日程管理软件

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| 2014.7.4 | 1.0 | 最初版本 | 张子辰 |
| 2014.7.18 | 2.0 |  | 张子辰 |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 2

1.1 目的 2

1.2 范围 2

1.3 定义、首字母缩写词和缩略语 2

1.4 参考资料 2

1.5 概述 2

2. 构架表示方式 2

3. 构架目标和约束 2

4. 用例视图 2

4.1 用例实现 2

5. 逻辑视图 2

5.1 概述 2

5.2 在构架方面具有重要意义的设计包 2

6. 进程视图 2

7. 部署视图 2

8. 实施视图 2

8.1 概述 2

8.2 层 2

9. 数据视图（可选） 2

10. 大小和性能 2

11. 质量 2

# 简介

## 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面作出的重要决策。

## 参考资料

《需求规约文档》

# 构架表示方式

包括用例视图、逻辑视图、进程视图、部署视图、实施视图、和数据视图。

用例视图描述了该软件的各个用例以及几个比较突出的具体实现。逻辑视图描述了各个类的主要组成以及它们之间的关系。进程视图描述了该软件运行时内部各个进程的交互方式。部署视图描述了该软件运行的环境及预备工作。实施视图描述了该软件的分层情况和各个层之间的交互方式。数据视图描述了用于该软件持久化存储的数据库结构。

# 构架目标和约束

开发语言：Java。

PC端开发工具：NetBeans。

安卓端开发工具：AndroidStudio。

PC端窗体框架：Swing。

安卓端窗体框架：原生框架。

设计模式：面向对象的UML设计模式。

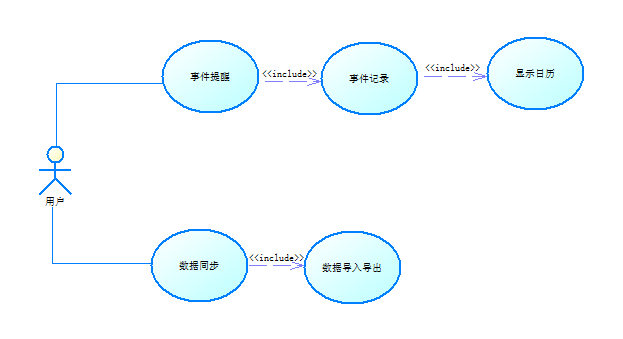
建模工具：PowerDesign。

数据库：Sqlite。

团队结构：单人项目。

# 用例视图

## 概述



## 用例实现

用户打开软件时，软件首先获取当前的年份和月份显示日历。用户可以点击年份及月份输入框旁边的小箭头调整日历的年月。用户可以通过点击日历上的某个日期格子来显示当天的计划，也可以通过点击计划表格的某一行来修改这个计划，通过点击添加或删除按钮来删除这个计划。

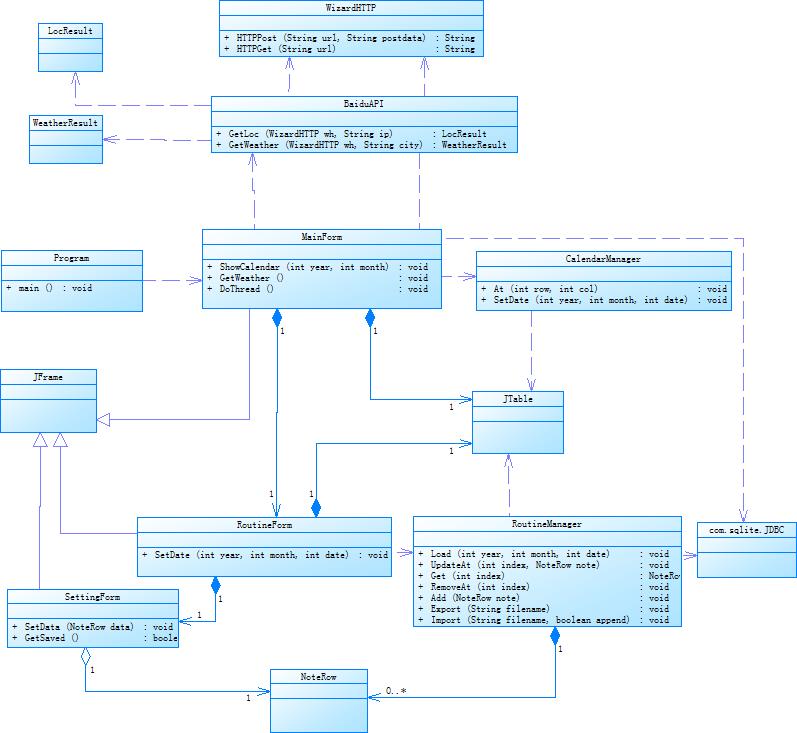
后台线程循环读取待提醒的计划安排，并在用户设定的时间上发出提醒。

用户可以手动点击按钮以导出导入全部数据，并在移动端或PC段共享。

# 逻辑视图

## 概述

主要类的UML图：



包结构：

Routine:

Program, MainForm, RoutineForm, SettingForm

Routine.Utility:

CalendarManager, RoutineManager, WizardHTTP, NoteRow, StreamWriter, StreamReader

Routine.BaiduAPI:

BaiduAPI, LocResult, WeatherResult

## 在构架方面具有重要意义的设计包

Program：存放入口函数main，该函数中实例化主窗体。

MainForm：绘制并显示主窗体。

RoutineForm：绘制并显示某一天的日程计划窗体。

SettingForm：绘制并显示日程计划设置窗体。

CalenderManager：根据输入的年和月生成日历，并把数据传给表格控件。

RoutineManager：加载指定年和月的数据，添加、删除和更新数据，并与数据库同步。

NoteRow：存放一条计划，便于传参。

# 进程视图

该软件只有一个进程，并分为两个线程。主线程用于显示界面，获取及显示数据。另外一个线程用于获取并显示提醒信息。

启动时有一个短期运行的线程获取天气。

# 部署视图

本程序为Win32系统上的PC程序，无需安装即可直接使用，运行前需要安装JRE1.7或以上版本。获取天气功能需要连接互联网，其他功能不需要。

# 实施视图

## 概述

整个程序分为三层：数据持久化层、显示层和控制层。

## 数据层

数据持久化层负责读写或封装数据库中的数据，包括sqlite的驱动及NoteRow。

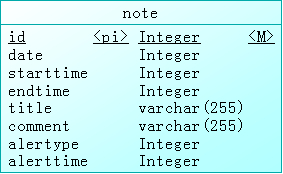
## 显示层

显示层负责在屏幕上绘制并显示窗体，包括所有继承自JFrame的窗体，以及各种控件类。

## 控制层

控制层衔接数据和显示层，负责从数据层得到数据，并按照显示层的要求传给显示层，包括CalendarManager、RoutineManager。

# 数据视图



# 大小和性能

对于用户每个点击操作，软件的响应时间不超过3秒。

# 质量

采用面向对象方法合理地设计系统的结构以保证较高的可维护性。