|  |
| --- |
| Luckers股票数据分析系统 |
| **软件设计文档** |
| 版本号：2.1.3 |

|  |
| --- |
| 马张弛，孟鑫，刘兴，朱宇翔  2016-4-12 |

# 引言

## 1.1 目的

本文档简要描述了Luckers股票分析软件的架构设计，主要描述逻辑层与数据层接口规范以及关键部位的实现设计。

## 1.2 参考资料

《Luckers需求规格说明文档》

# 2. 中层设计

## 2.1 模块划分与职责

本软件采用分层模式，分为界面层、逻辑层与数据层。本文档仅记录逻辑层与数据层的设计情况。

### 2.1.1 逻辑层模块划分

|  |  |
| --- | --- |
| 模块名 | 职责 |
| LogicServer | 提供逻辑层对外的所有接口 |
| DetailService | 提供股票信息的获取与筛选功能 |
| FavouriteService | 提供本地收藏夹功能，包括增删查 |
| MarketService | 提供大盘信息的获取功能 |
| IndustryService | 提供股票行业获取、评分与对比功能 |
| KGraphService | 提供K线图的绘制功能 |
| ListService | 提供列表化传递股票数据的功能 |

### 2.1.2 数据层模块划分

|  |  |
| --- | --- |
| 模块名 | 职责 |
| NetWork | 提供网络访问的功能 |
| LocalFileSystem | 提供本地数据存储与访问功能 |

## 2.2 逻辑层模块接口说明

### 2.2.1 LogicServer模块

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| LogicController.getService() | 语法 | Public Service getService（） |
| 前置条件 | 无 |
| 后置条件 | 返回逻辑层相应的接口，这里的Service是接口的泛指名 |
| MyCalendar.getTime（） | 语法 | Public String getTime（） |
| 前置条件 | 无 |
| 后置条件 | 根据需要返回各种需要的日期；比如系统当前日期、最近的交易日日期等；详细可参考该类注释代码 |
| ExceptionReporter.getException() | 语法 | Public String getException() |
| 前置条件 | 系统运行发生异常时 |
| 后置条件 | 返回异常信息 |

### 2.2.2 DetailService模块

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| StockDetailService.getDetails（） | 语法 | Public terator<StockInformVO> getDetails(String Id,String start,String end); |
| 前置条件 | 网路正常，输入的id与日期格式正确 |
| 后置条件 | 返回要求代号股票在对应日期中的数据  当不选择开始结束日期时，只返回最近一个月的数据 |
| StockDetailService.getOneDay（） | 语法 | Public StockInformVO getOneDay（String id，String date） |
| 前置条件 | 网路正常，输入的id与日期格式正确 |
| 后置条件 | 返回要求id的股票在指定日期的数据 |
| DataScreenService.getChosen() | 语法 | public Iterator<StockInformVO> getChosen(String id, String startDate, String endDate,  ArrayList<StockScreenVO> voList); |
| 前置条件 | 网路正常，输入的id与日期格式正确 |
| 后置条件 | 根据股票id与日期以及StockScreenVO中的筛选条件返回筛选结果 |
| 需要的接口 | | |
| 服务名 | 服务 | |
| StockDataGetter.getStockById | 提供网络访问股票数据的功能 | |
| CacheDataGetter.get | 提供本地缓存数据的访问 | |
| StockDataGetter.getStockName | 提供网络访问股票名字的功能 | |

### 2.2.3 FavouriteService模块

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| FavouriteService.getList | 语法 | Public ArrayList<String> getList() |
| 前置条件 | 无 |
| 后置条件 | 返回本地收藏的所有股票的id |
| FavouriteService.getContent | 语法 | public Iterator<StockInformVO>  GetContent(); |
| 前置条件 | 网路正常 |
| 后置条件 | 返回收藏夹中所有股票最新一天的数据 |
| FavouriteService.add | 语法 | Public boolean add(String id) |
| 前置条件 | 网路正常，输入的id正确 |
| 后置条件 | 将目标id的股票添加到收藏夹中 |
| FavouriteService.remove | 语法 | Public boolean remove(String id) |
| 前置条件 | 网路正常，输入的id正确 |
| 后置条件 | 将目标id的股票从收藏夹中 |
| 需要的接口 | | |
| 服务名 | 服务 | |
| StockDataGetter.getStockById | 提供网络访问股票数据的功能 | |
| StringStorageService.getContent | 提供本地持久化数据的访问 | |

### 2.2.4 MarketService模块

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| MarketDetailService.getDetails | 语法 | public MarketDetailVO getDetails(String id,String start,String end); |
| 前置条件 | 网络正常，输入的大盘id与日期正确 |
| 后置条件 | 返回起始与结束日期间指定大盘的数据 |
| MarketInformService.getItems() | 语法 | public MarketInformVO getItems(String id); |
| 前置条件 | 网路正常 |
| 后置条件 | 返回指定大盘最近一天的数据 |
| 需要的接口 | | |
| 服务名 | 服务 | |
| StockDataGetter.getBench  MarketByID | 提供网络访问大盘的功能 | |

### 2.2.5 IndustryService模块

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| IndustryInformService  .getIndustryList | 语法 | public Iterator<IndustryVO> getIndustryList(); |
| 前置条件 | 网络正常 |
| 后置条件 | 返回本地收藏的所有股票所能划分出的行业列表 |
| IndustryInformService  .getIndustryByStock | 语法 | public String getIndustryByStock(String id); |
| 前置条件 | 网路正常 |
| 后置条件 | 根据股票id获得其所在的行业 |
| IndustryInformService  .getIndustryScore | 语法 | public IndustryScoreVO getIndustryScore(String name,String start,String end); |
| 前置条件 | 网路正常，输入的行业名与日期正确 |
| 后置条件 | 获得该行业的得分情况 |
| 需要的接口 | | |
| 服务名 | 服务 | |
| FavouriteService.getList | 获得收藏夹中的股票名单 | |
| StockDataGetter.getIndustry | 提供行业信息的网络访问接口 | |
| IndustryScoreCalculator.cal | 提供行业得分计算的功能 | |

### 2.2.6 KGraphService模块

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| KGraphService.getDayK | 语法 | public Iterator<KGraphVO> getDayK(String id,String start,String end); |
| 前置条件 | 网络正常，股票id和日期输入正确 |
| 后置条件 | 返回目标股票在指定日期间的日K图数据 |
| KGraphService.getWeekK | 语法 | public Iterator<KGraphVO> getWeekK(String id,int number); |
| 前置条件 | 网路正常，股票id输入正确 |
| 后置条件 | 根据股票id返回指定周数的周K图 |
| KGraphService.getMonthK | 语法 | public Iterator<KGraphVO> getMOnthK(String id,int number); |
| 前置条件 | 网路正常，股票id输入正确 |
| 后置条件 | 根据股票id返回指定月数的月K图 |
| 需要的接口 | | |
| 服务名 | 服务 | |
| StockDetailService.getDetail | 获得股票信息 | |

### 2.2.7 ListService模块

|  |  |  |
| --- | --- | --- |
| 提供的接口 | | |
| ListDelayService.getItems | 语法 | public Iterator<StockInformVO> getItems(String exchanger,int num); |
| 前置条件 | 网络正常，大盘日期输入正确 |
| 后置条件 | 返回指定数目的数据 |
| KGraphService.getIdList | 语法 | public Iterator<String> getIdList(String exchanger); |
| 前置条件 | 网路正常，大盘id输入正确 |
| 后置条件 | 获得当天的股票id列表 |
| 需要的接口 | | |
| 服务名 | 服务 | |
| StockDataService.getStockById | 根据id获得股票信息 | |

## 2.3 数据层模块接口说明

### 2.3.1 Network模块

此处直接上代码段（不是我懒主要是作者注释写得太好了）

以下接口均在StockDataGetter接口中

/\*\*

\* **@param** year 年份，可以为null，将返回默认结果（我也不知道默认结果是啥，下同）

\* **@param** exchange 交易所，只有三个可选值:StockDataGetter.SZ（代表深圳交易所）

\* ，StockDataGetter.SH（代表上海交易所） 和 null，如果输入其他值

\* ，将返回未知结果或抛出未知异常，输入null将返回默认结果

\* **@return** 返回在指定条件下的股票代码名单

\* **@throws** IOException 代表从服务器上读取此列表失败

\* **@throws** ConnectionFailure

\*/

**public** ArrayList<String> getStockIDList(String year, String exchange) **throws** IOException, ConnectionFailure;

/\*\*

\* **@param** id 比如：sh600000 不能为null

\* **@param** startDay 起始时间，格式'YYYY-mm-dd'，可以为null（起始时间和结束时间不同时为null时，我也不知道会返回什么）

\* **@param** endDay 结束时间，格式'YYYY-mm-dd'，可以为null（起始时间和结束时间不同时为null时，我也不知道会返回什么）

\* **@param** fields 指定数据字段（可以为null，为null时返回全部交易数据）

\* 格式： name1+name2+name3（可以有多个），具体字段请参考为StockDataPO.STOCK\_FIELDS\_TYPE

\* ：open: 开盘价，high: 最高价

\* ，low: 最低价，close: 收盘价，adj\_price: 后复权价，volume: 成交量，turnover: 换手率

\* ，pe: 市盈率，pb: 市净率

\* **@return** 返回指定股票代码的股票交易数据，默认返回过去一个月的全部交易数据

\* **@throws** IOException 代表从服务器上读取此股票失败

\* PS by MZC:该方法最快只能获得昨天的数据，且返回时不包括结束日期的数据，想要某一天的数据

\* 起始日期选择当天，结束日期选择后一天

\* 获得不了当天的数据真实脑残....

\* **@throws** ConnectionFailure \*/

**public** StockPO getStockByID(String id,String startDay, String endDay,String fields) **throws** IOException, ConnectionFailure;

/\*\*

\* **@param** id 比如：hs300，也可以调用此接口的静态常量 不能为null

\* **@param** startDay 起始时间，格式'YYYY-mm-dd'，可以为null（起始时间和结束时间不同时为null时，我也不知道会返回什么）

\* **@param** endDay 结束时间，格式'YYYY-mm-dd'，可以为null（起始时间和结束时间不同时为null时，我也不知道会返回什么）

\* **@param** fields 指定数据字段（可以为null，为null时返回全部交易数据）

\* 格式： name1+name2+name3（可以有多个），具体字段请参考StockDataPO.BENCHMARK\_FIELDS\_TYPE

\* **@return** 返回指定大盘指数代码的交易数据，默认返回一年内的所有数据（经过测试得出的结论）

\* **@throws** IOException 代表从服务器上读取此股票失败

\* **@throws** ConnectionFailure

\*/

**public** StockPO getBenchmarkByID(String id,String startDay, String endDay,String fields) **throws** IOException, ConnectionFailure;

/\*\*

\* 根据输入的股票去读取其对应的名字

\* Copyright@ 新浪财经，如侵必删，详请联系@author zzzzzyx@outlook.com

\* **@param** stockCodeList

\* **@return** 返回的结果列表与输入的列表按顺序一一对应，如果列表某项股票无法读取，对应的一项值为null

\* **@throws** IOException

\* **@throws** ConnectionFailure

\*/

**public** ArrayList<String> getStockName(ArrayList<String> stockCodeList) **throws** IOException, ConnectionFailure;

/\*\*

\* 根据股票代码从外网读取其行业名，如输入为'sh600485'（即信威集团）, 将返回'通信设备'

\* Copyright@ 网易财经，如侵必删，详请联系@author zzzzzyx@outlook.com

\* **@param** stockCode 股票代码

\* **@return** 行业名称

\* **@throws** IOException

\* **@throws** ConnectionFailure

\*/

**public** String getIndustryName(String stockCode) **throws** IOException, ConnectionFailure;

### 2.3.2 LocalFileSystem模块

以下方法均在StringStorageService接口中

/\*\*

\* 基于此类的deleteAllAndWrite方法编写，并不会提高效率，增加大量内容请尽量使用deleteAllAndWrite方法以提高效率

\* 注意：content的内容不得包含SPECIAL\_SPLITER中的内容，否则会发生解析文件内容异常

\* **@param** addContent

\* **@throws** IOException

\*/

**public** **void** addString(String addContent) **throws** IOException;

/\*\*

\* 基于此类的deleteAllAndWrite方法编写，并不会提高效率

\* **@param** deleteContent

\* **@throws** IOException

\*/

**public** **void** deleteString(String deleteContent) **throws** IOException;

/\*\*

\* 获取存储在本地的内容

\* **@return**

\* **@throws** IOException

\*/

**public** ArrayList<String> getContent() **throws** IOException;

/\*\*

\* 此方法将存储所有的ArrayList中的String

\* 注意：content的内容不得包含SPECIAL\_SPLITER中的内容，否则会发生解析文件内容异常

\* **@param** content

\* **@throws** IOException

\*/

**public** **void** deleteAllAndWrite(ArrayList<String> content) **throws** IOException;

# 3.细节设计描述

## 3.1 逻辑层设计描述

### 3.1.1 总体描述

此次逻辑层的设计想对简单，但对算法的效率提出了比较高的要求。在返回大量的数据时尽量采用迭代器模式；在异常处理上使用观察者模式；使用内存缓存来尽可能地提升数据访问速度。

### 3.1.2 DetailService模块实现描述

该模块是整个软件的核心，所有对股票数据的访问几乎都由该模块提供。为了实现高速访问，在程序运行开始时就开始下载可能会使用到的数据。访问数据时，先访问内存查找目标数据是否存在，如果不存在再访问网络。内存缓存功能由Cache模块实现。

Cache模块构成如下：

|  |  |
| --- | --- |
| 类名 | 职责 |
| CacheDataGetter | 提供内存缓存访问的接口 |
| CacheRecord | 辅助类，记录已经下载的数据 |
| StockGetThread | 辅助类，进行下载的线程 |
| ThreadRecorder | 辅助类，记录各个线程的下载情况 |

## 3.2 数据层设计描述

### 3.2.1 总体描述

数据层处理网络访问和本地存储，此处为了提升效率也使用了缓存功能。部分数据下载下来后会持久化保存在本地，下次访问时会优先访问本地数据。实现该功能的是LocalBufferSystem模块。详细的实现不再赘述。