600000' 或者 ['600000', '000001']，默认为 None，查询所有证券
- **boardName** – 上市板块名称或者列表，例如：'主板' 或者 ['主板', '创业板']，默认为 None，查询所有板块
- **field** – 需要获取的字段类型，例如：['instrumentID', 'windCode']，不填的话，默认获取所有字段；
- **refDate** – 指定日期，将查询范围限制于当日依然处于上市状态的证券，格式为：YYYY-MM-DD
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True

返回 pandas.DataFrame

```
api.GetThemeInfo(themeName=None, themeID=None, field='*', forceUpdate=True)
```

获取相关的主题列表

#### 参数

- **themeName** – 关注的主题名，支持模糊查找，默认查找全部主题
- **themeID** – 关注的主题 id，为整数，与 themeName 二选一输入
- **field** – 需要获取的字段类型，例如：['newsNumPercent']，不填的话，默认获取所有字段；可用的 field 包括：[themeID, themeName, isActive, insertTime, updateTime]

- **forceUpdate** –

### 返回

`api.GetThemeHotness(themeName, startDate, endDate, field='*', forceUpdate=True)`

获取主题热度时间序列

### 参数

- **themeName** – 主题名称, 例如: u' 金融'
- **startDate** – 起始日, 格式: YYYY-MM-DD
- **endDate** – 结束日, 格式: YYYY-MM-DD
- **field** – 需要获取的字段类型, 例如: ['newsNumPercent'], 不填的话, 默认获取所有字段; 可用的 field 包括: [themeID, themeName, date, newsNum, newsNumPercent]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

`api.GetStocksByTheme(themeName, refDate=None, field='*', forceUpdate=True)`

获取指定主题相关的股票

### 参数

- **themeName** – 主题名称, 例如: u' 金融'
- **refDate** – 参考日, 格式: YYYY-MM-DD
- **field** – 需要获取的字段类型, 例如: ['instrumentID'], 不填的话, 默认获取所有字段; 可用的 field 包括: [themeID, themeName, instrumentID, cnName, exchangeName, score]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

`api.GetActiveThemesRelatedStocks(refDate=None, windows=None, topThemes=20, topStocks=20, field='*', forceUpdate=True)`

获取指定数量的热门主题高相关度的股票

### 参数

- **refDate** – 参考日, 格式: YYYY-MM-DD。默认为 None, 取当前日期
- **windows** – 参考周期, 可填写正数表示日; 或者字符串形式的时间长度, 例如: 1m。默认为 None, 只取 1 日
- **topThemes** – 选取排名靠前多少的主题, 默认值为 20

- **topStocks** – 在同主题下选取排名靠前多少的股票，默认值为 20
- **field** – 需要获取的字段类型，例如：['instrumentID']，不填的话，默认获取所有字段；可用的 field 包括：[themeID, themeName, instrumentID, cnName, date, themeHotness, score]
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True

返回 pandas.DataFrame

`api.GetIndustryNetCashSnapshot(industryID=None, industryName=None, forceUpdate=True)`

获取行业资金流向的快照

#### 参数

- **industryID** – 证监会行业分类 id，可以为列表，例如: ['J66', 'J68']
- **industryName** – 证监会分类行业名称，例如: '保险业'
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True

返回 pandas.DataFrame

证券基本信息

5.1 基本检索

目前，此模块包含以下功能：

- `GetFutureInstrumentInfo`：获取期货基本信息；
- `GetEquityInstrumentInfo`：获取股票基本信息

5.2 获取期货基本信息

5.2.1 基本使用

具体示例如下：

```
In [1]: api.GetFutureInstrumentInfo('IF1606',['market','windCode'])
Out[1]:
```

	market	windCode	instrumentID
0	CFFEX	IF1606.CFE	IF1606

上面的代码，获取了沪深 1606（代码 IF1606）的交易市场和万得证券代码。

默认情况下，若不输入任何 `field` 和 `refdate` 参数，函数将返回所查询期货证券的所有基本信息，并且不受限于查询时刻的交易状态：

```
In [2]: api.GetFutureInstrumentInfo('IF1606')
Out[2]:
```

	instrumentID	cnName	enName	market	windCode
0	IF1606	IF1606	CFFEX-IF1606	CFFEX	IF1606.CFE



若需要限定查询范围于某一具体日期仍处于可交易状态的期货合约，需要在 api 调用中指定 `refdate` 参数。完整的函数使用说明可以通过 python 的 `help` 函数查看：

```
help(api.GetFutureInstrumentInfo)
```

Help on function GetFutureInstrumentInfo in module DataAPI.InstrumentInfo:

```
GetFutureInstrumentInfo(instrumentDescList=None, field='*', refdate=None, forceUpdate=True)
```

获取期货基本信息数据

```
:param instrumentDescList: 证券名或者列表，例如：'ZN1502' 或者 ['ZN1502', 'A1502']; 不分大小写，
    例如：'a1502' 同'A1502'; 支持模糊查询，例如：'if*' 返回所有以'IF' 开头的证券，'*15*' 返回
    'IF1501','ZN1502' 等证券名称中包含'15' 的证券及其检索信息；不填的话，默认获取所有证券
:param field: 需要获取的字段类型，例如：['market', 'cnName'], 不填的话，默认获取所有字段
:param refdate: 指定日期，将查询范围限制于当日依然处于交易状态的期货合约，例如：instrumentDescList='IF*',
    refdate='2015-11-11' 时，返回 IF1511, IF1512, IF1603, IF1606 四份期货的信息，非交易日时
    返回空值；不填的话，默认不限制具体某交易日
:param forceUpdate: 当为 True 时强制刷新数据，不使用缓存。默认为 True
:return: pandas.DataFrame
```

## 5.2.2 模糊检索期货合约信息

函数支持模糊检索功能，具体使用方法为使用字符 `*` 替代需要模糊处理的部分。另外，需要查询的证券名称不区分大小写，例如：

```
In [3]: api.GetFutureInstrumentInfo('if*').head()
```

```
Out[3]:
```

	instrumentID	cnName	enName	market	windCode
0	IF1507	IF1507	CFFEX-IF1507	CFFEX	IF1507.CFE
1	IFM0805	IF0805(仿真)	IF0805(Simulation)	CFFEX	IFM0805.CFE
2	IFM1009	IF1009(仿真)	IF1009(Simulation)	CFFEX	IFM1009.CFE
3	IF1506	IF1506	CFFEX-IF1506	CFFEX	IF1506.CFE
4	IF1308	IF1308	CFFEX-IF1308	CFFEX	IF1308.CFE

以上函数用于查询以 **IF** 开头的合约（即所有沪深 300 指数期货合约）。

```
In [4]: api.GetFutureInstrumentInfo('*1512', field = ['windCode', 'market']).head()
```

```
Out[4]:
```

	windCode	market	instrumentID
0	FU1512.SHF	SHFE	FU1512

1	NI1512.SHF	SHFE	NI1512
2	RB1512.SHF	SHFE	RB1512
3	FB1512.DCE	DCE	FB1512
4	AG1512.SHF	SHFE	AG1512

以上函数用于查询以 **1512** 结尾的合约（即所有在 2015 年 12 月到期的合约）。

```
In [5]: api.GetFutureInstrumentInfo('*15*',field = ['windCode','market']).head()
```

Out[5]:

	windCode	market	instrumentID
0	CS1505.DCE	DCE	CS1505
1	IF1507.CFE	CFFEX	IF1507
2	BU1511.SHF	SHFE	BU1511
3	SM1508.CZC	CZCE	SM1508
4	WH1503.CZC	CZCE	WH1503

以上函数用于查询包含 **15** 字段的合约（即所有在 2015 年到期的合约）。

### 5.2.3 限定交易日检索合约信息

若以日期格式 YYYY-MM-DD 输入参数 `refdate`，函数将返回于该日处于可交易状态的相关合约信息（若指定日期为非交易日，没有合约处于可交易状态），例如：

```
In [6]: api.GetFutureInstrumentInfo('if*',refdate = '2015-12-01')
```

Out[6]:

	instrumentID	cnName	enName	market	windCode
0	IF1603	IF1603	CFFEX-IF1603	CFFEX	IF1603.CFE
1	IF1606	IF1606	CFFEX-IF1606	CFFEX	IF1606.CFE
2	IF1601	IF1601	CFFEX-IF1601	CFFEX	IF1601.CFE
3	IF1512	IF1512	CFFEX-IF1512	CFFEX	IF1512.CFE

以上函数用于查询在 2015 年 12 月 1 日可交易的沪深指数期货合约。

在函数设定中，参数证券名 `instrumentDescList` 可以缺省，此时将返回满足其它限定条件的全部证券信息：

```
In [7]: api.GetFutureInstrumentInfo(field = 'market',refdate = '2015-12-01').head()
```

Out[7]:

	market	instrumentID
0	CZCE	TC604
1	SHFE	HC1602
2	SHFE	FU1605

3	DCE	V1603
4	SHFE	AG1608

以上函数用于查询在 2015 年 12 月 1 日可交易的所有期货合约名称以及其交易场所。

## 5.3 获取股票基本信息

### 5.3.1 基本使用

具体示例如下：

```
In [8]: api.GetEquityInstrumentInfo(instrumentIDList='601198',
...: field=['cnName', 'listDate', 'listBoardName'])
...:
Out[8]:
cnName listDate listBoardName
0 东兴证券 20150226 主板
```

### 5.3.2 按上市板块查找

可以搜索，例如，所有 **创业板** 股票：

```
In [9]: api.GetEquityInstrumentInfo(boardName=u'主板',
...: field=['instrumentID', 'cnName', 'listBoardName']).tail()
...:
Out[9]:
instrumentID cnName listBoardName
1991 600335 国机汽车 主板
1992 600010 包钢股份 主板
1993 600527 江南高纤 主板
1994 000699 S*ST 佳纸（退市） 主板
1995 600129 太极集团 主板
```

### 5.3.3 按日期查找

可以按照日期查询当日处于上市状态的股票。例如查询 2015 年 1 月 1 日处于上市状态的股票：

```
In [10]: api.GetEquityInstrumentInfo(refDate=u'2015-01-01',
...: field=['instrumentID', 'cnName', 'listDate']) \
```

```
.....: .sort_values('listDate').tail()
.....:
```

Out[10]:

	instrumentID	cnName	listDate
2508	603889	新澳股份	20141231
2563	300412	迦南科技	20141231
2529	603017	中衡设计	20141231
2509	300411	金盾股份	20141231
2546	300410	正业科技	20141231

可以尝试换个时间:

```
In [11]: api.GetEquityInstrumentInfo(refDate=u'2015-02-28',
.....: field=['instrumentID', 'cnName', 'listDate']) \
.....: .sort_values('listDate').tail()
.....:
```

Out[11]:

	instrumentID	cnName	listDate
2536	002742	三圣特材	20150217
2565	603118	共进股份	20150225
2520	601198	东兴证券	20150226
2522	603828	柯利达	20150226
2557	603969	银龙股份	20150227

## 5.4 函数列表

Created on Tue Dec 15 09:50:06 2015

@author: cheng.li, weijun.shen, yucheng.lai

```
DataAPI.InstrumentInfo.GetEquityInstrumentInfo(instrumentIDList=None, boardName=None,
                                                field='*', refDate=None, forceUp-
                                                date=True)
```

获取股票基本信息数据

### 参数

- **instrumentIDList** – 证券名称或者列表, 例如: ‘600000’ 或者 [‘600000’, ‘000001’], 默认为 None, 查询所有证券
- **boardName** – 上市板块名称或者列表, 例如: ‘主板’ 或者 [‘主板’, ‘创业板’], 默认为 None, 查询所有板块

- **field** – 需要获取的字段类型, 例如: ['instrumentID', 'windCode'], 不填的话, 默认获取所有字段;
- **refDate** – 指定日期, 将查询范围限制于当日依然处于上市状态的证券, 格式为: YYYY-MM-DD
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

```
DataAPI.InstrumentInfo.GetFutureInstrumentInfo(instrumentDescList=None, field='', ref-  
date=None, forceUpdate=True)
```

获取期货基本信息数据

### 参数

- **instrumentDescList** – 证券名或者列表, 例如: 'ZN1502' 或者 ['ZN1502', 'A1502']; 不分大小写, 例如: 'a1502' 同 'A1502'; 支持模糊查询, 例如: 'if\*' 返回所有以 'IF' 开头的证券, '15' 返回 'IF1501', 'ZN1502' 等证券名称中包含 '15' 的证券及其检索信息; 不填的话, 默认获取所有证券;
- **field** – 需要获取的字段类型, 例如: ['market', 'cnName'], 不填的话, 默认获取所有字段; 可用的 field 包括: [instrumentID, windCode, market, enName, cnName]
- **refdate** – 指定日期, 将查询范围限制于当日依然处于交易状态的期货合约, 例如: instrumentDescList = 'IF\*', refdate = '2015-11-11' 时, 返回 IF1511, IF1512, IF1603, IF1606 四份期货的信息, 非交易日时返回空值; 不填的话, 默认不限制具体某交易日;
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

证券行情

6.1 如何使用

在 DataAPI 中我们设计了如下的函数获取多种证券的行情（证券这里可以是股票、期货或者指数等）：

- GetGeneralBarData

**警告：** 在使用以上 api 时，所有的证券代码必须带上交易所后缀。例如：‘600000’ 必须写为 ‘600000.xshg’，‘if1512’ 必须写为 ‘if1512.cffex’ 等。

使用方式如下：

```
In [1]: from DataAPI import api

In [2]: data = api.GetGeneralBarData(['600000.xshg', 'if1512.cffex', '000300.zicn'], \
...:                                '2015-01-01', \
...:                                '2015-10-01', \
...:                                api.BAR_TYPE.EOD)
...:
```

获取股票的信息：

```
In [3]: data[data['instrumentID'] == '600000'][['instrumentID', 'closePrice']].tail(5)
Out[3]:
```

	instrumentID	closePrice
timeStamp		
2015-09-24	600000	15.27
2015-09-25	600000	15.59
2015-09-28	600000	15.92
2015-09-29	600000	15.97
2015-09-30	600000	16.63

获取期货的信息:

```
In [4]: data[data['instrumentID'] == 'IF1512'][['instrumentID', 'closePrice']].tail(5)
```

Out[4]:

	instrumentID	closePrice
timeStamp		
2015-09-24	IF1512	3069.0
2015-09-25	IF1512	2994.0
2015-09-28	IF1512	3024.8
2015-09-29	IF1512	2941.0
2015-09-30	IF1512	2989.2

获取指数的信息:

```
In [5]: data[data['instrumentID'] == '000300'][['instrumentID', 'closePrice']].tail(5)
```

Out[5]:

	instrumentID	closePrice
timeStamp		
2015-09-24	000300	3284.9991
2015-09-25	000300	3231.9514
2015-09-28	000300	3242.7524
2015-09-29	000300	3178.8544
2015-09-30	000300	3202.9475

## 6.2 函数列表

Created on 2015-12-11

@author: cheng.li, weijun.shen

`DataAPI.General.GetGeneralBarData(instrumentIDList, startDate, endDate, bType, field='*', chunk-size=None, baseDate=None, forceUpdate=True, instrumentIDasCol=False)`

获取一般证券的行情数据

### 参数

- **instrumentIDList** – 证券列表, 例如: ['600000.xshg', 'if1512', '000300.zicn']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **bType** – bar 的类型, 例如: BAR\_TYPE.MIN1

- **field** – 需要获取的字段类型，例如：['closePrice']，不填的话，默认获取所有字段；可用的 field 包括：[productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **baseDate** – 获取股票复权数据时的默认基准日，可填：日期，'start'，'end'，以及 None，默认为 None：不复权。'YYYY-MM-DD'：以固定日期为复权基准日。若该日不是交易日，向前调整至最近的交易日；'start'：以 startDate 起始日期为基准日。若该日不是交易日，向前调整至最近的交易日；'end'：以 endDate 结束日期为基准日。若该日不是交易日，向前调整至最近的交易日；None：不复权
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame



7.1 获取基本行情

现阶段，如下 api 用于行情的获取：

- GetEquityBarEOD：获取日线数据；
- GetEquityBarMin1：获取 1 分钟线数据；
- GetEquityBarMin5：获取 5 分钟线数据。

以上的数据接口都有类似的使用方式：

```
In [1]: api.GetEquityBarEOD('600000', '2012-10-08', '2015-09-30', \
...: ['closePrice', 'openPrice']).head()
...:
Out[1]:
```

	closePrice	openPrice	instrumentID	tradingDate	tradingTime	\
timeStamp						
2012-10-08	7.32	7.38	600000	2012-10-08	15:00:00.0000000	
2012-10-09	7.45	7.35	600000	2012-10-09	15:00:00.0000000	
2012-10-10	7.44	7.43	600000	2012-10-10	15:00:00.0000000	
2012-10-11	7.41	7.41	600000	2012-10-11	15:00:00.0000000	
2012-10-12	7.45	7.43	600000	2012-10-12	15:00:00.0000000	

  

	securityID
timeStamp	
2012-10-08	600000.xshg
2012-10-09	600000.xshg
2012-10-10	600000.xshg

```
2012-10-11 600000.xshg
2012-10-12 600000.xshg
```

上面的代码, 获取了浦发银行(代码 600000)从 2012 年 10 月 8 日到 2015 年 9 月 30 日的开盘价和收盘价数据。

默认情况下, 行情 api 获取的数据都是未复权的。为了获取复权的数据, 需要在 api 调用中指定 `baseDate` 参数, 完整的函数使用说明可以通过 python 的 `help` 函数查看:

```
help(api.GetEquityBarEOD)
```

Help on function GetEquityBarEOD in module DataAPI.Equity:

```
GetEquityBarMin1(instrumentIDList,
                 startDate,
                 endDate,
                 field="*",
                 chunksize=None,
                 baseDate=None,
                 forceUpdate=True,
                 instrumentIDasCol=False)
```

获取股票日线数据

:param instrumentIDList: 证券名或者列表, 例如: '600000' 或者 ['600000', '000001']

:param startDate: 开始日期, 格式: 'YYYY-MM-DD'

:param endDate: 结束日期, 格式: 'YYYY-MM-DD'

:param field: 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段

:param chunksize: 以分段的形式获取 chunksize 大小的数据

:param baseDate: 获取复权数据时的默认基准日, 可填: 日期, 'start', 'end', 以及 None, 默认为 None: 不复权。

'YYYY-MM-DD': 以固定日期为复权基准日。若该日不是交易日, 向前调整至最近的交易日;

'start': 以 startDate 起始日期为基准日。若该日不是交易日, 向前调整至最近的交易日;

'end': 以 endDate 结束日期为基准日。若该日不是交易日, 向前调整至最近的交易日;

None: 不复权

:param forceUpdate: 当为 True 时强制刷新数据, 不使用缓存。默认为 True

:param instrumentIDasCol: 联合使用 field 以及 instrumentIDs 作为 column 的名字

:return: pandas.DataFrame

例如我们以 2015 年 9 月 30 日为基准日, 获取之前时间段的数据 (即为向前复权方式):

```
In [2]: api.GetEquityBarEOD('600000', '2012-10-08', '2015-09-30', \
...: ['closePrice', 'openPrice'], baseDate='2015-09-30').head()
...:
```

Out[2]:

	closePrice	openPrice	instrumentID	tradingDate	tradingTime	\
timeStamp						
2012-10-08	6.173933	6.224539	600000	2012-10-08	15:00:00.0000000	
2012-10-09	6.283580	6.199236	600000	2012-10-09	15:00:00.0000000	
2012-10-10	6.275145	6.266711	600000	2012-10-10	15:00:00.0000000	
2012-10-11	6.249842	6.249842	600000	2012-10-11	15:00:00.0000000	
2012-10-12	6.283580	6.266711	600000	2012-10-12	15:00:00.0000000	
securityID						
timeStamp						
2012-10-08	600000.xshg					
2012-10-09	600000.xshg					
2012-10-10	600000.xshg					
2012-10-11	600000.xshg					
2012-10-12	600000.xshg					

## 7.2 分批获取数据

有的时候，有获取大量数据的需求，例如拿全市场股票的过去 3 年的分钟线。在这样的情况下，一次获取全部数据并且塞入内存，会对机器产生极大的负担。特别是在一些内存容量比较低的老机器上面，可能会使界面僵死甚至程序崩溃。所以为了缓和这种情况，行情 api 中我们加入了分批次（chunk）获取数据的功能，通过 `chunksize` 关键字参数实现：

```
In [3]: data = api.GetEquityBarMin5('600000', '2012-10-08', '2015-09-30', \
...: ['closePrice', 'openPrice'], chunksize=5)
...:
```

```
In [4]: data
```

```
Out[4]: <generator object _GetBarDataInChunk at 0x00000209EFB88308>
```

在这样的情况下，我们看到，现在返回的是一个 python 的生成器（generator）。用户可以通过遍历的方法，获取全部数据，每次获取 5 个数据点（`chunksize=5`）：

```
In [5]: print(next(data))
```

	closePrice	openPrice	instrumentID	tradingDate	\
timeStamp					
2012-10-08 09:30:00	7.39	7.38	600000	2012-10-08	
2012-10-08 09:35:00	7.40	7.39	600000	2012-10-08	

2012-10-08 09:40:00	7.39	7.41	600000	2012-10-08
2012-10-08 09:45:00	7.39	7.40	600000	2012-10-08
2012-10-08 09:50:00	7.37	7.39	600000	2012-10-08
	tradingTime	securityID		
timeStamp				
2012-10-08 09:30:00	09:30:00.00000000	600000.xshg		
2012-10-08 09:35:00	09:35:00.00000000	600000.xshg		
2012-10-08 09:40:00	09:40:00.00000000	600000.xshg		
2012-10-08 09:45:00	09:45:00.00000000	600000.xshg		
2012-10-08 09:50:00	09:50:00.00000000	600000.xshg		

### 7.3 使用缓存加速获取数据

获取大量的数据，会消耗许多的时间。如果用户需要反复获取相同的数据，总是从远端服务器读取会将大量的时间浪费在传输上，得不偿失。

为了节省这部分无谓的消耗,DataAPI 设计了缓存的机制,这个功能可以通过设置函数参数: `forceUpdate` 为 `False` 来开启。在缓存机制开启的状态下, DataAPI 会优先尝试从本地缓存获取数据; 如果获取失败, 然后才会向远端服务器请求数据。

**警告:** 缓存机制开启的时候, 需要在当前的运行目录下建立缓存文件夹。所以用户需要确保在当前目录下有写权限。如果 DataAPI 发现本地缓存建立失败, 会直接尝试从远端获取数据。

缓存机制作为一个高级功能, 需要本地建立文件夹保存数据, 会产生副作用, 不适合不了解该功能的用户使用。所以默认情况下缓存机制并不会开启。

**注解:** 缓存机制对于所有的行情 DataAPI 都是有效的, 例如下面的指数、期货。

```
# 读取 50 支股票 2015 年的 5 分钟线数据
import time
from DataAPI import api
from AlgoTrading.Data import set_universe
tickers = set_universe('000300.zicn')[:50]

# 第一次读取, 尚未建立缓存, 数据会从远端服务器获取
start = time.time()
data = api.GetEquityBarMin5(tickers, '2015-01-01', '2015-10-10', forceUpdate=False)
```

```
print(u"直接从远程服务器获取耗时: %.2fs" % (time.time() - start))

# 第二次读取, 直接使用本地缓存
start = time.time()

data = api.GetEquityBarMin5(tickers, '2015-01-01', '2015-10-10', forceUpdate=False)

print(u"从缓存获取耗时: %.2fs" % (time.time() - start))
```

表 7.1: 缓存性能比较

方式	运行时间 (秒)
直接读取	10.18
使用缓存	0.38

如果缓存获取成功的话, 运行窗口中会打印类似的语句:

```
{'args': ([ '000001.XSHE', '000002.XSHE', '000009.XSHE' ... ], '2015-01-01', '2015-10-10',
 '*', None, None, False), 'name': 'GetEquityBarMin5', 'kwargs': {}} is reading from cach
D:\.dstore\bccf5f7f4a3d6fa8b6c4d1ba6476d4e0e533c00b689b19b83a01f843.hdf...
```

该返回的内容, 主要包含两部分的信息:

- 函数签名

```
{'args': ([ '000001.XSHE', '000002.XSHE', '000009.XSHE' ... ], '2015-01-01', '2015-10-10',
 '*', None, None, False), 'name': 'GetEquityBarMin5', 'kwargs': {}}
```

- 缓存文件位置

```
D:\.dstore\bccf5f7f4a3d6fa8b6c4d1ba6476d4e0e533c00b689b19b83a01f843.hdf
```

具体的缓存文件名与地址与用户的运行环境有关。

## 7.4 将 instrumentID 转为 column index

上面的 DataAPI 演示中, instrumentID 数据都是出现在 column 中。有些时候, 我们希望将 instrumentID 作为 column index 使用, 这样方便同一字段的数据在不同证券之间进行比较。在 api 中, 通过 instrumentIDasCol 参数来实现该功能。

例如下面的例子:

```
In [6]: data = api.GetEquityBarEOD(['600000.xshg', '000001.xshe'],
...:                               '2012-01-01',
...:                               '2015-12-12',
...:                               field=['openPrice', 'closePrice', 'turnover'],
```

```
...: instrumentIDasCol=True)
...:
```

返回的结果:

```
In [7]: data.tail()
```

```
Out[7]:
```

	openPrice		closePrice		turnover		\
instrumentID	000001	600000	000001	600000	000001	600000	
timeStamp							
2015-12-07	12.18	18.71	12.15	19.45	488370699.4	1592198369	
2015-12-08	12.07	19.28	11.96	20.10	602568625.9	4764109259	
2015-12-09	11.92	19.58	11.99	19.23	515596474.9	1776578423	
2015-12-10	11.95	19.34	11.96	19.27	502326072.6	1108934588	
2015-12-11	11.91	19.01	11.83	18.60	448800941.4	1630390309	

  

	securityID	
instrumentID	000001	600000
timeStamp		
2015-12-07	000001.xshe	600000.xshg
2015-12-08	000001.xshe	600000.xshg
2015-12-09	000001.xshe	600000.xshg
2015-12-10	000001.xshe	600000.xshg
2015-12-11	000001.xshe	600000.xshg

如果要获取这里面所有证券收盘价:

```
In [8]: data['closePrice'].tail()
```

```
Out[8]:
```

instrumentID	000001	600000
timeStamp		
2015-12-07	12.15	19.45
2015-12-08	11.96	20.10
2015-12-09	11.99	19.23
2015-12-10	11.96	19.27
2015-12-11	11.83	18.60

如果只是想获取单支证券的所有信息:

```
In [9]: data.loc[:, pd.IndexSlice[:, '600000']].tail()
```

```
Out[9]:
```

	openPrice	closePrice	turnover	securityID
--	-----------	------------	----------	------------

instrumentID	600000	600000	600000	600000
timeStamp				
2015-12-07	18.71	19.45	1592198369	600000.xshg
2015-12-08	19.28	20.10	4764109259	600000.xshg
2015-12-09	19.58	19.23	1776578423	600000.xshg
2015-12-10	19.34	19.27	1108934588	600000.xshg
2015-12-11	19.01	18.60	1630390309	600000.xshg

获取单支证券的开盘价的话:

```
In [10]: data['openPrice', '600000'].tail()
Out[10]:
timeStamp
2015-12-07    18.71
2015-12-08    19.28
2015-12-09    19.58
2015-12-10    19.34
2015-12-11    19.01
Name: (openPrice, 600000), dtype: float64
```

## 7.5 函数列表

Created on 2015-11-12

@author: cheng.li, weijun.shen

`DataAPI.Equity.GetEquityBarEOD(instrumentIDList, startDate, endDate, field='*', chunksize=None, baseDate=None, forceUpdate=True, instrumentIDasCol=False)`

获取股票日线数据

### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '600000' 或者 ['600000', '000001']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据

- **baseDate** – 获取复权数据时的默认基准日，可填：日期，‘start’，‘end’，以及 None，默认为 None：不复权。‘YYYY-MM-DD’：以固定日期为复权基准日。若该日不是交易日，向前调整至最近的交易日；‘start’：以 startDate 起始日期为基准日。若该日不是交易日，向前调整至最近的交易日；‘end’：以 endDate 结束日期为基准日。若该日不是交易日，向前调整至最近的交易日；None：不复权
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Equity.GetEquityBarMin1(instrumentIDList, startDate, endDate, field='*', chunk-size=None, baseDate=None, forceUpdate=True, instrumentIDasCol=False)
```

获取股票 1 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表，例如：‘600000’ 或者 [‘600000’, ‘000001’]
- **startDate** – 开始日期，格式：‘YYYY-MM-DD’
- **endDate** – 结束日期，格式：‘YYYY-MM-DD’
- **field** – 需要获取的字段类型，例如：[‘closePrice’]，不填的话，默认获取所有字段；可用的 field 包括：[productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **baseDate** – 获取复权数据时的默认基准日，可填：日期，‘start’，‘end’，以及 None，默认为 None：不复权 ‘YYYY-MM-DD’：以固定日期为复权基准日。若该日不是交易日，向前调整至最近的交易日；‘start’：以 startDate 起始日期为基准日。若该日不是交易日，向前调整至最近的交易日；‘end’：以 endDate 结束日期为基准日。若该日不是交易日，向前调整至最近的交易日；None：不复权
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Equity.GetEquityBarMin5(instrumentIDList, startDate, endDate, field='*', chunk-size=None, baseDate=None, forceUpdate=True, instrumentIDasCol=False)
```

获取股票 5 分钟线数据



### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '600000' 或者 ['600000', '000001']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **baseDate** – 获取复权数据时的默认基准日, 可填: 日期, 'start', 'end', 以及 None, 默认为 None: 不复权 'YYYY-MM-DD': 以固定日期为复权基准日。若该日不是交易日, 向前调整至最近的交易日; 'start': 以 startDate 起始日期为基准日。若该日不是交易日, 向前调整至最近的交易日; 'end': 以 endDate 结束日期为基准日。若该日不是交易日, 向前调整至最近的交易日; None: 不复权
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

## 8.1 行情信息

现阶段，如下 api 用于行情的获取：

- `GetIndexBarEOD`: 获取日线数据；
- `GetIndexBarMin1`: 获取 1 分钟线数据；
- `GetIndexBarMin5`: 获取 5 分钟线数据。

## 8.2 成分股信息

可以通过下面的 api 获取指定指数在某日的成分股信息：

- `GetIndexConstitutionInfo`

该 api 具有如下的参数要求：

- `instrumentIDList`: 指数名称或者列表，例如：‘000300’ 或者 [‘000300’]
- `refDate`: 基准日，格式：‘YYYY-MM-DD’；不填的话，默认当前日期最新
- `field`: 需要获取的字段类型，例如：[‘inDate’]，不填的话，默认获取所有字段

```
In [1]: api.GetIndexConstitutionInfo('000300',
...:                                field=['instrumentID', 'conInstrumentID']) \
...:                                .tail()
...:

Out[1]:
   instrumentID conInstrumentID
295         000300         000768
```

296	000300	000540
297	000300	601628
298	000300	600048
299	000300	000895

你可以指定具体的日期：

```
In [2]: api.GetIndexConstitutionInfo('000300',
...:                                refDate='2015-01-01',
...:                                field=['instrumentID', 'conInstrumentID']) \
...:                                .tail()
...:
```

```
Out[2]:
   instrumentID  conInstrumentID
295      000300          600489
296      000300          300024
297      000300          000883
298      000300          600036
299      000300          000895
```

## 8.3 函数列表

Created on 2015-11-12

@author: cheng.li, weijun.shen

`DataAPI.Index.GetIndexBarEOD(instrumentIDList, startDate, endDate, field='*', chunksize=None, forceUpdate=True, instrumentIDasCol=False)`

获取指数日线数据

### 参数

- **instrumentIDList** – 证券名或者列表，例如：‘000300’ 或者 [‘000300’]
- **startDate** – 开始日期，格式：‘YYYY-MM-DD’
- **endDate** – 结束日期，格式：‘YYYY-MM-DD’
- **field** – 需要获取的字段类型，例如：[‘closePrice’]，不填的话，默认获取所有字段；可用的 field 包括：[productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据，不使用缓存。默认为 True

- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Index.GetIndexBarMin1(instrumentIDList, startDate, endDate, field='*', chunksize=None,  
forceUpdate=True, instrumentIDasCol=False)
```

获取指数 1 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '000300' 或者 ['000300']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Index.GetIndexBarMin5(instrumentIDList, startDate, endDate, field='*', chunksize=None,  
forceUpdate=True, instrumentIDasCol=False)
```

获取指数 5 分钟线数据

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '000300' 或者 ['000300']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Index.GetIndexConstitutionInfo(instrumentIDList, refDate=None, field='*', forceUpdate=True)
```

获取指定日的指数成分股信息

#### 参数

- **instrumentIDList** – 证券名或者列表, 例如: '000300' 或者 ['000300']
- **refDate** – 基准日, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['inDate'], 不填的话, 默认获取所有字段; 可用的 field 包括: [instrumentID, windCode, conInstrumentID, conWindCode, inDate, outDate]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

## 9.1 函数列表

Created on 2015-11-12

@author: cheng.li, weijun.shen

```
DataAPI.Future.GetFutureBarEOD(instrumentIDList, startDate, endDate, field='*', chunksize=None,  
                                forceUpdate=True, instrumentIDasCol=False)
```

获取期货日线数据

### 参数

- **instrumentIDList** – 证券名或者列表, 例如: 'IF1506' 或者 ['IF1502', 'IH1506']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Future.GetFutureBarEODContinuing(productIDList, startDate, endDate, field='*',  
                                           chunksize=None, forceUpdate=True, productIDasCol=False)
```

获取期货连续合约日线数据

### 参数

- **productIDList** – 证券名或者列表, 例如: 'IF' 或者 ['IF', 'IH']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **productIDasCol** – 联合使用 field 以及 productIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Future.GetFutureBarMin1(instrumentIDList, startDate, endDate, field='*', chunk-
                                size=None, forceUpdate=True, instrumentIDasCol=False)
```

获取期货 1 分钟线数据

### 参数

- **instrumentIDList** – 证券名或者列表, 例如: 'IF1506' 或者 ['IF1502', 'IH1506']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover, matchItems]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Future.GetFutureBarMin5(instrumentIDList, startDate, endDate, field='*', chunk-
                                size=None, forceUpdate=True, instrumentIDasCol=False)
```

获取期货 5 分钟线数据

### 参数

- **instrumentIDList** – 证券名或者列表, 例如: 'IF1506' 或者 ['IF1502', 'IH1506']

- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover, matchItems]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.Future.GetFutureBarMin5Continuing(productIDList, startDate, endDate, field='*',
                                             chunksize=None, forceUpdate=True, productIDasCol=False)
```

获取期货连续合约 5 分钟线数据

#### 参数

- **productIDList** – 证券名或者列表, 例如: 'IF' 或者 ['IF', 'IH']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, instrumentID, tradingDate, tradingTime, openPrice, highPrice, lowPrice, closePrice, volume, turnover, matchItems]
- **chunksize** – 以分段的形式获取 chunksize 大小的数据
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **productIDasCol** – 联合使用 field 以及 productIDs 作为 column 的名字

返回 pandas.DataFrame



## 10.1 获取大宗商品库存数据

我们有如下的 api 获取各地每日库存的信息：

- GetCommodityStocksInfo

使用方式如下：

```
In [1]: api.GetCommodityStocksInfo('CU', '2015-01-01', '2016-01-01').head()
```

```
Out[1]:
```

	tradingDate	productID	wareHouse	stocks
timeStamp				
2015-01-05	2015-01-05	CU	浙江：宁波九龙仓	0
2015-01-05	2015-01-05	CU	江苏：上港物流苏州	0
2015-01-05	2015-01-05	CU	上海：合计	21706
2015-01-05	2015-01-05	CU	上海：中储吴淞	6176
2015-01-05	2015-01-05	CU	浙江：国储 837 处	0

用户也可以指定具体的仓储地：

```
In [2]: api.GetCommodityStocksInfo('CU', '2015-01-01', '2016-01-01', wareHouse=[u'总计', u'合计']).head()
```

```
Out[2]:
```

	tradingDate	productID	wareHouse	stocks
timeStamp				
2015-01-05	2015-01-05	CU	总计	30627
2015-01-06	2015-01-06	CU	总计	30827
2015-01-07	2015-01-07	CU	总计	30827
2015-01-08	2015-01-08	CU	总计	31128
2015-01-09	2015-01-09	CU	总计	32104

也可以同时关心多个品种的库存:

```
In [3]: api.GetCommodityStocksInfo(['CU', 'PP'], '2016-02-17', '2016-02-17', wareHouse=[u'总计', u'合计'])
Out[3]:
```

	tradingDate	productID	wareHouse	stocks
timeStamp				
2016-02-17	2016-02-17	PP	合计	NaN
2016-02-17	2016-02-17	CU	总计	116655

## 10.2 函数列表

Created on 2016-2-16

@author: cheng.li, weijun.shen

```
DataAPI.Commodity.GetCommodityStocksInfo(productIDList, startDate, endDate, wareHouse='',
                                             field='', forceUpdate=True)
```

获取商品期货的每日仓单库存信息

### 参数

- **productIDList** – 证券名或者列表, 例如: 'IF' 或者 ['IF', 'IH']
- **startDate** – 开始日期, 格式: 'YYYY-MM-DD'
- **endDate** – 结束日期, 格式: 'YYYY-MM-DD'
- **wareHouse** – 所在仓库, 例如: [u' 江苏: 中储无锡']。默认为'', 获取全部仓库的信息
- **field** – 需要获取的字段类型, 例如: ['closePrice'], 不填的话, 默认获取所有字段; 可用的 field 包括: [productID, tradingDate, wareHouse, stocks]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

---

## 量化基金研究库

---

### 11.1 获取在池基金信息

现阶段我们按照以下标砖挑选符合标准的量化基金：

- 投资范围（investScope）包含股票、期货、期权、基金、ETF、债券或者二级市场；
- 到期日大于当前日期或者不存在到期日；
- 有投顾信息；
- 没有锁定日的条款或者信息；
- 最近一年净值公布信息次数大于等于 50 次；

这个池会滚动变化，周期约为 1 个月。可以通过如下的方式，使用 `GetHedgeFundPool` 获取指定日的基金池：

```
In [1]: api.GetHedgeFundPool(refDate='2016-03-22', field=['instrumentID', 'fullName']).head()
```

```
Out[1]:
```

	eventDate	instrumentID	fullName
0	20160115	XT090776.XT	奕金安一期证券投资信托
1	20160115	XT090981.XT	投资精英之朱雀证券投资信托（A）
2	20160115	XT091121.XT	投资精英之朱雀证券投资信托（B）
3	20160115	XT101310.XT	富锦 7 号（汇利）证券投资信托
4	20160115	XT110161.XT	黄金优选五期 16 号信托

### 11.2 获取特定基金的信息

用户也可以通过基金代码直接查询信息：

```
In [2]: api.GetHedgeFundInfo(instruments=['XT142798.XT'])
```

```
Out[2]:
```

```

      name fundManagementComp \
0  艾方多策略对冲套利 1 期      艾方资产

      investScope instrumentID \
0  本信托计划投资于中国金融期货交易所挂牌交易的股指期货产品、国内证券交易所挂牌交易的 A 股股票... XT142798.XT

      fullName firstInvestType maturityDate
0  艾方多策略对冲套利一期信托      股票型      None

```

或者也可以通过投资类型，输出在 wind 分类中符合这一投资类型标准的基金：

```
In [3]: api.GetHedgeFundInfo(firstInvestType=u'固定收益型').head()
```

```
Out[3]:
```

```

      name fundManagementComp \
0  佑瑞持第 9 期      佑瑞持投资
1  小金牛固收 2 号      东航金控
2  汇势通方相纯债 1 号      汇势通投资
3  汇势通相德纯债 1 期      汇势通投资
4  融信力通 1 号      融信力通投资

      investScope instrumentID \
0  主要投资于国债、央行票据、金融债、短期融资券、中期票据、企业债等银行间债券市场交易的品种，公... XT125045.XT
1  通过积极投资证券市场、债券市场、股指期货、商品期货和国债期货等，灵活采取多种投资策略，谋求私... XT1410445.XT
2  本基金的投资范围为本基金主要投资于交易所和银行间债券（国债、企业债、公司债及资产支持证券等）... XT144790.XT
3  本基金主要投资于交易所和银行间债券（国债、企业债、公司债及资产支持证券等）、国债逆回购、公开... XT145164.XT
4  信托资金投资于“河南晋开化工投资控股集团有限责任公司 2013-2015 年度非公开定向债务融资... XT145346.XT

      fullName firstInvestType maturityDate
0  佑瑞持债券投资第九期信托      固定收益型 20,181,207.00
1  小金牛固收 2 号私募基金      固定收益型      nan
2  汇势通方相纯债 1 号基金      固定收益型      nan
3  汇势通相德纯债一期基金      固定收益型      nan
4  融信力通 1 号债券投资信托      固定收益型      nan

```

## 11.3 获取基金净值表现

现阶段，我们分别按周以及按月的方式采集基金的净值数据。

## 1 使用 GetHedgeFundBarWeek 获取周线

```
In [4]: api.GetHedgeFundBarWeek(instruments=['XT142798.XT', 'XT149942.XT', 'XT1410935.XT'], \
...: startDate='2015-01-01', endDate='2016-03-11', instrumentIDasCol=True)['navAcc'].head()
...:
```

Out[4]:

instrumentID	XT1410935.XT	XT142798.XT	XT149942.XT
timeStamp			
2015-01-02	1.00	1.57	1.00
2015-01-09	1.00	1.58	1.00
2015-01-16	1.00	1.59	1.00
2015-01-23	1.00	1.58	0.98
2015-01-30	1.00	1.57	0.99

## 2 使用 GetHedgeFundBarMonth 获取月线

```
In [5]: api.GetHedgeFundBarMonth(instruments=['XT142798.XT', 'XT149942.XT', 'XT1410935.XT'], \
...: startDate='2015-01-01', endDate='2016-03-11', instrumentIDasCol=True)['navAcc'].head()
...:
```

Out[5]:

instrumentID	XT1410935.XT	XT142798.XT	XT149942.XT
timeStamp			
2015-01-30	1.00	1.57	0.99
2015-02-27	1.01	1.62	1.02
2015-03-27	1.21	1.77	1.02
2015-04-24	1.41	1.88	1.05
2015-05-29	1.51	2.02	1.22

## 11.4 获取基金风险收益指标

```
In [6]: api.GetHedgeFundPerfComparison(instruments=['XT090776.XT', 'XT090981.XT', 'XT091121.XT', 'XT101310.XT',
...: startDate='2015-01-01', endDate='2016-03-11')
...:
```

Out[6]:

	基金代码	参考起始日	参考结束日	年化收益	波动率	最大回撤	sharp	sortino	收益回撤比	\
0	XT090981.XT	2015-01-02	2016-03-11	0.16	0.30	0.45	0.53	0.66	0.35	
1	XT091121.XT	2015-01-02	2016-03-11	0.16	0.30	0.45	0.53	0.66	0.35	
2	XT090776.XT	2015-01-02	2016-03-11	0.09	0.26	0.24	0.36	0.44	0.40	
3	XT101310.XT	2015-01-02	2016-03-11	-0.02	0.42	0.76	-0.04	-0.06	-0.02	
4	XT110161.XT	2015-01-02	2016-03-11	-0.09	0.31	0.60	-0.29	-0.37	-0.15	

	年化收益得分	sharp 得分	最大回撤得分	sortino 得分	收益回撤比得分	波动率得分	总分
0	8.00	0.00	0.00	10.00	0.00	5.00	23.00
1	8.00	0.00	0.00	10.00	0.00	2.00	20.00
2	4.00	0.00	0.00	5.00	0.00	5.00	14.00
3	0.00	0.00	0.00	5.00	0.00	2.00	7.00
4	0.00	0.00	0.00	5.00	0.00	2.00	7.00

## 11.5 获取基金的风格分析

### 1 通过代码获取

```
In [7]: api.GetHedgeFundStyleAnalysis(instruments='XT133479.XT', startDate='2015-02-23', endDate='2016-02-23')
XT133479.XT style analysis with 11 data observations.
Out[7]:
          债券型  多空仓型  宏观策略  市场中性  管理期货  股票型  货币型
XT133479.XT 0.13 -0.00 -0.00  0.50  0.38 -0.00 0.00
```

### 2. 通过输入净值表

可以首先准备外部数据, 例如存放在 excel 中:

```
In [8]: nav_table = pd.read_excel('data/XT1515811.xlsx', index_col='tradingDate')
...: nav_table.head()
...:
Out[9]:
          XT1515811
tradingDate
2015-07-30          1.00
2015-08-28          1.00
2015-09-02          1.00
2015-09-11          1.02
2015-09-18          1.01
```

然后使用 `GetHedgeFundStyleAnalysis` 函数进行分析:

```
In [10]: api.GetHedgeFundStyleAnalysis(navTable=nav_table)
XT1515811 style analysis with 7 data observations.
Out[10]:
          债券型  多空仓型  宏观策略  市场中性  管理期货  股票型  货币型
XT1515811 -0.00 -0.00  0.26  0.00  0.64 0.00 0.10
```

## 11.6 函数列表

Created on 2016-1-4

@author: cheng.li

```
DataAPI.HedgeFund.GetHedgeFundBarMonth(instruments=None, startDate='2003-01-01',
                                         endDate='2015-12-31', field='*', forceUpdate=True,
                                         instrumentIDasCol=False)
```

获取对冲基金历史表现信息 (月)

### 参数

- **instruments** – 基金代码列表, 例如: 'XT1515141.XT'。默认为 None, 获取所有基金历史表现
- **field** – 需要获取的字段类型, 例如: ['logRetAcc'], 不填的话, 默认获取所有字段; 可用的 field 包括: [tradingDate, instrumentID, navUnit, navAcc, logRetUnit, logRetAcc]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.HedgeFund.GetHedgeFundBarWeek(instruments=None, startDate='2003-01-01',
                                         endDate='2015-12-31', field='*', forceUpdate=True,
                                         instrumentIDasCol=False)
```

获取对冲基金历史表现信息 (周)

### 参数

- **instruments** – 基金代码列表, 例如: 'XT1515141.XT'。默认为 None, 获取所有基金历史表现
- **field** – 需要获取的字段类型, 例如: ['logRetAcc'], 不填的话, 默认获取所有字段; 可用的 field 包括: [tradingDate, instrumentID, navUnit, navAcc, logRetUnit, logRetAcc]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字

返回 pandas.DataFrame

```
DataAPI.HedgeFund.GetHedgeFundInfo(instruments=None, instrumentName=None, field='*',
                                     firstInvestType=None, forceUpdate=True)
```

获取对冲基金基础信息

### 参数

- **instruments** – 基金代码列表, 例如: 'XT1515141.XT'。默认为 None, 获取所有的基金信息
- **field** – 需要获取的字段类型, 例如: ['firstInvestType'], 不填的话, 默认获取所有字段; 可用的 field 包括: [instrumentID, fullName, firstInvestType, investScope, maturityDate, advisory]
- **firstInvestType** – 需要获取的基金所属的投资类型列表, 例如: [u' 期货型']
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

```
DataAPI.HedgeFund.GetHedgeFundPerfComparison(instruments=None, startDate='2015-01-01',  
endDate='2015-12-31', forceUpdate=True)
```

获取指定基金的历史表现风险收益指标以及打分结果。

### 参数

- **instruments** – 基金 wind 代码列表, 可以为空。若为空, 则获取所有在池基金的结果。
- **startDate** – 历史表现起始日
- **endDate** – 历史表现结束日
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

```
DataAPI.HedgeFund.GetHedgeFundPool(field='*', refDate=None, forceUpdate=True)
```

获取指定日期下, 基金备选池的信息

### 参数

- **field** – 需要获取的字段类型, 例如: ['logRetAcc'], 不填的话, 默认获取所有字段; 可用的 field 包括: [tradingDate, instrumentID, navUnit, navAcc, logRetUnit, logRetAcc]
- **refDate** – 指定日期, 将查询范围限制于当日在基金备选池中的基金, 格式为: YYYY-MM-DD; 不填的话, 默认查询最新的备选池信息
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame



```
DataAPI.HedgeFund.GetHedgeFundStyleAnalysis(instruments=None, navTable=Empty DataFrame
                                              Columns: [] Index: [], startDate='2012-01-01',
                                              endDate='2015-01-01', forceUpdate=True)
```

获取根据基金历史表现得到的风格分析的结果

#### 参数

- **instruments** – 基金 wind 代码列表, 可以为空。若为空, 则使用用户输入的 navTable
- **navTable** – N\* (M+1) 的 pandas.DataFrame, 表示 M 个基金历史上 N 个时间点的净值
- **startDate** – 历史表现起始日, 在使用 navTable 时候会被忽略

:param endDate: 历史表现结束日, 在使用 navTable 时候会被忽略: param forceUpdate: 当为 True 时强制刷新数据, 不使用缓存。默认为 True :return: pandas.DataFrame

```
DataAPI.HedgeFund.GetHedgeFundStylePerf(styleName=None, startDate='2012-01-01',
                                          endDate='2015-12-31', instrumentIDasCol=False,
                                          forceUpdate=True)
```

获取特定风格的基金类型的历史表现

#### 参数

- **styleName** – 基金风格类型, 可选的风格有: ['u' 债券型', 'u' 多空仓型', 'u' 宏观策略', 'u' 市场中性', 'u' 管理期货', 'u' 股票型', 'u' 货币型']
- **startDate** – 历史表现起始日期
- **endDate** – 历史表现结束日期
- **instrumentIDasCol** – 联合使用 field 以及 instrumentIDs 作为 column 的名字
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

## 主题与情感信息

现阶段，如下 api 用于主题信息的获取：

- `GetThemeInfo`：获取主题列表；
- `GetStocksByTheme`：获取主题相关个股；
- `GetThemeHotness`：获取主题热度；
- `GetActiveThemesRelatedStocks`：获取热门股票热门股。

### 12.1 获取主题列表

```
In [1]: api.GetThemeInfo(themeName=u'火箭军')
```

```
Out[1]:
```

	isActive	themeName	updateTime	themeID	insertTime
0	0	火箭军	2016-03-24 09:14:08	122275	2016-01-08 09:03:28

### 12.2 获取主题相关个股

```
In [2]: api.GetStocksByTheme(themeName=u'火箭军')
```

```
Out[2]:
```

	cnName	themeID	instrumentID	themeName	exchangeName	score
0	江南红箭	122275	000519	火箭军	深圳证券交易所	0.74
1	烽火电子	122275	000561	火箭军	深圳证券交易所	0.94
2	中信国安	122275	000839	火箭军	深圳证券交易所	0.00
3	中国卫星	122275	600118	火箭军	上海证券交易所	0.00
4	航天通信	122275	600677	火箭军	上海证券交易所	0.63
5	航天长峰	122275	600855	火箭军	上海证券交易所	1.00

## 12.3 获取主题热度

```
In [3]: api.GetThemeHotness(themeName=u'长江经济',
...:                        startDate='2016-01-01',
...:                        endDate='2016-01-08')
...:
```

Out[3]:

	themeName	date	newsNumPercent	themeID	newsNum
0	长江经济带	2016-01-01	0.70	346	45
1	长江经济带	2016-01-02	0.24	346	13
2	长江经济带	2016-01-03	0.51	346	34
3	长江经济带	2016-01-04	0.51	346	169
4	长江经济带	2016-01-05	0.29	346	97
5	长江经济带	2016-01-06	0.49	346	157
6	长江经济带	2016-01-07	0.82	346	232
7	长江经济带	2016-01-08	1.21	346	341
8	长江经济	2016-01-01	0.70	1664	45
9	长江经济	2016-01-02	0.24	1664	13
10	长江经济	2016-01-03	0.51	1664	34
11	长江经济	2016-01-04	0.51	1664	169
12	长江经济	2016-01-05	0.29	1664	97
13	长江经济	2016-01-06	0.49	1664	157
14	长江经济	2016-01-07	0.82	1664	232
15	长江经济	2016-01-08	1.21	1664	341

## 12.4 获取热门股票热门股

```
In [4]: api.GetActiveThemesRelatedStocks(refDate='2015-01-07',
...:                                       topThemes=5,
...:                                       topStocks=5)
...:
```

Out[4]:

	cnName	date	themeHotness	instrumentID	themeName	themeID	score
0	中国太保	2015-01-07	2,184.00	601601	证券	2210	0.91
1	中国平安	2015-01-07	2,184.00	601318	证券	2210	0.87
2	中国人寿	2015-01-07	2,184.00	601628	证券	2210	0.86
3	西水股份	2015-01-07	2,184.00	600291	证券	2210	0.75
4	新华保险	2015-01-07	2,184.00	601336	证券	2210	0.39

5	华夏银行	2015-01-07	2,017.00	600015	银行	46	1.00
6	光大银行	2015-01-07	2,017.00	601818	银行	46	0.98
7	建设银行	2015-01-07	2,017.00	601939	银行	46	0.96
8	农业银行	2015-01-07	2,017.00	601288	银行	46	0.91
9	中国银行	2015-01-07	2,017.00	601988	银行	46	0.88
10	南京中北	2015-01-07	1,352.00	000421	综合	2443	1.00
11	东湖高新	2015-01-07	1,352.00	600133	综合	2443	1.00
12	广弘控股	2015-01-07	1,352.00	000529	综合	2443	1.00
13	天山纺织	2015-01-07	1,352.00	000813	综合	2443	1.00
14	创元科技	2015-01-07	1,352.00	000551	综合	2443	1.00
15	华懋科技	2015-01-07	849.00	603306	汽车	1336	1.00
16	禾嘉股份	2015-01-07	849.00	600093	汽车	1336	1.00
17	广东鸿图	2015-01-07	849.00	002101	汽车	1336	1.00
18	双环传动	2015-01-07	849.00	002472	汽车	1336	0.99
19	精锻科技	2015-01-07	849.00	300258	汽车	1336	0.99
20	湖南黄金	2015-01-07	843.00	002155	黄金	66	1.00
21	恒邦股份	2015-01-07	843.00	002237	黄金	66	0.91
22	ST 荣华	2015-01-07	843.00	600311	黄金	66	0.91
23	中金黄金	2015-01-07	843.00	600489	黄金	66	0.90
24	山东黄金	2015-01-07	843.00	600547	黄金	66	0.88

## 12.5 函数列表

Created on 2016-1-7

@author: cheng.li, weijun.shen, yuchen.lai

`DataAPI.Themes.GetActiveThemesRelatedStocks(refDate=None, windows=None, topThemes=20, topStocks=20, field='*', forceUpdate=True)`

获取指定数量的热门主题高相关度的股票

### 参数

- **refDate** – 参考日，格式：YYYY-MM-DD。默认为 None，取当前日期
- **windows** – 参考周期，可填写正数表示日；或者字符串形式的时间长度，例如：1m。默认为 None，只取 1 日
- **topThemes** – 选取排名靠前多少的主题，默认值为 20
- **topStocks** – 在同主题下选取排名靠前多少的股票，默认值为 20
- **field** – 需要获取的字段类型，例如：['instrumentID']，不填的话，默认获取

所有字段; 可用的 field 包括: [themeID, themeName, instrumentID, cnName, date, themeHotness, score]

- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

`DataAPI.Themes.GetStocksByTheme(themeName, refDate=None, field='*', forceUpdate=True)`

获取指定主题相关的股票

#### 参数

- **themeName** – 主题名称, 例如: u' 金融'
- **refDate** – 参考日, 格式: YYYY-MM-DD
- **field** – 需要获取的字段类型, 例如: ['instrumentID'], 不填的话, 默认获取所有字段; 可用的 field 包括: [themeID, themeName, instrumentID, cnName, exchangeName, score]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

`DataAPI.Themes.GetThemeHotness(themeName, startDate, endDate, field='*', forceUpdate=True)`

获取主题热度时间序列

#### 参数

- **themeName** – 主题名称, 例如: u' 金融'
- **startDate** – 起始日, 格式: YYYY-MM-DD
- **endDate** – 结束日, 格式: YYYY-MM-DD
- **field** – 需要获取的字段类型, 例如: ['newsNumPercent'], 不填的话, 默认获取所有字段; 可用的 field 包括: [themeID, themeName, date, newsNum, newsNumPercent]
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 pandas.DataFrame

`DataAPI.Themes.GetThemeInfo(themeName=None, themeID=None, field='*', forceUpdate=True)`

获取相关的主题列表

#### 参数

- **themeName** – 关注的主题名, 支持模糊查找, 默认查找全部主题
- **themeID** – 关注的主题 id, 为整数, 与 themeName 二选一输入

- **field** – 需要获取的字段类型, 例如: ['newsNumPercent'], 不填的话, 默认获取所有字段; 可用的 field 包括: [themeID, themeName, isActive, insertTime, updateTime]
- **forceUpdate** –

返回

---

## 市场快照

---

市场快照可以用来获取实时的市场信息:

### 13.1 行业资金流向快照

可以轻松的获取全部证监会分类行业的资金流向:

```
In [1]: api.GetIndustryNetCashSnapshot().head()
Out[1]:
```

	dateTime	industryID	industryName	netCashFlow
0	2016-03-24 15:00:19	J66	货币金融服务	-1,952,251,929.90
1	2016-03-24 15:00:25	K70	房地产业	-1,434,228,268.27
2	2016-03-24 15:00:37	C27	医药制造业	-2,893,610,105.03
3	2016-03-24 15:00:25	H61	住宿业	-57,770,580.68
4	2016-03-24 15:00:37	C37	铁路、船舶、航空航天和其他运输设备制造业	-707,611,130.35

也可以通过指定的行业 id 或者行业名称查询:

```
In [2]: api.GetIndustryNetCashSnapshot(industryID='J68')
Out[2]:
```

	dateTime	industryID	industryName	netCashFlow
0	2016-03-24 15:00:00	J68	保险业	-475,341,320.24

### 13.2 函数列表

Created on 2016-1-11

@author: cheng.li, weijun.shen, yuchen.lai

`DataAPI.Snapshots.GetIndustryNetCashSnapshot(industryID=None, industryName=None, force-Update=True)`

获取行业资金流向的快照

#### 参数

- **industryID** – 证监会行业分类 id, 可以为列表, 例如: ['J66', 'J68']
- **industryName** – 证监会分类行业名称, 例如: '保险业'
- **forceUpdate** – 当为 True 时强制刷新数据, 不使用缓存。默认为 True

返回 `pandas.DataFrame`



---

面向开发者

---

## 14.1 为自定义的数据 api 开启缓存

访问数据库总是一个比较耗时的过程，假设你有如下的一个 python 中的 sql 访问函数：

```
import pandas as pd
import pymssql

def sqlquery():
    conn = pymssql.connect(host=_HOST, user=_USER, password=_PWD, database=_DB, timeout=_TIME_OUT)
    sql = "select * from EQY_EOD"
    return pd.read_sql(sql, conn)
```

这样一次完整的查询，在作者的开发机上大概运行了 50 秒钟（包括数据库访问时间以及返回结果规整为 pandas.DataFrame 的时间）。

一个很自然的想法是将大规模查询的结果在本地保存，下次需要获取同样的数据时，直接在本地获取，而不是访问远程数据库。由于该功能非常常用，所以开发包中提供了一个单独的装饰器 `enableCache` 来实现这样的功能：

```
import pandas as pd
import pymssql

@enableCache
def sqlquery_cached():
    conn = pymssql.connect(host=_HOST, user=_USER, password=_PWD, database=_DB, timeout=_TIME_OUT)
    sql = "select * from EQY_EOD"
    return pd.read_sql(sql, conn)
```

在经过这样的包装后，我们为原来的函数提供了缓存功能。分别调用该函数两次，第一次会从远程获取

并保存本地，第二次就会直接获取本地缓存。在作者的开发机上运行以下代码：

```
import datetime as dt

start = dt.datetime.now()
print(sqlquery_cached().tail())
print(dt.datetime.now() - start)

start = dt.datetime.now()
print(sqlquery_cached().tail())
print(dt.datetime.now() - start)
```

两次时间比较如下：

表 14.1: 缓存效果

方式	运行时间（秒）
直接读取	52.67
使用缓存	1.98

## 14.2 中文编码处理

在 python2 中使用 pandas 访问 sql server 数据库的时候，如果带有中文字段的话，可能会产生乱码。像下面的例子：

```
def unicode_error():
    conn = pymssql.connect(host=_HOST, user=_USER, password=_PWD, database=_DB, timeout=_TIME_OUT)
    sql = "select * from ASHAREDESCRIPTION"
    return pd.read_sql(sql, conn)
```

如果直接调用可能返回如下的结果：

	OBJECT_ID	S_INFO_WINDCODE	S_INFO_CODE	\
3884	{FFB9AF1E-A93A-4CE9-B0E6-CDF4A387A2E6}	600527.SH	600527	
3885	{FFC3045C-29B5-463F-9660-4FF46E3B6065}	000699.SZ	000699	
3886	{FFE0A4FC-E8A3-4918-B070-5FED495B76F4}	002011.SZ	002011	
3887	{FFE0E7BE-DF71-42CC-92A3-45E1D868E45B}	600129.SH	600129	
3888	{FFE43FAE-C990-40CE-88C4-2B268BEC8919}	002233.SZ	002233	
	S_INFO_NAME	S_INFO_COMPNAME	\	
3884	%ĂĬ_BĬĚ	%ĚŎ%ĂĬ_BĬĚ¹Ě · ÝŎĎĬp¹«Ě%		

```

3885 S*ST%NÜ%(ÍÈÈD)      %ÑÄ%Ê¹½ðµØÖiÖ½¹Ê · ÝÓÐİP¹«È%
3886      ¶Û°²» · %³      Õã%¶Û°²ÈÈ¹¤» · %³¹Ê · ÝÓÐİP¹«È%
3887      İ«%«%ÍÁ      ÖØÇiİ«%«ÊµÖµ(%ÍÁ)¹Ê · ÝÓÐİP¹«È%
3888      ÊpÁÆ%ÍÁ      ¹ã¶«ÊpÁÆ%ÍÁ¹Ê · ÝÓÐİP¹«È%

```

可以注意到, S\_INFO\_NAME 以及 S\_INFO\_COMPNAME 列都显示错误。为了正确的显示中文, 我们提供了 cleanColsForUnicode 这样的中文处理装饰器:

```

@cleanColsForUnicode
def unicode_corrected():
    conn = pymysql.connect(host=_HOST, user=_USER, password=_PWD, database=_DB, timeout=_TIME_OUT)
    sql = "select * from ASHAREDESCRIPTION"
    return pd.read_sql(sql, conn)

```

现在调用 unicode\_corrected 函数, 就会显示正常的结果:

	OBJECT_ID	S_INFO_WINDCODE	S_INFO_CODE	\
3884	{FFB9AF1E-A93A-4CE9-B0E6-CDF4A387A2E6}	600527.SH	600527	
3885	{FFC3045C-29B5-463F-9660-4FF46E3B6065}	000699.SZ	000699	
3886	{FFE0A4FC-E8A3-4918-B070-5FED495B76F4}	002011.SZ	002011	
3887	{FFE0E7BE-DF71-42CC-92A3-45E1D868E45B}	600129.SH	600129	
3888	{FFE43FAE-C990-40CE-88C4-2B268BEC8919}	002233.SZ	002233	

  

	S_INFO_NAME	S_INFO_COMPNAME	\
3884	江南高纤	江苏江南高纤股份有限公司	
3885	S*ST 佳纸 (退市)	佳木斯金地造纸股份有限公司	
3886	盾安环境	浙江盾安人工环境股份有限公司	
3887	太极集团	重庆太极实业 (集团) 股份有限公司	
3888	塔牌集团	广东塔牌集团股份有限公司	

## 14.3 函数列表

Created on 2015-11-12

@author: cheng.li, weijun.shen

**class DataAPI.Utilities.ASSET\_TYPE**

An enumeration.

**class DataAPI.Utilities.BAR\_TYPE**

An enumeration.

`DataAPI.Utilities.cleanColsForUnicode(f, *args, **kwargs)`

装饰器，为返回值为 `pandas.DataFrame` 的函数进行中文编码的转换

#### 参数

- `f` – 返回值为 `pandas.DataFrame` 的函数
- `args` – 准备传递给 `f` 的位置参数
- `kwargs` – 准备传递给 `f` 的关键字参数

返回 `pandas.DataFrame`

`DataAPI.Utilities.enableCache(f, *args, **kwargs)`

装饰器，为返回值为 `pandas.DataFrame` 的函数提供本地缓存功能

#### 参数

- `f` – 返回值为 `pandas.DataFrame` 的函数
- `args` – 准备传递给 `f` 的位置参数
- `kwargs` – 准备传递给 `f` 的关键字参数

返回 `pandas.DataFrame`

- api, 7
- ASSET\_TYPE (DataAPI.Utilities 中的类), 64
- BAR\_TYPE (DataAPI.Utilities 中的类), 64
- change log, 1
- cleanColsForUnicode() (在 DataAPI.Utilities 模块中), 64
- DataAPI.Commodity (模块), 47
- DataAPI.Equity (模块), 36
- DataAPI.Future (模块), 43
- DataAPI.General (模块), 28
- DataAPI.HedgeFund (模块), 52
- DataAPI.Index (模块), 40
- DataAPI.InstrumentInfo (模块), 25
- DataAPI.Snapshots (模块), 60
- DataAPI.Themes (模块), 57
- DataAPI.Utilities (模块), 64
- enableCache() (在 DataAPI.Utilities 模块中), 65
- GetActiveThemesRelatedStocks() (在 DataAPI.Themes 模块中), 57
- GetCommodityStocksInfo() (在 DataAPI.Commodity 模块中), 47
- GetEquityBarEOD() (在 DataAPI.Equity 模块中), 36
- GetEquityBarMin1() (在 DataAPI.Equity 模块中), 37
- GetEquityBarMin5() (在 DataAPI.Equity 模块中), 37
- GetEquityInstrumentInfo() (在 DataAPI.InstrumentInfo 模块中), 25
- GetFutureBarEOD() (在 DataAPI.Future 模块中), 43
- GetFutureBarEODContinuing() (在 DataAPI.Future 模块中), 43
- GetFutureBarMin1() (在 DataAPI.Future 模块中), 44
- GetFutureBarMin5() (在 DataAPI.Future 模块中), 44
- GetFutureBarMin5Continuing() (在 DataAPI.Future 模块中), 45
- GetFutureInstrumentInfo() (在 DataAPI.InstrumentInfo 模块中), 26
- GetGeneralBarData() (在 DataAPI.General 模块中), 28
- GetHedgeFundBarMonth() (在 DataAPI.HedgeFund 模块中), 52
- GetHedgeFundBarWeek() (在 DataAPI.HedgeFund 模块中), 52
- GetHedgeFundInfo() (在 DataAPI.HedgeFund 模块中), 52
- GetHedgeFundPerfComparison() (在 DataAPI.HedgeFund 模块中), 53
- GetHedgeFundPool() (在 DataAPI.HedgeFund 模块中), 53
- GetHedgeFundStyleAnalysis() (在 DataAPI.HedgeFund 模块中), 53
- GetHedgeFundStylePerf() (在 DataAPI.HedgeFund 模块中), 54
- GetIndexBarEOD() (在 DataAPI.Index 模块中), 40

[GetIndexBarMin1\(\)](#) (在 DataAPI.Index 模块中), [41](#)

[GetIndexBarMin5\(\)](#) (在 DataAPI.Index 模块中), [41](#)

[GetIndexConstitutionInfo\(\)](#) (在 DataAPI.Index 模块中), [41](#)

[GetIndustryNetCashSnapshot\(\)](#) (在 DataAPI.Snapshots 模块中), [60](#)

[GetStocksByTheme\(\)](#) (在 DataAPI.Themes 模块中), [58](#)

[GetThemeHotness\(\)](#) (在 DataAPI.Themes 模块中), [58](#)

[GetThemeInfo\(\)](#) (在 DataAPI.Themes 模块中), [58](#)

[主题](#), [55](#)

[商品期货](#), [46](#)

[市场快照](#), [60](#)

[情感](#), [55](#)

[指数](#), [39](#)

[指数期货](#), [43](#)

[更新历史](#), [1](#)

[股票](#), [30](#)

[证券基本信息](#), [21](#)

[证券行情](#), [27](#)

[量化基金研究库](#), [48](#)

[面向开发者](#), [62](#)