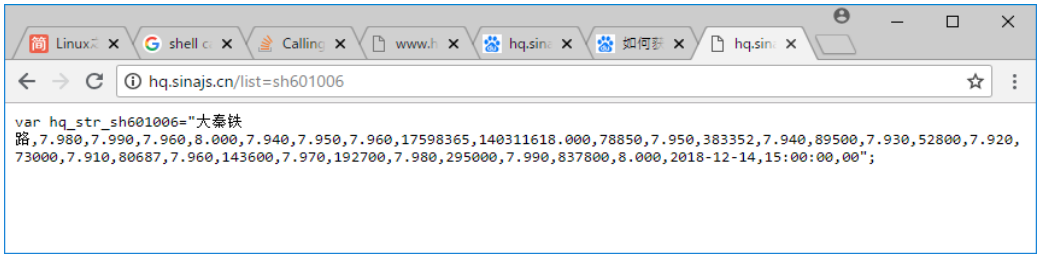
要件设计

20181215 版

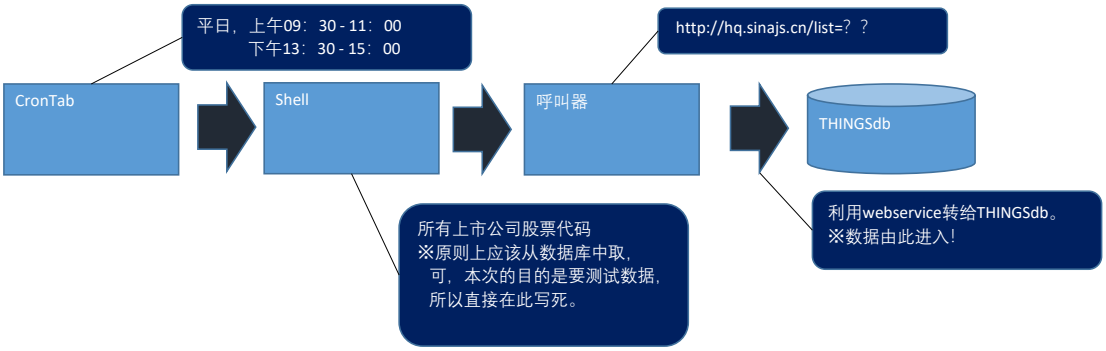
1、数据接口

参考　　https://zhidao.baidu.com/question/166686795.html

http://hq.sinajs.cn/list=sh601006



2、全体图



3、定时Batch

平日，上午09：30 - 11：00
下午13：30 - 15：00

4、Shell

java -cp <路径>/CallStockData.calss <股票代码>

5、Java

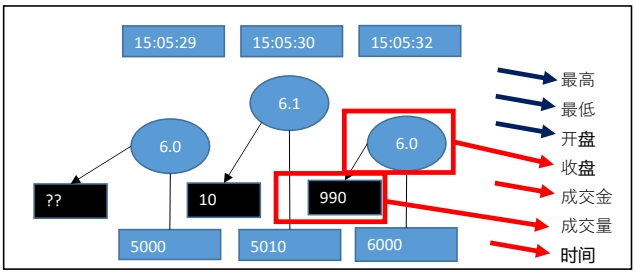
1、做成请求URL // 例，　http://hq.sinajs.cn/list=<股票代码>

2、请求URL // 例,

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.PrintStream;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLConnection;

public class WWWGet {

    public static void main(String[] args) {
        String urlString = "http://hq.sinajs.cn/list=sh600734";
        try {
            URL url = new URL(urlString);
            URLConnection uc = url.openConnection();
            uc.setDoOutput(true);//POST可能にする

            uc.setRequestProperty("User-Agent", "@IT java-tips URLConnection");// ヘッダを設定
            uc.setRequestProperty("Accept-Language", "ja");// ヘッダを設定
            OutputStream os = uc.getOutputStream();//POST用のOutputStreamを取得

            String postStr = "foo1=bar1&foo2=bar2";//POSTするデータ
            PrintStream ps = new PrintStream(os);
            ps.print(postStr);//データをPOSTする
            ps.close();

            InputStream is = uc.getInputStream();//POSTした結果を取得
            BufferedReader reader = new BufferedReader(new InputStreamReader(is));
            String s;
            while ((s = reader.readLine()) != null) {
                System.out.println(s);
            }
            reader.close();
        } catch (MalformedURLException e) {
            System.err.println("Invalid URL format: " + urlString);
            System.exit(-1);
        } catch (IOException e) {
            System.err.println("Can't connect to " + urlString);
            System.exit(-1);
        }
    }
}
```

20181215 版

业务数据



0："大秦铁路"，股票名字；
1："27.55"，今日开盘价；
2："27.25"，昨日收盘价；
3："26.91"，当前价格；　　当前成交价格
4："27.55"，今日最高价；
5："26.20"，今日最低价；
6："26.91"，竞买价，即"买一"报价；
7："26.92"，竞卖价，即"卖一"报价；
8："22114263"，成交的股票数，由于股票交易以一百股为基本单位，所以在使用时，通常把该值除以一百；
9："589824680"，成交金额，单位为"元"，为了一目了然，通常以"万元"为成交金额的单位，所以通常把该值除以一万；
10："4695"，"买一"申请4695股，即47手；
11："26.91"，"买一报价"；
12："57590"，"买二申请股数"
13："26.90"，"买二报价"；
14："14700"，"买三申请股数"
15："26.89"，"买三报价"；
16："14300"，"买四申请股数"
17："26.88"，"买四报价"；
18："15100"，"买五申请股数"
19："26.87"，"买五报价"；
20："3100"，"卖一"申报3100股，即31手；
21："26.92"，　"卖一报价"
22：-　　卖二申请股数"
23：-　　卖二报价
24：-　　卖三申请股数"
25：-　　卖三报价
26：-　　卖四申请股数"
27：-　　卖四报价
28：-　　卖五申请股数"
29：-　　卖五报价
30："2008-01-11"，日期；
31："15:05:32"，时间；

与前次计算得来
直接拷贝

20181215 版

概要设计

如果存在【上一条记录】（※１），则进行1、2、3、4、5
如果不存在【上一条记录】（※１），则进行5

!----------

1 你需要用当前记录与上一条记录做对比
2 你需要计算出差值，作为【收盘】【成交量】【成交额】
3 你需要直接采用当前值，作为【开盘】【最低】【最高】【时间】
4 将信息与接收到的信息发送给THINGSdb
5 你需要保留当前记录，作为上一条记录(※２)

!----------

※１　以当日期为名的文件，如果存在，且内容不为空，即为【存在上一条记录】
　　反之，即为【不存在上一条记录】
※２　将当前记录更新以当日期为名的文件

CronTab每秒执行一次Java

ini文件：与Java同地址，主要记录日期文件的地址。
文件位置：
batch/stockOnTimeInfo/年/月/日期为名的文件

参考资料

20181215 版

crontab+shell实现java文件定时运行
https://blog.csdn.net/fengoh/article/details/18791297

linux crontab执行jar简单demo
https://blog.csdn.net/gongzi2311/article/details/54582914

linux利用crontab定时执行java代码（jar）
https://blog.csdn.net/huihuiph/article/details/80263374

19. crontab 定时任务
https://linuxtools-rst.readthedocs.io/zh_CN/latest/tool/crontab.html

20181216 版

详细设计

就是每秒做一次收盘处理。即将一秒间所有的成交量都作为成交价来存储。

这样做其实也会出现误差，但还是会更精准一些（比一天的价格都算到最后成交价上要准取得多）

版本0.1

取得最新信息

取得时间

根据【时间】查看对应的文件是否存在

如果不存在

执行【5】

如果存在

执行【1】【2】【3】【4】【5】

版本0.2

取得股票数据_by字符串

取得临时文件名_by成交时间YYYYMMDD_默认路径

取得指定文件内容_by对象文件全路径

如果【指定文件内容】不存在

执行【5】

如果【指定文件内容】存在

执行【1】【2】【3】【4】【5】

股票实时数据Class

20181216 版

```
// 关键字
String[]    股票关键字 = ['','','','','',''];
Map 股票数据Map;


取得股票数据Map_by字符串()
        将字符串转成【股票数据数组】;
        如果【股票数据数组】与【股票关键字数组】的Size不一致，退出！
        根据【股票关键字数组】循环处理
            // ii 为本次处理番号;
            股票数据Map.put(
                    【股票关键字数组】[ii] ,
                    【股票数据数组】[ii]
                );
        return 股票数据Map;

取得临时文件名_by成交日期YYYYMMDD_默认路径_股票代号
        // 时间默认为 2008-01-11
        YYYY = 取得YYYY_byYYYYMMDD
        MM = 取得MM_byYYYYMMDD
        DD = 取得DD_byYYYYMMDD

        文件Full路径= s默认路径 + YYYY/MM/DD_股票代号.txt

        return 文件Full路径;

取得指定文件内容_by对象文件全路径

        判断文件是否存在
        否          return null;

        判断文件是否存在内容
        否          return  null;

        return 文件内容
```

20181216 版

1 你需要用当前记录与上一条记录做对比
2 你需要计算出差值，作为【收盘】【成交量】【成交额】
3 你需要直接采用当前值，作为【开盘】【最低】【最高】【时间】
4 将信息与接收到的信息发送给THINGSdb
5 你需要保留当前记录，作为上一条记录(※２)

股票实时数据Class

```
// 关键字
String[]    股票关键字 = ['','','','','',''];
Map 前回股票数据Map;
Map 当前股票数据Map;
Map 提交股票数据Map;
String s临时文件名 = null;
String s默认路径 = "/Users/haoyan/Desktop/batch/stockInfo";
取得股票数据Map_by字符串()
取得临时文件名_by成交日期YYYYMMDD_默认路径_股票代号()
取得指定文件内容_by对象文件全路径()
取得前回股票数据Map()
取得当前股票数据Map()
取得当前成交量()
取得当前成交额()
发送数据_byURL_送信数据()
写入前股票数据_by对象文件全路径_写入数据()
```

20181216 版

序号	股票关键字
0	股票名字
1	开盘价
2	昨日收盘价
3	收盘价
4	最高价
5	最低价
6	竞买价
7	竞卖价
8	成交股票数
9	成交金额
10	买一股数
11	买一报价
12	买二股数
13	买二报价
14	买三股数
15	买三报价
16	买四股数
17	买四报价
18	买五股数
19	买五报价
20	卖一股数
21	卖一报价
22	卖二股数
23	卖二报价
24	卖三股数
25	卖三报价
26	卖四股数
27	卖四报价
28	卖五股数
29	卖五报价
30	日期
31	时间

股票实时数据Class

20181216 版

取得前回股票数据Map_byURL()

```
// 每次可能要全取所有股票，或者每十只的取得，否则速度肯定会跟不上
String s股票数据str = 网络取得_byURL
return 取得股票数据Map_by字符串(s股票数据str);
```

取得当前股票数据Map_by前回股票数据Map()

```
s临时文件全路径 = 取得临时文件名_by成交日期YYYYMMDD_默认路径(
                        前回股票数据Map.get(),
                        s默认路径);
String s股票数据str = 取得指定文件内容_by对象文件全路径(s临时文件全路径);
return 取得股票数据Map_by字符串(s股票数据str);
```

取得当前成交量_by前回股票数据Map_当前股票数据Map()

```
前回成交量 = 前回股票数据Map.get(股票关键字[定数_成交量]);
当前成交量 = 当前股票数据Map.get(股票关键字[定数_成交量]);
return 当前股票数据Map.put(成交量, 当前成交量 - 前回成交量);
```

取得当前成交额_by前回股票数据Map_当前股票数据Map()

```
前回成交额 = 前回股票数据Map.get(股票关键字[定数_成交额]);
当前成交额 = 当前股票数据Map.get(股票关键字[定数_成交额]);
return 当前股票数据Map.put(成交额, 当前成交额 - 前回成交额);
```

发送数据_byURL_送信数据()

```
// 这里恐怕要每10次一送信了
```

写入前股票数据_by对象文件全路径List_写入数据ListMap()

```
/**
个股数据Map
    |----- 股票代码Str   // 从URL获取的信息
    |----- 取得信息Str   // 从URL获取的信息
    |----- 发送信息Map
*/

for(个股数据Map：写入数据ListMap){
        股票代号 = 个股数据Map.get(股票关键字[定数_股票代码]);
        对象文件全路径 =
                取得临时文件名_by成交日期YYYYMMDD_默认路径_股票代号（
                        this.s默认路径,
                        股票代码Map.get(股票关键字[定数_股票名字])
                        );
        写入前股票数据_by对象文件全路径_写入数据(
                        对象文件全路径 ,
                        个股数据Map.get(定数_发送信息)
                        );
}
```

写入前股票数据_by对象文件全路径_写入数据()

```
// 文件IO
```

详细设计

20181217 版

Class **股票代码List**

```
取得股票代码文件地址_SH
        return PROPERTY.取得SH股票代码默认路径();
取得股票代码文件地址_SZ
        return PROPERTY.取得SZ股票代码默认路径();
取得代码文件默认文件名_SH
        return PROPERTY.取得SH股票代码默认文件名();
取得代码文件默认文件名_SZ
        return PROPERTY.取得SZ股票代码默认文件名();
做成股票代码List
        取得SH股票代码List();
        取得SZ股票代码List();
取得SH股票代码List
        return 文件IO.取得文件内容_by文件全路径(
                取得股票代码文件地址_SH + "/" + 取得代码文件默认文件名_SH;
                );
取得SZ股票代码List
        return 文件IO.取得文件内容_by文件全路径(
                取得股票代码文件地址_SZ + "/" + 取得代码文件默认文件名_SZ;
                );
```

Class **请求URL**

```
取得每次请求数
        return PROPERTY.取得每次送信数();
取得股票代码List
        股票代码List.做成股票代码List();
做成请求URLList
        PROPERTY . 取得请求URL_head
        for (股票代码 : 股票代码List){

                if ( 计数 < 每次请求数)
                        // 未满足计数。即累积股票代码。
                else{
                        // 满足计数。做成一个URL。存入【请求URLList】。累积清空。计数清空
                        未满足计数_处理();
                }
        };
        return 【请求URLList】;
未满足计数_处理
        已累积股票代码  += 当前本次代码；
满足计数_处理(请求URLList, 计数, 已累积股票代码 )
        URL = 请求URL_head + 已累积股票代码; // 做成URL
        请求URLList. Add(URL ); // 存入【请求URLList】
        计数 = 0; // 清空计数
        已累积股票代码 = “” ； // 清空已累积股票代码
```

Class **实时信息MapList**

20181217 版

```
取得实时信息MapList_by请求URLList() {
        请求URLList = 请求URL . 取得请求URL();
        for (s请求URL : 请求URLList ){
                s实时信息STR = 取得实时信息STR_by请求URL(s请求URL);
                实时信息STR数组[] = s实时信息STR.split(";");
                实时信息MapList = 取得实时信息MapList_by实时信息STR数组(实时信息STR[]);
                全部实时信息MapList.addAll(实时信息MapList);
        }
        retrun 全部实时信息MapList;
}

取得实时信息MapList_by实时信息STR数组(实时信息STR[]){
        for (s实时信息STR : 实时信息STR[]){
                实时信息Map = 取得实时信息Map_by实时信息STR(s实时信息STR){
                实时信息MapList.add(实时信息MapList);
        }
        retrun 实时信息MapList;
}

取得实时信息STR_by请求URL(s请求URL){

        // 参考例子

取得实时信息Map_by实时信息STR(s实时信息STR){

        // 纯体力活
```

Class **前回文件地址**

```
取得年（YYYY）_by实时信息Map
        return Format（实时信息Map. get日期）;
取得月（MM）_by实时信息Map
        return Format（实时信息Map. get日期）;
取得日（DD）_by实时信息Map
        return Format（实时信息Map. get日期）;
取得股票代码_by实时信息Map
        return 实时信息Map. get(股票代码）;

取得前回默认路径
        return PROPERTY . 取得前回文件默认路径();

取得前回文件地址_by实时信息Map
        年（YYYY）= 取得年（YYYY）_by实时信息Map(实时信息Map);
        月（MM） = 取得月（MM）_by实时信息Map(实时信息Map);
        日（DD）= 取得日（DD）_by实时信息Map(实时信息Map);
        股票代码 = 取得股票代码_by实时信息Map(实时信息Map);
        return 前回默认路径 + "/" + 年（YYYY）+"/" +月（MM）+"/" +日(DD) +股票代码 +".txt";
```

Class **前回信息Map**

20181217 版

```
取得前回信息STR_by实时信息Map
        前回文件地址 = 前回文件地址 . 取得前回文件地址_by实时信息Map( 实时信息Map );
        return 取得前回信息STR_by前回文件地址( 前回文件地址);

取得前回信息STR_by前回文件地址
        // 文件IO

取得前回信息Map_by前回信息STR
        前回信息STR =

取得前回信息Map_by实时信息Map
        s前回信息STR = 取得前回信息STR_by实时信息Map （） ；
        return 实时信息MapList . 取得实时信息Map_by实时信息STR(s前回信息STR)；
```

Class **收盘电文**

```
取得送信URL
        return PROPERTY . 取得送信URL_head();

取得每次送信数
        return PROPERTY . 取得每次送信数();

取得收盘电文_实时信息MapList
        for( 实时信息Map : 实时信息MapList){
                o前回信息Map = 前回信息Map . 取得前回信息Map_by实时信息Map(实时信息Map);
                s收盘电文 = 取得收盘电文_by实时信息Map_前回信息Map(实时信息Map, o前回信息Map );
                if ( 计数 < 每次发送数)
                        // 累计【收盘电文】
                else{
                        送信_by收盘电文
                        // 清空【收盘电文】
                }

取得收盘电文_by实时信息Map_前回信息Map
        收盘电文Map. putAll(实时信息Map);
        s成交量 = 实时信息Map.get(成交量) - 前回信息Map.get(成交量);
        s成交额 = 实时信息Map.get(成交额) - 前回信息Map.get(成交额);
        收盘电文Map . Put(定数_成交量, s成交量);
        收盘电文Map . Put(定数_成交额, s成交额);
        return 收盘电文Map . toString();


送信_by收盘电文
        // 这个和请求处理是一样的。
```

Class　　static **PROPERTY**

20181217 版

取得每次送信数
取得每次**请**求数
取得SH股票代**码**默**认**路径
取得SZ股票代**码**默**认**路径
取得前回文件默**认**路径
取得SH股票代**码**默**认**文件名
取得SZ股票代**码**默**认**文件名
取得**请**求URL_head
取得送信URL_head
取得取得Log出力文件全路径

stockInfo.property

每次送信数
每次**请**求数
SH股票代**码**默**认**路径
SZ股票代**码**默**认**路径
前回文件默**认**路径
SH股票代**码**默**认**文件名
SZ股票代**码**默**认**文件名
请求URL_head
送信URL_head
Log出力文件全路径

20181218版

```
import java.util.List;

public class 请求URL对象 {
	private String 取得每次请求数() {
		return PROPERTY.取得每次送信数();
	}

	private List<String> 做成请求URLList() {
		PROPERTY.取得请求URL_head();
		int 计数 = 0;
		List 请求URLList = null;
		String s已累积股票代码 = null;
		String s当前股票代码 = null;
		for (String s股票代码 : 股票代码List对象.做成股票代码List()) {

			int 每次请求数 = 0;
			if (计数 < 每次请求数)
				// 未满足计数。即累积股票代码。
				未满足计数_处理(s已累积股票代码, s当前股票代码);
			else {
				// 满足计数。做成一个URL。存入【请求URLList】。累积清空。计数清空
				满足计数_处理(请求URLList, 计数, s已累积股票代码);
			}
		}
		return 请求URLList;
	}

	private void 未满足计数_处理(String s已累积股票代码, String s当前股票代码) {
		s已累积股票代码 += s当前股票代码;
	}

	private void 满足计数_处理(List 请求URLList, long 计数, String s已累积股票代码) {
		String sURL = PROPERTY.取得请求URL_head() + s已累积股票代码; // 做成URL
		请求URLList.add(sURL); // 存入【请求URLList】
		计数 = 0; // 清空计数
		s已累积股票代码 = ""; // 清空已累积股票代码
	}
}
```

20181218版

```
import java.util.ArrayList;
import java.util.List;
import java.util.Map;

public class 实时信息MapList对象 {

	private List<Map>  取得实时信息MapList_by请求URLList(List<String> 请求URLList) {
		List<Map> 全部实时信息MapList = new ArrayList();
		// 请求URLList = 请求URL对象.取得请求URL();
		for (String s请求URL : 请求URLList ){
			String s实时信息STR = 取得实时信息STR_by请求URL(s请求URL);
			String 实时信息STR数组[] = s实时信息STR.split(";");
			List<Map> 实时信息MapList = 取得实时信息MapList_by实时信息STR数组(实时信息STR数组);
			全部实时信息MapList.addAll(实时信息MapList);
		}
		return 全部实时信息MapList;
	}

	private List<Map> 取得实时信息MapList_by实时信息STR数组(String[] 实时信息STR数组){
		List<Map> 实时信息MapList = new ArrayList();
		for (String s实时信息STR : 实时信息STR数组){
			Map 实时信息Map = 取得实时信息Map_by实时信息STR(s实时信息STR);

			实时信息MapList.add(实时信息Map);
		}
		return 实时信息MapList;
	}

	private String 取得实时信息STR_by请求URL(String s请求URL){

		// 参考例子
		return null;
	}

	public Map 取得实时信息Map_by实时信息STR(String s实时信息STR){

		// 纯体力活
		return  null;

	}

}
```

20181218版

```
import java.util.List;
import java.util.Map;

public class 收盘电文 {
	private static final String 定数_成交额 = "成交额";
	private Map 收盘电文Map;
	private static final String 定数_成交量 = "成交量";

	private String 取得送信URL() {
		return PROPERTY.取得送信URL_head();
	}

	private String 取得每次送信数() {
		return PROPERTY.取得每次送信数();
	}

	private void 取得收盘电文_实时信息MapList(List<Map> 实时信息MapList) {
		前回信息Map对象 o前回信息Map对象 = new 前回信息Map对象();
		for (Map 实时信息Map : 实时信息MapList) {
			Map o前回信息Map = o前回信息Map对象.取得前回信息Map_by实时信息Map(实时信息Map);
			String s收盘电文 = 取得收盘电文_by实时信息Map_前回信息Map(实时信息Map, o前回信息Map);
			int 计数 = 0;
			int 每次发送数 = 0;
			if (计数 < 每次发送数) {
				// 累计【收盘电文】
			} else {
				送信_by收盘电文();
				// 清空【收盘电文】
			}
		}
	}

	private String 取得收盘电文_by实时信息Map_前回信息Map(Map 实时信息Map, Map 前回信息Map) {
		收盘电文Map.putAll(实时信息Map);
		long l成交量 = (Long) 实时信息Map.get(定数_成交量) - (Long) 前回信息Map.get(定数_成交量);
		long l成交额 = (Long) 实时信息Map.get(定数_成交额) - (Long) 前回信息Map.get(定数_成交额);
		收盘电文Map.put(定数_成交量, l成交量 + "");
		收盘电文Map.put(定数_成交额, l成交额 + "");
		return 收盘电文Map.toString();
	}

	private void 送信_by收盘电文() {
		// 这个和请求处理是一样的。
	}
}
```

20181218版

```
import java.util.Map;

public class 前回文件地址对象 {

	private static final Object 定数_股票代码 = "股票代码";

	private String 取得年YYYY_by实时信息Map(Map 实时信息Map) {
		//return Format(实时信息Map. get日期);
		return null;
	}

	private String 取得月MM_by实时信息Map(Map 实时信息Map) {
		//return Format(实时信息Map. get日期);
		return null;
	}

	private String 取得日DD_by实时信息Map(Map 实时信息Map) {
		//return Format(实时信息Map. get日期);
		return null;
	}

	private String 取得股票代码_by实时信息Map(Map 实时信息Map) {
		return (String) 实时信息Map.get(定数_股票代码);
	}

	public String 取得前回文件地址_by实时信息Map(Map 实时信息Map) {
		String 年_YYYY = 取得年YYYY_by实时信息Map(实时信息Map);
		String 月_MM = 取得月MM_by实时信息Map(实时信息Map);
		String 日_DD = 取得日DD_by实时信息Map(实时信息Map);
		String 股票代码 = 取得股票代码_by实时信息Map(实时信息Map);
		return PROPERTY.取得前回文件默认路径() + "/" + 年_YYYY + "/" + 月_MM + "/" + 日_DD + 股票代码 + ".txt";
	}
}
```

```
import java.util.Map;

public class 前回信息Map对象 {

	private String 取得前回信息STR_by前回文件地址(String 前回文件地址) {
		// 文件IO
		return 文件IO.取得文件内容_by文件全路径(前回文件地址).get(0);
	}

	public Map 取得前回信息Map_by实时信息Map(Map 实时信息Map) {
		String s前回信息STR = 取得前回信息STR_by实时信息Map(实时信息Map);
		实时信息MapList对象 o实时信息MapList对象 = new 实时信息MapList对象();
		return o实时信息MapList对象.取得实时信息Map_by实时信息STR(s前回信息STR);
	}

	private String 取得前回信息STR_by实时信息Map(Map 实时信息Map) {
		前回文件地址对象 o前回文件地址对象 = new 前回文件地址对象();
		String s前回文件地址 = o前回文件地址对象.取得前回文件地址_by实时信息Map(实时信息Map);
		return 取得前回信息STR_by前回文件地址(s前回文件地址);
	}
}
```

20181218版

```
import java.util.ArrayList;
import java.util.List;

public class 股票代码List对象 {
	private static String 取得股票代码文件地址_SH() {
		return PROPERTY.取得SH股票代码默认路径();
	}

	private static String 取得股票代码文件地址_SZ() {
		return PROPERTY.取得SZ股票代码默认路径();
	}

	private static String 取得代码文件默认文件名_SH() {
		return PROPERTY.取得SH股票代码默认文件名();
	}

	private static String 取得代码文件默认文件名_SZ() {
		return PROPERTY.取得SZ股票代码默认文件名();
	}

	public static List<String> 做成股票代码List() {
		List<String> 股票代码List = new ArrayList();
		股票代码List.addAll(取得SH股票代码List());
		股票代码List.addAll(取得SZ股票代码List());
		return 股票代码List;
	}

	private static List<String> 取得SH股票代码List() {
		return 文件IO.取得文件内容_by文件全路径(
				取得股票代码文件地址_SH() + "/" + 取得代码文件默认文件名_SH());
	}

	private static List<String> 取得SZ股票代码List() {
		return 文件IO.取得文件内容_by文件全路径(
				取得股票代码文件地址_SZ() + "/" + 取得代码文件默认文件名_SZ());
	}
}
```