

# 系統設計書

## System Design Document

## 目錄

1. 修訂版本記錄 .....	4
2. 簡介 .....	5
2.1 目標 .....	5
2.2 產品概觀 .....	5
2.3 範圍 .....	5
3. 開發環境 .....	6
3.1 Development tools .....	6
3.2 Development standards .....	6
3.3 System environment .....	6
4. 系統架構 .....	7
4.1 Architecture diagram .....	7
4.2 產品整合順序 .....	7
5. 系統設計 .....	8
5.1 使用者介面描述 .....	8
5.1.1 畫面 .....	8
5.1.2 物件及動作 .....	9
5.2 主要介面 設計規格 .....	9
5.2.1 主要介面 介面描述 .....	9
5.2.2 主要介面 運作說明 .....	10
5.2.3 互動圖 .....	11
5.3 資料庫 設計規格 .....	11
5.3.1 資料庫 介面描述 .....	11
5.3.2 資料庫 運作說明 .....	11
5.3.3 互動圖 .....	12
5.4 月曆 設計規格 .....	12
5.4.1 月曆 介面描述 .....	12
5.4.2 月曆 運作說明 .....	13
5.4.3 互動圖 .....	13
5.5 資料處理 設計規格 .....	13
5.5.1 資料處理介面描述 .....	13
5.5.2 資料處理 運作說明 .....	13
5.5.3 互動圖 .....	14
5.6 資料顯示 設計規格 .....	15
5.6.1 資料顯示 介面描述 .....	15
5.6.2 資料顯示運作說明 .....	15
5.6.3 互動圖 .....	15
5.7 音樂撥放器 設計規格 .....	16
5.7.1 音樂撥放器介面描述 .....	16
5.7.2 音樂撥放器 運作說明 .....	16
5.7.3 互動圖 .....	16
5.8 元件整合條件 .....	16
5.9 設計規則 .....	17
6. 非功能設計 .....	17

6.1 效能設計 .....	17
6.2 安全性設計 .....	17
6.3 可靠性設計 .....	17
6.4 可維護性設計 .....	17
6.5 邏輯資料庫設計 .....	17
<b>7. 資料設計</b> .....	<b>18</b>
7.1 內部軟體資料結構 .....	18
7.2 全域資料結構 .....	19
7.3 暫存性資料結構 .....	19
7.4 資料庫說明 .....	20

## 1. 修訂版本記錄

版本	修改日期	說明	修改者
1	2018/05/16	未完整第一版	賴俊霖
2	2018/05/19	主要設計討論完成。非功能性設計完成	許新源
3	2018/05/20	主要元件規劃撰寫完成	楊軒宇
4	2018/05/21	補上資料庫設計，系統設計書完成	鄭紹雄
5	2018/05/22	排版完成	鄭紹雄

## 2. 簡介

---

### 2.1 目標

主要是為私人公司所設計的行事曆，能夠讓員工在同一臺電腦上記錄自己的行程，方便安排行程、記憶公事。

管理者可以透過查詢得知員工於上班時間的行程。

### 2.2 產品概觀

我們的好行事曆提供月曆查詢，行程與帳號的新增、更新及刪除。

也可以依照類別或日期搜尋行程、並提醒當日行程。

並隨時撥放著輕鬆愉悅♂的音樂，當然，你也可以停止它 d(・▽・)b。

### 2.3 範圍

適用於無連接上網的電腦，這樣可以保證員工的行程不會因為網路而外洩出去

### 3. 開發環境

---

#### 3.1 Development tools

所需軟體: Eclipse、TortoiseSVN、StarUML

所需工具: 電腦、筆記型電腦

#### 3.2 Development standards

1. 必須要按照 MVC 架構所訂好的結構進行開發，除非有特別功能可以獨立。

2. 函式左大括號要在小括號右邊

Ex. void XXX(){

}

3. 註解必須在函式上面。

//我是函式

Void XXX(){

}

4. 函式、變數命名必須為小駝峰形式

Ex. String firstName、void secondFunction()。

#### 3.3 System environment

作業系統: windows

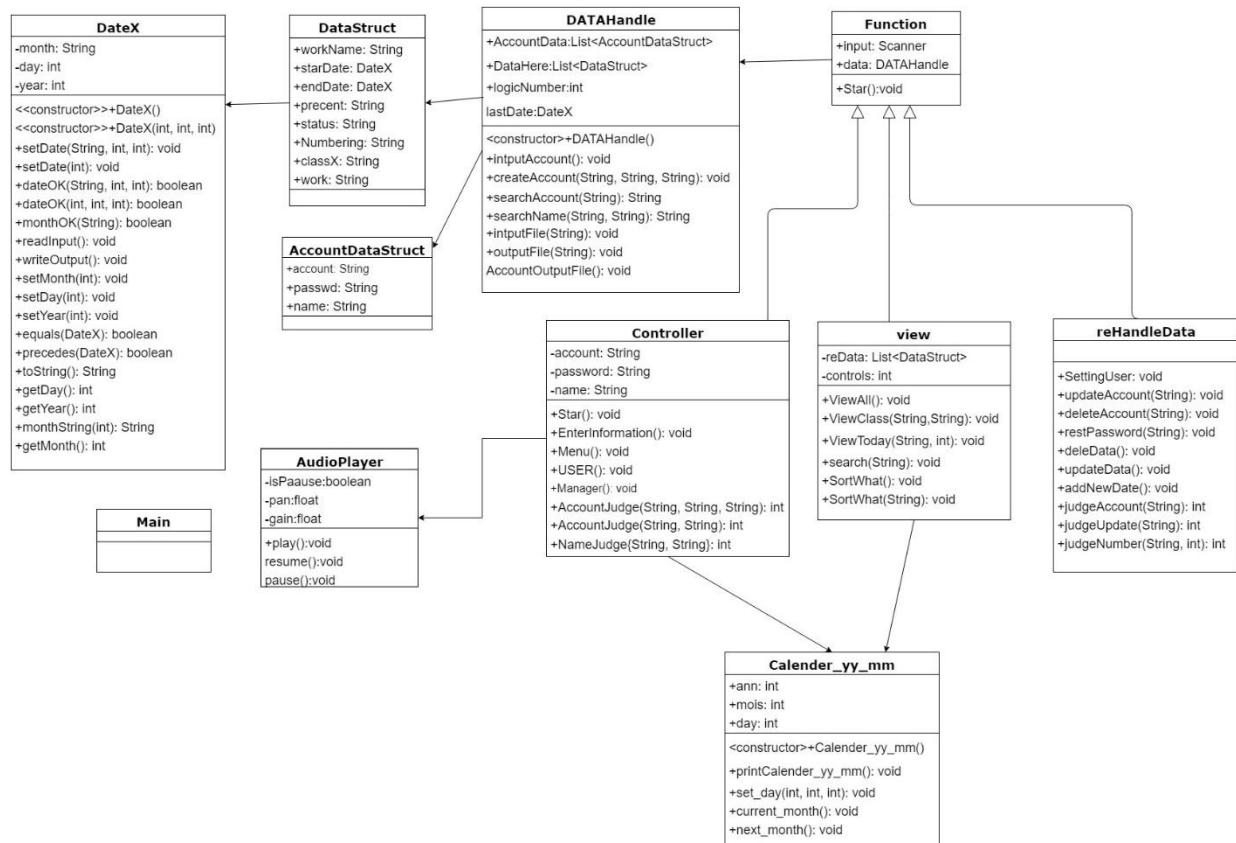
程式語言: Java

操作介面: Command 介面

## 4. 系統架構

### 4.1 Architecture diagram

此專案的 UML 檔案。



圖一、整體 UML

### 4.2 產品整合順序

先訂好資料結構之後，從 Controllers 為基礎開始開發使用者介面，在開發過程中，同步製作資料庫系統。

之後由使用者介面延伸將其功能實作，同時寫出 view、reHandleData 類別，完成整合。

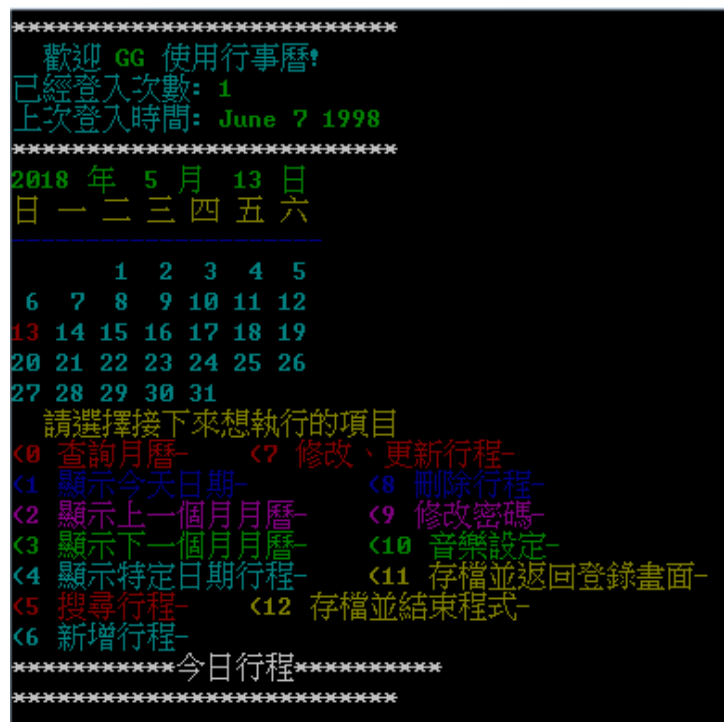
## 5. 系統設計

有一個介面敘述及六個元件設計介紹。

### 5.1 使用者介面描述

1. 創建帳號及登入
2. 顯示上次登入時間及登入次數
3. 顯示目前日期及月曆
4. 提醒今天的行程
5. 觀看上下月，或輸入要觀看的月份查詢日曆
6. 利用名子、開始時間、完成百分比、行程編號、行程分類、內容，創建一個行程
7. 搜尋特定日期行程，以及觀看行程
8. 使用名子、開始時間、完成百分比、行程編號、行程分類、內容，檢索想看的行程
9. 觀看日期的時候可以使用參數進行排序
10. 刪除、修改、更新行程
11. 修改帳號密碼
12. 調整音樂大小聲、音樂有無

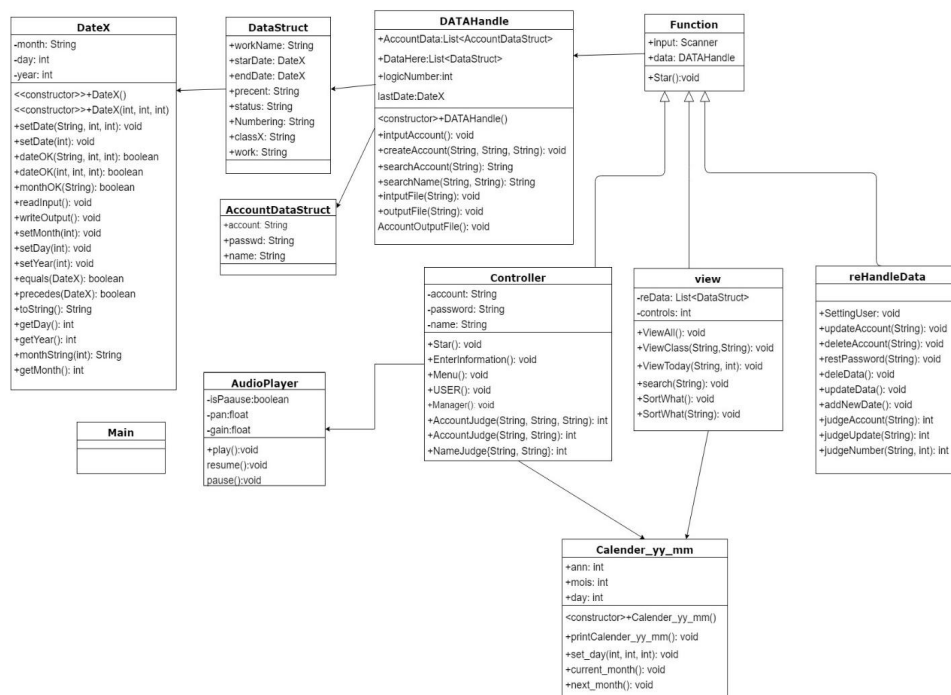
#### 5.1.1 畫面



圖二、主畫面



## 5.1.2 物件及動作



圖三、物件動作

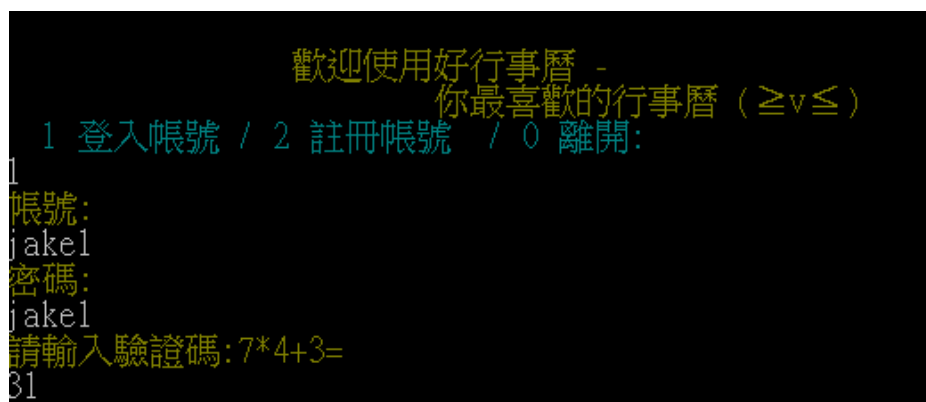
## 5.2 主要介面 設計規格

為主要介面，負責登入畫面，當日行程提醒，提供使用者及管理者的主畫面，音樂控制子畫面，行程搜尋子畫面。

登入畫面判斷帳號密碼、驗證碼是否正確，註冊則判斷帳號是否重複。

### 5.2.1 主要介面 介面描述

輸入：使用者輸入帳號(account)密碼(passwd)登入畫面。



圖四、主要介面

輸出：登入畫面，主畫面指令提示部分，當日行程提醒，音樂控制，行程搜尋子畫面。

```
*****
歡迎 jake 使用行事曆!
已經登入次數: 7
上次登入時間: June 18 1998
*****
2018 年 5 月 21 日
日 一 二 三 四 五 六
-----
      1  2  3  4  5
6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30 31
請選擇接下來想執行的項目
(0) 查詢月曆-          (7) 修改、更新行程-
(1) 顯示今天日期-      (8) 刪除行程-
(2) 顯示上一個月月曆-  (9) 修改密碼-
(3) 顯示下一個月月曆-  (10) 音樂設定-
(4) 顯示特定日期行程-  (11) 存檔並返回登錄畫面-
(5) 搜尋行程-          (12) 存檔並結束程式-
(6) 新增行程-
*****今日行程*****
工作名稱:
實習
完成狀態:
完成
*****
```

圖五、登入畫面

### 5.2.2 主要介面 運作說明

演算法：登入與註冊的帳號密碼判斷使用 Sequential Search。

層次結構：為輸入/輸出管理及管理員控制層。

限制：帳號密碼限制特殊符號及長度，其餘則無。

性能：資料量多時，因時間複雜度平均為  $O((N+1)/2)$ ，較消耗時間。

Function：

Start():匯入帳號資訊之後進入登入畫面

EnterInformation():登入或註冊帳號，帳號、密碼及驗證碼正確則依照權限進入主畫面。

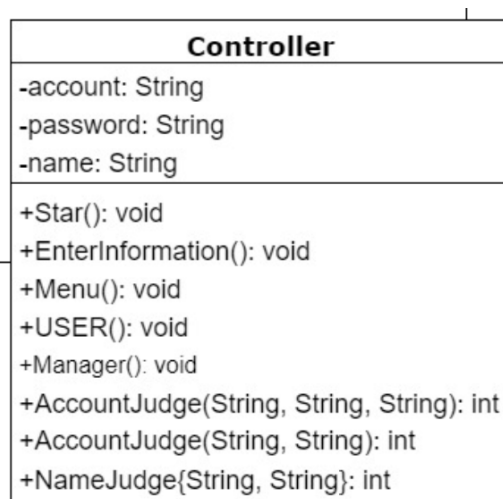
Menu():使用者主畫面，顯示月曆、指令及當日行程並依照輸入給予相對應的回應。

Manager():管理者主畫面，指令提示及輸入判斷。

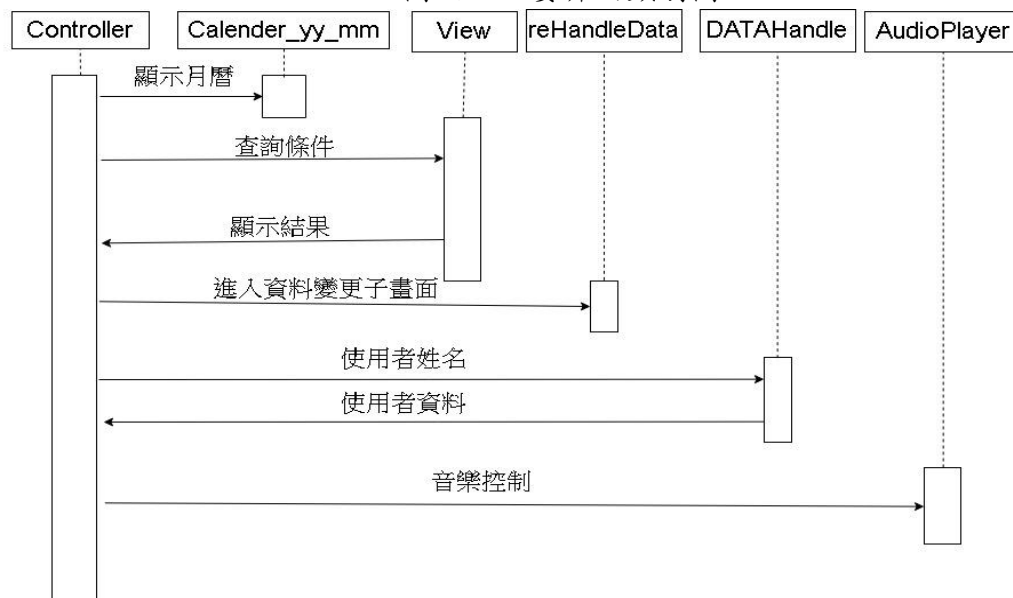
AccountJudge():帳號密碼判斷。

NameJudge():姓名帳號判斷。

### 5.2.3 互動圖



圖六、主要介面類別圖



圖七、主要介面互動圖

## 5.3 資料庫 設計規格

負責處理帳號資料的匯入/匯出，與帳號驗證，以及資料匯入/匯出，資料庫定義。輸入帳號密碼後判斷是否相符，呼叫後匯出或匯入資料。

### 5.3.1 資料庫 介面描述

輸入：從資料庫匯入資料，獲得 Controller 需判斷的帳號密碼。

輸出：匯出資料到資料庫，判斷帳號密碼後的結果 return。

### 5.3.2 資料庫 運作說明

演算法：帳號密碼是否相符使用 Sequential Search。

層次結構：為儲存管理及輸入\輸出層。

限制：資料應符合結構。

性能：資料量多時，因時間複雜度平均為  $O((N+1)/2)$ ，較消耗時間。

Function：

`inputAccount()`: 匯入帳號相關。

createAccount():依照匯入資料新增帳號。

searchAccount():尋找帳號是否存在。

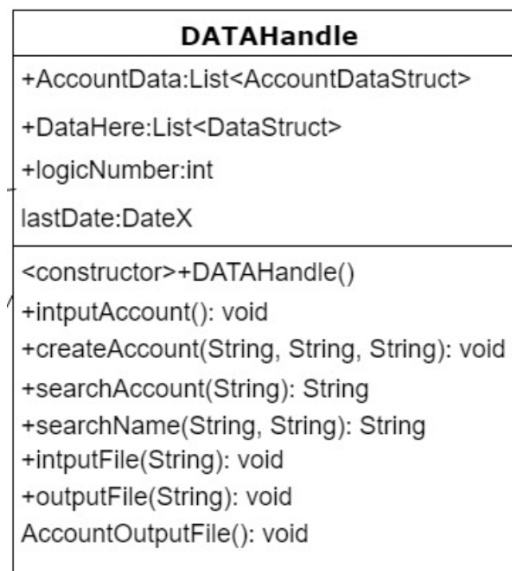
searchName():尋找姓名是否存在。

inputFile():依姓名匯入資料庫。

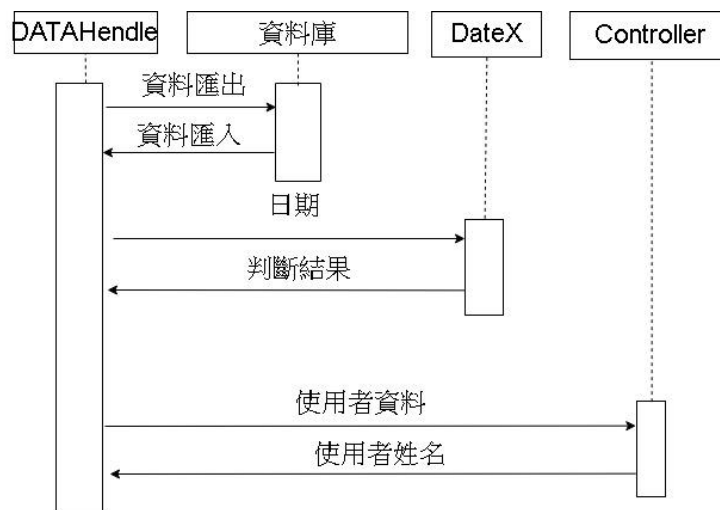
outputFile():匯出資料。

AccountOutputFile():匯出帳號資料庫。

### 5.3.3 互動圖



圖八、資料庫類別圖



圖九、資料庫互動圖

## 5.4 月曆 設計規格

呼叫則印出當天月曆，可以顯示上下月，也可以查詢某年某月的月曆。

### 5.4.1 月曆 介面描述

輸入：可有日期可無。

輸出：依照日期印出月曆。

### 5.4.2 月曆 運作說明

層次結構：為輸入\輸出層。

Function：

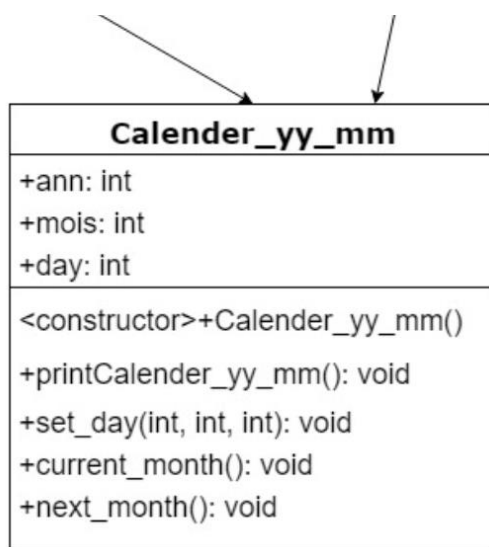
printCalender\_yy\_mm()依照儲存日期印出月曆。

Set\_day()給予日期來改變要印的月曆。

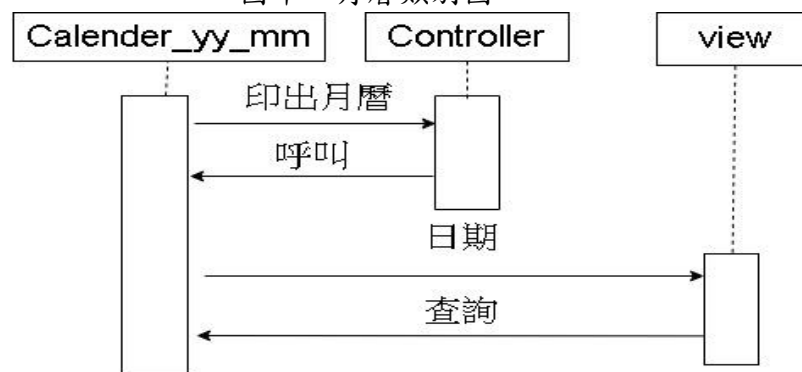
current\_month()月份-1

next\_month()月份+1

### 5.4.3 互動圖



圖十、月曆類別圖



圖十一、月曆介面互動圖

## 5.5 資料處理 設計規格

新增、更新或刪除帳號與資料。判斷帳號是否有非法字元。

### 5.5.1 資料處理介面描述

輸入：欲更新的使用者。

輸出：更新後的資料。

### 5.5.2 資料處理 運作說明

演算法：刪除、更新帳號、資料使用 Sequential Search。

層次結構：為儲存管理層。

限制：資料只是暫存於 list。

性能：資料量多時，因時間複雜度平均為  $O((N+1)/2)$ ，較消耗時間。

Function：

SettingUser():輸入使用者，排序其資料。

updateAccount():改變密碼。

deleteAccount():刪除帳號。

restPassword():密碼設為 00000000。

deleData():刪除資料。

updateData():更新資料。

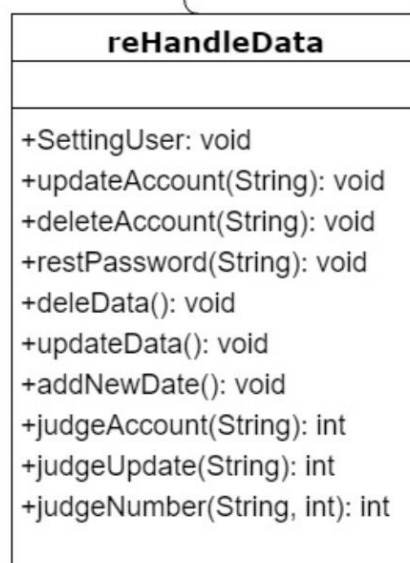
addNewData():新增資料。

judgeAccount():判斷是否為帳號。

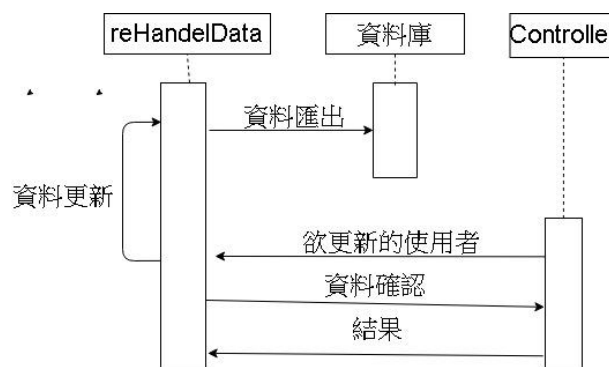
judgeUpdata():判斷是第幾個資料要更新。

judgeNumber():判斷是否為數字’。

### 5.5.3 互動圖



圖十二、資料處理類別圖



圖十三、資料處理互動圖

## 5.6 資料顯示 設計規格

依照排序的資料分頁顯示。

### 5.6.1 資料顯示 介面描述

輸入：無

輸出：愈顯示的資料

### 5.6.2 資料顯示運作說明

演算法：尋找資料使用 Sequential Search。

層次結構：為輸入/輸出層。

性能：資料量多時，因時間複雜度平均為  $O((N+1)/2)$ ，較消耗時間。

Function：

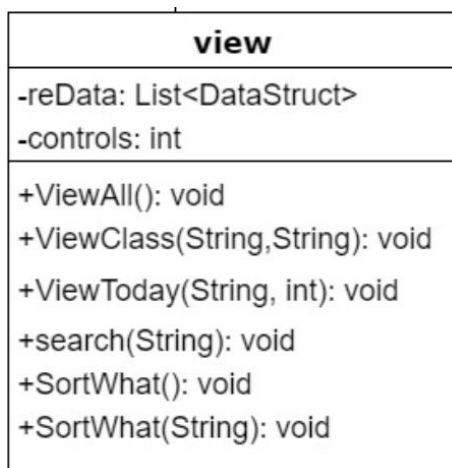
ViewAll():看所有行程。

ViewClass():依照 class 來找行程。

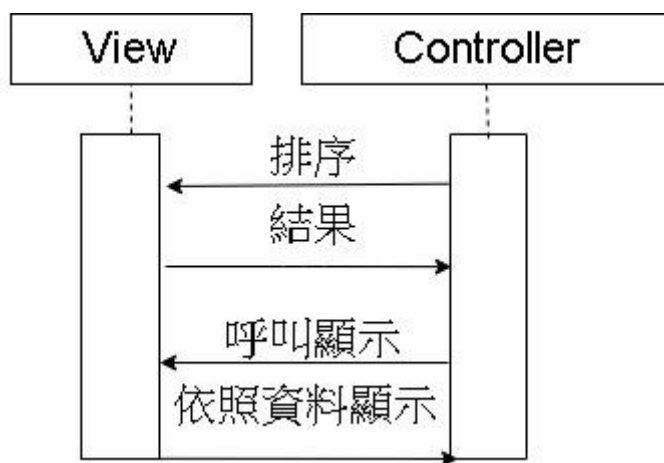
ViewToday():找今天所有進行中的行程。

sortWhat():子畫面，可以排序行程。

### 5.6.3 互動圖



圖十四、資料顯示類別圖



圖十五、資料顯示互動圖

## 5.7 音樂撥放器 設計規格

控制音樂大小聲，開始、暫停與結束。

### 5.7.1 音樂撥放器介面描述

輸入：音樂檔。

輸出：音樂。

### 5.7.2 音樂撥放器 運作說明

限制：需給予音樂檔

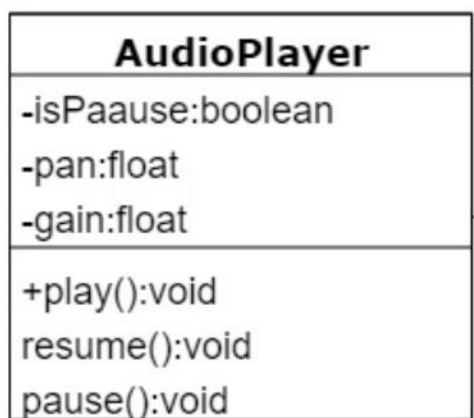
Function：

Play():開始撥放。

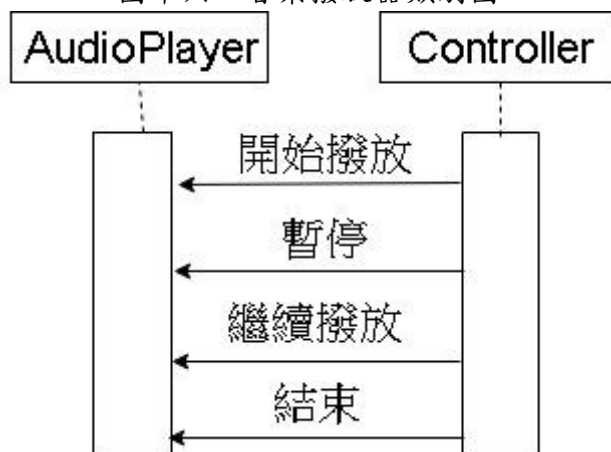
Resume():繼續撥放。

Pause():暫停撥放。

### 5.7.3 互動圖



圖十六、音樂撥放器類別圖



圖十七、音樂撥放器類別圖

## 5.8 元件整合條件

在同一資料夾底下。

每個模組內的程式都提供接口讓其他程式呼叫。



## 5.9 設計規則

1. 保持設計的一致性
2. 提供重度使用者便捷操作
3. 適時提供對使用者有意義的回饋
4. 明確告知使用者動作的狀態
5. 簡化錯誤的處理
6. 務必提供可回復的設計
7. 滿足使用者控制的慾望
8. 簡化操作，降低使用者記憶需求

## 6. 非功能設計

---

### 6.1 效能設計

主要演算法為 sequence sort。設計方便但平均時間複雜度為  $O((n+1)/2)$ 。

### 6.2 安全性設計

不支援多人以及網路。

### 6.3 可靠性設計

結構標準化，有 Fool proof，當機也不會損毀資料庫，

### 6.4 可維護性設計

所有程式需要以模組化的方式開發，以加快維護效率。將來需要與更大的系統做整合時，修改的幅度能降到最低。

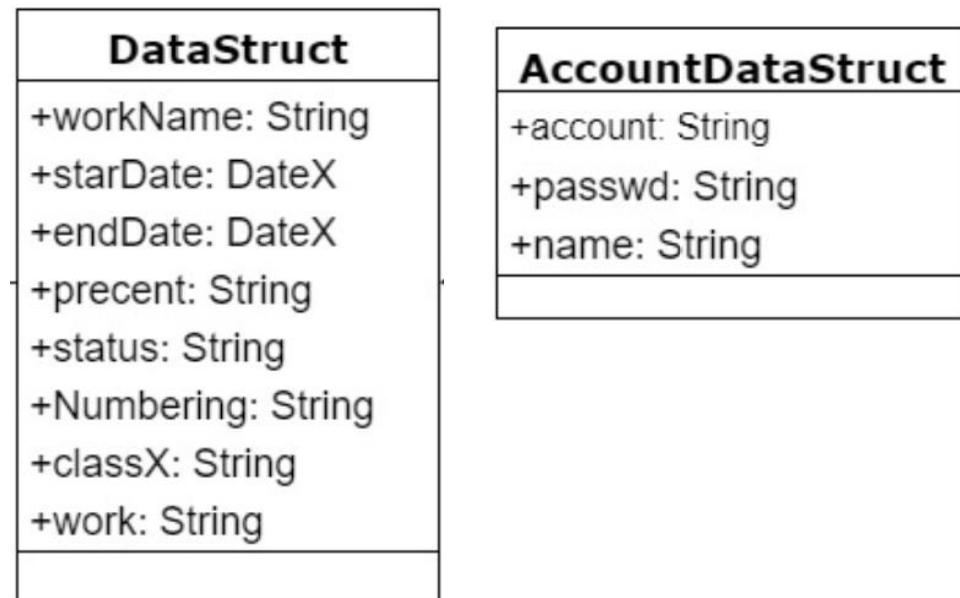
### 6.5 邏輯資料庫設計

詳見第七點

## 7. 資料設計

### 7.1 內部軟體資料結構

我們軟體資料結構主要有兩個部分，分別是:DataStruct、AccountDataStruct



圖十九、圖二十、資料結構圖

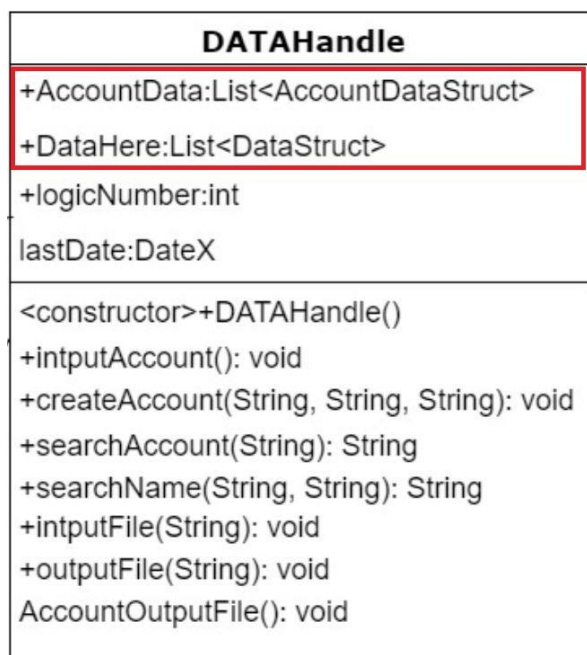
在 DataStruct 資料結構中清楚指定了行程資料的格式跟內容

資料名稱	資料格式	資料名稱說明
workName	String	行程名稱
startDate	DateX	行程開始時間
endDate	DateX	行程結束時間
precent	String	行程完成度
status	String	行程狀態(完成、執行中、未開始)
Numbering	String	行程編號
classX	String	行程分類
work	String	行程內容說明

在 AccountDataStruct 資料結構中說明了帳號資料的格式

資料名稱	資料格式	資料名稱說明
account	String	帳號
passwd	String	密碼
name	String	使用者名稱

## 7.2 全域資料結構



圖二十一、DATAHandle 類別圖

在 DATAHandle 類別中宣告名稱為 AccountData 的 ArrayList，放入 AccountDataStruct 資料結構、

宣告名稱為 DataHere 的 ArrayList，放入 DataStruct 資料結構。


這兩個 ArrayList 中宣告為 static 確保資料只有一組且不會消失。

## 7.3 暫存性資料結構

在使用者操作完程式之後，暫存紀錄資料的方式為 txt 檔案。

所有的帳號會被存在 AccountData.txt，會存入 7.2 所說明 ArrayList 內容。

創建新的使用者後，會新增該使用者的名稱.txt 檔案，用於分開其他使用者的資料，加強其可靠性。

 AccountData.txt	2018/5/13 下午 0...	文字文件
 jake.txt	2018/5/13 下午 0...	文字文件
 Sam.txt	2018/5/13 下午 0...	文字文件
 Zero.txt	2018/5/13 下午 0...	文字文件

圖二十二、資料庫檔案

## 7.4 資料庫說明

主要處理資料庫的存取類別為 DATAHandle.java

啟動程式時 DATAHandle.java 中 inputAccount() 會去讀取 AccountData.txt 內所有的帳戶內的名稱帳號密碼存入 AccountData 的 ArrayList。

登入帳號時會去尋找 AccountData 的 ArrayList 中有無資料，沒有找到則顯示帳號不存在，找到資料時會去呼叫 使用者名稱.txt 檔案內內容存入 DataHere 的 ArrayList 中，因為我們一次只登入一個帳號所以可以這樣使用。

使用者可以藉由介面選項去改變 DataHere 的 ArrayList 內容做處理。

詳細函數說明：

inputAccount(): 匯入帳號相關。

createAccount(): 依照匯入資料新增帳號。

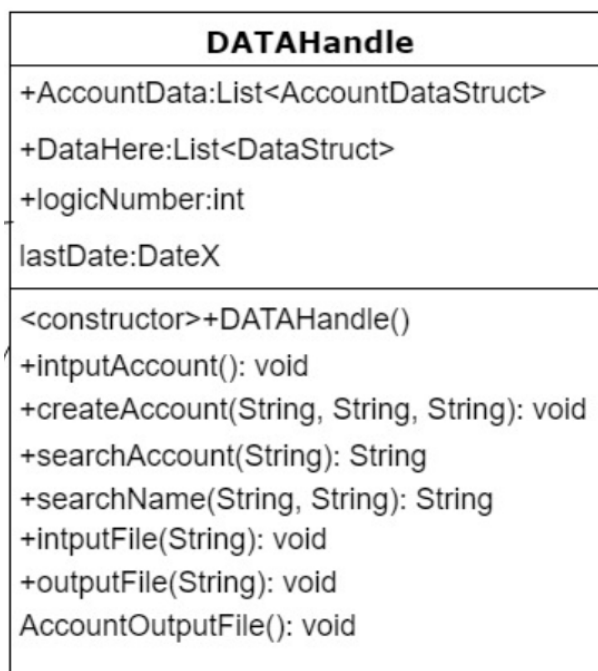
searchAccount(): 尋找帳號是否存在。

searchName(): 尋找姓名是否存在。

inputFile(): 依姓名匯入資料庫。

outputFile(): 匯出資料。

AccountOutputFile(): 匯出帳號資料庫。



圖二十三、資料庫處理類別圖