## 逻辑层

可考虑分为：

显示（显示大盘指数、显示股票列表）

筛选（筛选股票、搜索股票）

查询（查看详细信息、查看大盘指数）

关注（关注、取消关注）

历史（关注历史）

以上都是建立在已获得数据的基础上的，漏掉了获取数据的模块

筛选股票信息似乎无意义，可以添加根据时间显示变化的功能（迭代二）

### 5.查询大盘指数（InquireMarketIndexbl）

1. 模块概述

查询大盘指数模块提供按天查询大盘指数服务，因系统默认显示当天大盘指数（开盘前显示前一天数据），同时可以查询其它日期数据，故提供两个接口，分别提供当天数据及其它选定日期数据。

1. 整体结构
2. 模块内部类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的接口（供接口） | | | |
| InquireMarketIndex  Service.getIndex | 语法 | public IndexPO getIndex(String code) | |
| 前置条件 | 已选择一个大盘指数 | |
| 后置条件 | 返回当天大盘指数 | |
| InquireMarketIndex  Service.getIndex | 语法 | public IndexPO getIndex(String code, Date date) | |
| 前置条件 | 已选择一个大盘指数，且已修改显示日期（默认为当天） | |
| 后置条件 | 返回指定日期的大盘指数 | |
| 需要的服务（需接口） | | | |
| 服务名 | | | 服务 |
| IndexDataService.find(String code) | | | 根据代码返回当天大盘指数信息 |
| IndexDataService.find(String code, Date date) | | | 根据代码返回指定日期大盘指数信息 |

1. 业务逻辑层的动态模型
2. 业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

### 8.查询关注历史（inquirefollowhistorybl）

1. 模块概述

系统记录用户所有关注股票的历史，取消关注后自动添加到历史关注；用户也可以清空当前关注和历史关注。本模块所提供服务对应于上述需求。

1. 整体结构
2. 模块内部类的接口规范

|  |  |  |  |
| --- | --- | --- | --- |
| 提供的接口（供接口） | | | |
| FollowHistoryService.  getFollow | 语法 | | public Iterator<StockPO> getFollow() |
| 前置条件 | | 启动一个查询关注历史回合 |
| 后置条件 | | 返回当前关注列表的迭代器 |
| FollowHistoryService.  GetFollowed | 语法 | | public Iterator<StockPO> getFollowed() |
| 前置条件 | | 启动一个查询关注历史回合 |
| 后置条件 | | 返回历史关注列表（不包括当前关注列表）的迭代器 |
| FollowHistoryService.  getFollowNum | 语法 | | public int getFollowNum() |
| 前置条件 | | 启动一个查询关注历史回合 |
| 后置条件 | | 返回当前关注股票数量 |
| FollowHistoryService.  getFollowedNum | 语法 | | public int getFollowedNum() |
| 前置条件 | | 启动一个查询关注历史回合 |
| 后置条件 | | 返回历史关注股票数量 |
| FollowHistoryService.  clearFollow | 语法 | | public boolean clearFollow() |
| 前置条件 | | 启动一个查询关注历史回合 |
| 后置条件 | | 取消当前所有关注，并将关注股票全部添加到历史关注列表 |
| FollowHistoryService.  clearFollowed | 语法 | | public boolean clearFollowed() |
| 前置条件 | | 启动一个查询关注历史回合 |
| 后置条件 | | 清空历史关注列表 |
| FollowHistoryService.  clearAll | 语法 | | public boolean clearAll() |
| 前置条件 | | 启动一个查询关注历史回合 |
| 后置条件 | | 取消所有关注，并清空关注列表 |
| 需要的服务（需接口） | | | |
| 服务名 | | 服务 | |
| FollowHistoryDataService.getFollow | | 返回当前关注列表 | |
| FollowHistoryDataService.getFollowed | | 返回历史关注列表 | |
| FollowHistoryDataService.clearFollow | | 取消当前所有关注，并将所有关注股票添加到历史关注列表 | |
| FollowHistoryDataService.clearFollowed | | 清空历史关注列表 | |
| FollowHistoryDataService.clearAll | | 取消所有关注，并清空关注列表 | |

1. 业务逻辑层的动态模型
2. 业务逻辑层的设计原理

利用委托式控制风格，每个界面需要访问的业务逻辑由各自的控制器委托给不同的领域对象。

## 数据层

### 5.查询大盘指数（InquireMarketIndexData）

1. 模块概述
2. 整体结构
3. 模块内部类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| IndexDataService.find | 语法 | public IndexPO find(String code) |
| 前置条件 | 无 |
| 后置条件 | 查找并返回当天大盘指数（IndexPO） |
| IndexDataService.find | 语法 | public IndexPO find(String code, Date date) |
| 前置条件 | 无 |
| 后置条件 | 查找并返回指定日期的大盘指数 |

1. 业务逻辑层的动态模型
2. 业务逻辑层的设计原理

### 8.查询关注历史（InquirefollowhistoryData）

1. 模块概述
2. 整体结构
3. 模块内部类的接口规范

|  |  |  |
| --- | --- | --- |
| 提供的接口（供接口） | | |
| FollowHistoryDataService.getFollow | 语法 | public List<StockPO> getFollow() |
| 前置条件 | 无 |
| 后置条件 | 返回当前关注列表 |
| FollowHistoryDataService.getFollowed | 语法 | public List<StockPO> getFollowed() |
| 前置条件 | 无 |
| 后置条件 | 返回历史关注列表（不包括当前关注列表） |
| FollowHistoryDataService.clearFollow | 语法 | public boolean clearFollow() |
| 前置条件 | 无 |
| 后置条件 | 取消当前所有关注，并将关注股票全部添加到历史关注列表 |
| FollowHistoryDataService.clearFollowed | 语法 | public boolean clearFollowed() |
| 前置条件 | 无 |
| 后置条件 | 清空历史关注列表 |
| FollowHistoryDataService.clearAll | 语法 | public boolean clearAll() |
| 前置条件 | 无 |
| 后置条件 | 取消所有关注，并清空关注列表 |

1. 业务逻辑层的动态模型
2. 业务逻辑层的设计原理