

物件導向軟體工程

(OBJECT-ORIENTED SOFTWARE ENGINEERING)

# Homework 2

(Java 航空訂票系統)

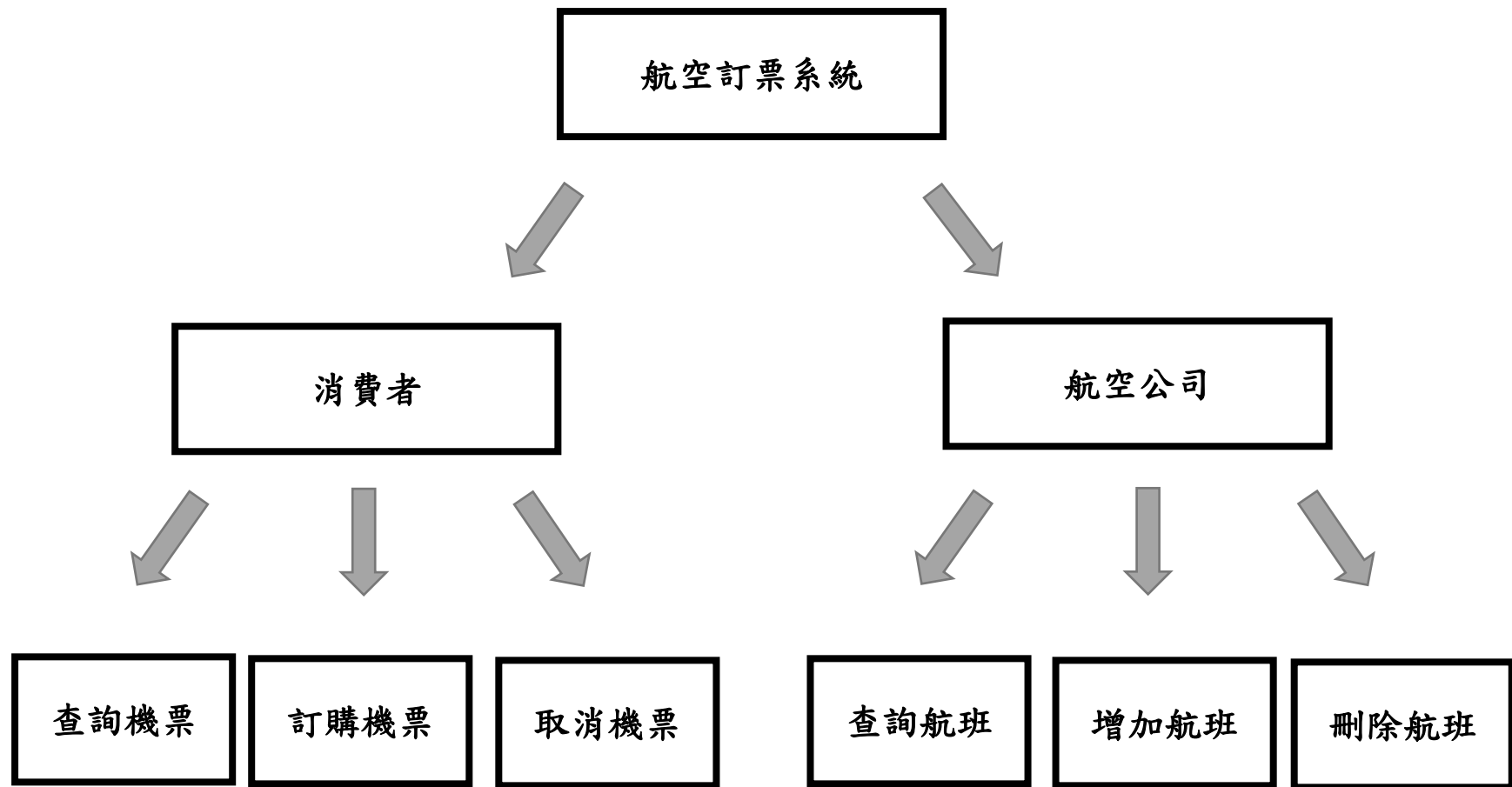
日期:2017/11/09

學號:P76064538

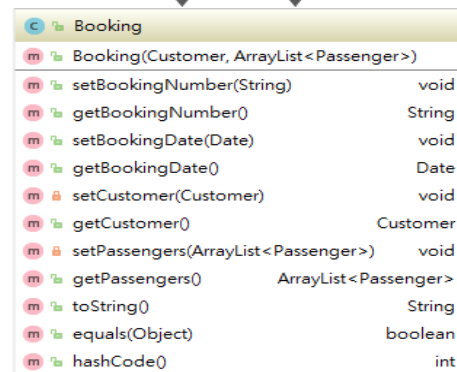
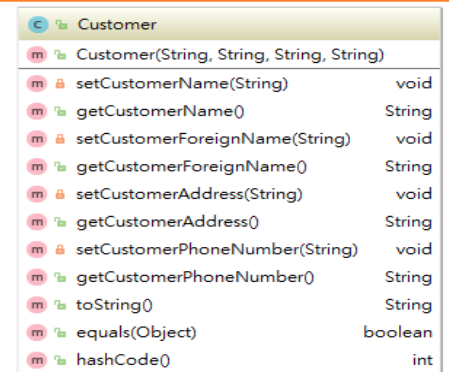
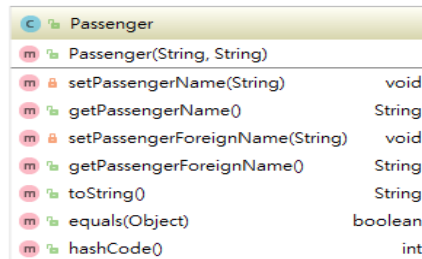
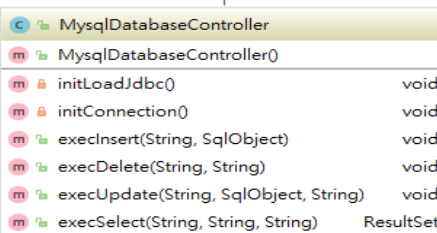
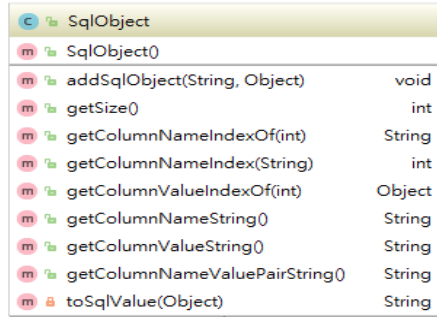
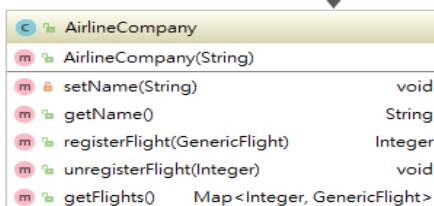
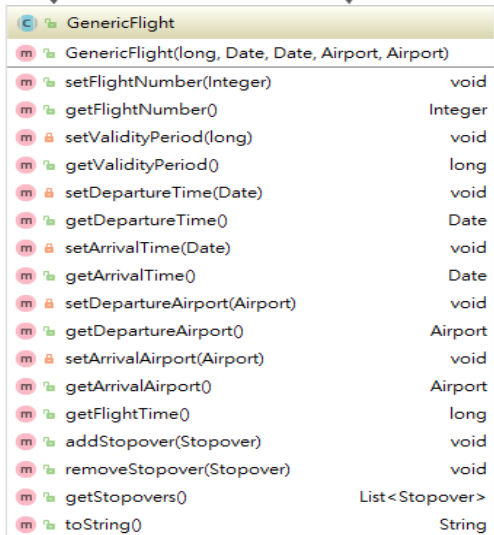
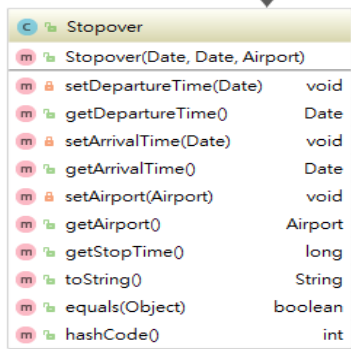
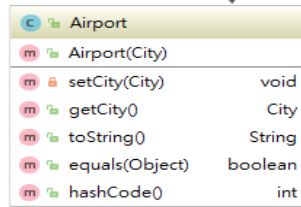
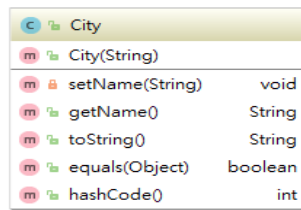
姓名:簡君聿

(程式碼已上傳至 <https://github.com/Alex-CHUN-YU/OOP/>)

## 系統主架構：



# UML



## 航空公司系統 DEMO

以連結至:jdbc:mysql://127.0.0.1/:flight\_booking:root:1234資料庫。

請輸入航班公司代號:

FE

歡迎 FE 航空公司登入本系統~

你好 ~ 此為航空公司增加航班系統 !

輸入1為增加航班資訊 | 輸入2為刪除航班 | 輸入3為查詢目前所有航班資訊 | 輸入4離開本系統

3

以下為 FE 航空公司目前所有航班資訊:

航空代碼:FE000

出發時間:2017-10-11 13:00 抵達時間:2017-10-11 16:00

出發地點:Melbourne Airport 抵達地點:Kaohsiung International Airport

總花費時間:3小時

航空代碼:FE001

出發時間:2017-12-25 13:00 抵達時間:2017-12-26 03:00

出發地點:Melbourne Airport 抵達地點:Songshan Airport

總花費時間:14小時

你好 ~ 此為航空公司增加航班系統 !

查詢航班

## 查詢航班 DB

查詢航班 DB		id	name ▲ 1	flight_number	departure_airport	departure_date	arrival_airport	arrival_date	validity_period		
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	1	BR	0	Taoyuan International	2017-10-18 13:00	Melbourne	2017-10-19 9:00	10
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	2	BR	1	Kaohsiung International	2017-10-18 13:00	Heathrow	2017-10-18 15:00	9
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	14	BR	3	Tainan	2017-11-19 13:00	Kaohsiung International	2017-11-19 15:00	13
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	13	BR	2	Melbourne	2017-12-19 13:00	Charles de Gaulle	2017-12-20 13:00	11
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	12	CI	2	Charles de Gaulle	2017-10-19 13:00	John F. Kennedy International	2017-10-20 9:00	5
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	11	CI	1	Songshan	2017-11-01 09:00	Charles de Gaulle	2017-11-02 05:00	7
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	4	CI	0	Melbourne	2017-11-11 13:00	Taoyuan International	2017-11-13 13:00	7
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	15	FE	0	Melbourne	2017-10-11 13:00	Kaohsiung International	2017-10-11 16:00	4
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	17	FE	0	Melbourne	2017-12-25 13:00	Songshan	2017-12-26 3:00	4
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	6	GE	1	Kaohsiung International	2017-10-19 15:00	Songshan	2017-10-19 20:00	5
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	5	GE	0	John F. Kennedy International	2017-12-19 13:00	Tainan	2017-12-20 07:00	7
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	25	GE	2	Taoyuan International	2017-12-19 13:00	Heathrow	2017-12-20 09:00	4
<input type="checkbox"/>	編輯	<input type="checkbox"/> 複製	<input type="checkbox"/> 刪除	16	JAL	0	Tokyo	2017-11-12 13:00	Osaka	2017-11-12 14:00	7

☐ 全選
 已選擇項目:
 編輯
 複製
 ☐ 刪除
 匯出


 全選
  已選擇項目:
  編輯
  複製
  刪除
  匯出

你好 ~ 此為航空公司增加航班系統 !

輸入1為增加航班資訊 | 輸入2為刪除航班 | 輸入3為查詢目前所有航班資訊 | 輸入4離開本系統  
1

請填寫以下航班資訊:

輸入出發城市之機場:

*Charles de Gaulle*

輸入出發時間:(EX:2017-10-19 13:00)

*2017-12-06 09:00*

輸入抵達城市之機場:

*Taoyuan International*

輸入抵達時間:(EX:2017-10-19 13:00)

*2017-12-07 07:00*










輸入航班定票有效期限:

4

增加航班資訊成功!(如沒有要繼續增加航班請輸入0)

增加航班

## 新增航班後 DB

	id	name ▲ 1	flight_number	departure_airport	departure_date	arrival_airport	arrival_date	validity_period
<input type="checkbox"/>  編輯  複製  刪除	1	BR	0	Taoyuan International	2017-10-18 13:00	Melbourne	2017-10-19 9:00	10
<input type="checkbox"/>  編輯  複製  刪除	14	BR	3	Tainan	2017-11-19 13:00	Kaohsiung International	2017-11-19 15:00	13
<input type="checkbox"/>  編輯  複製  刪除	13	BR	2	Melbourne	2017-12-19 13:00	Charles de Gaulle	2017-12-20 13:00	11
<input type="checkbox"/>  編輯  複製  刪除	2	BR	1	Kaohsiung International	2017-10-18 13:00	Heathrow	2017-10-18 15:00	9
<input type="checkbox"/>  編輯  複製  刪除	11	CI	1	Songshan	2017-11-01 09:00	Charles de Gaulle	2017-11-02 05:00	7
<input type="checkbox"/>  編輯  複製  刪除	12	CI	2	Charles de Gaulle	2017-10-19 13:00	John F. Kennedy International	2017-10-20 9:00	5
<input type="checkbox"/>  編輯  複製  刪除	4	CI	0	Melbourne	2017-11-11 13:00	Taoyuan International	2017-11-13 13:00	7
<input type="checkbox"/>  編輯  複製  刪除	17	FE	0	Melbourne	2017-12-25 13:00	Songshan	2017-12-26 3:00	4
<input type="checkbox"/>  編輯  複製  刪除	15	FE	0	Melbourne	2017-10-11 13:00	Kaohsiung International	2017-10-11 16:00	4
<input type="checkbox"/>  編輯  複製  刪除	27	FE	2	Charles de Gaulle	2017-12-06 09:00	Taoyuan International	2017-12-07 07:00	4
<input type="checkbox"/>  編輯  複製  刪除	6	GE	1	Kaohsiung International	2017-10-19 15:00	Songshan	2017-10-19 20:00	5
<input type="checkbox"/>  編輯  複製  刪除	25	GE	2	Taoyuan International	2017-12-19 13:00	Heathrow	2017-12-20 09:00	4
<input type="checkbox"/>  編輯  複製  刪除	5	GE	0	John F. Kennedy International	2017-12-19 13:00	Tainan	2017-12-20 07:00	7
<input type="checkbox"/>  編輯  複製  刪除	16	JAL	0	Tokyo	2017-11-12 13:00	Osaka	2017-11-12 14:00	7

以下為 FE 航空公司目前所有航班資訊：

航空代碼:FE000

出發時間:2017-12-25 13:00 抵達時間:2017-12-26 03:00

出發地點:Melbourne Airport 抵達地點:Songshan Airport

總花費時間:14小時

航空代碼:FE001

出發時間:2017-10-11 13:00 抵達時間:2017-10-11 16:00

出發地點:Melbourne Airport 抵達地點:Kaohsiung International Airport

總花費時間:3小時

航空代碼:FE002

出發時間:2017-12-06 09:00 抵達時間:2017-12-07 07:00

出發地點:Charles de Gaulle Airport 抵達地點:Taoyuan International Airport

總花費時間:22小時

你好 ~ 此為航空公司增加航班系統 !

輸入1為增加航班資訊 | 輸入2為刪除航班 | 輸入3為查詢目前所有航班資訊 | 輸入4離開本系統

2

請輸入航班號碼:

1

刪除成功!!

刪除航班



# 刪除航班後 DB

	id	name ▲ 1	flight_number	departure_airport	departure_date	arrival_airport	arrival_date	validity_period
<input type="checkbox"/> 編輯  複製  刪除	1	BR	0	Taoyuan International	2017-10-18 13:00	Melbourne	2017-10-19 9:00	10
<input type="checkbox"/> 編輯  複製  刪除	2	BR	1	Kaohsiung International	2017-10-18 13:00	Heathrow	2017-10-18 15:00	9
<input type="checkbox"/> 編輯  複製  刪除	14	BR	3	Tainan	2017-11-19 13:00	Kaohsiung International	2017-11-19 15:00	13
<input type="checkbox"/> 編輯  複製  刪除	13	BR	2	Melbourne	2017-12-19 13:00	Charles de Gaulle	2017-12-20 13:00	11
<input type="checkbox"/> 編輯  複製  刪除	12	CI	2	Charles de Gaulle	2017-10-19 13:00	John F. Kennedy International	2017-10-20 9:00	5
<input type="checkbox"/> 編輯  複製  刪除	11	CI	1	Songshan	2017-11-01 09:00	Charles de Gaulle	2017-11-02 05:00	7
<input type="checkbox"/> 編輯  複製  刪除	4	CI	0	Melbourne	2017-11-11 13:00	Taoyuan International	2017-11-13 13:00	7
<input type="checkbox"/> 編輯  複製  刪除	17	FE	0	Melbourne	2017-12-25 13:00	Songshan	2017-12-26 3:00	4
<input type="checkbox"/> 編輯  複製  刪除	27	FE	2	Charles de Gaulle	2017-12-06 09:00	Taoyuan International	2017-12-07 07:00	4
<input type="checkbox"/> 編輯  複製  刪除	6	GE	1	Kaohsiung International	2017-10-19 15:00	Songshan	2017-10-19 20:00	5
<input type="checkbox"/> 編輯  複製  刪除	5	GE	0	John F. Kennedy International	2017-12-19 13:00	Tainan	2017-12-20 07:00	7
<input type="checkbox"/> 編輯  複製  刪除	25	GE	2	Taoyuan International	2017-12-19 13:00	Heathrow	2017-12-20 09:00	4
<input type="checkbox"/> 編輯  複製  刪除	16	JAL	0	Tokyo	2017-11-12 13:00	Osaka	2017-11-12 14:00	7

以連結至:jdbc:mysql://127.0.0.1/:flight\_booking:root:1234資料庫。

你好 ~ 此為顧客訂票查詢系統 !

輸入1為訂購機票 | 輸入2為查詢機票 | 輸入3取消機票 | 輸入4離開本系統

2

請輸入機票訂單號碼:

3GE5c

請輸入本名:

盧廣仲

定位代碼:3GE5c

查詢機票

消費者資訊:

姓名:盧廣仲 外國名:Crowd Lu 住址:Taiwan 行動電話:0988745689

乘客資訊:

[姓名:盧廣仲 外國名:Crowd Lu, 姓名:陳綺貞 外國名:Cheer Chen]

## 消費者資訊

消費者資訊

		name	flight_number	booking_id	customer_name	customer_foreign_name	customer_address	customer_phone_number					
<input type="checkbox"/>		編輯		複製		刪除	CI	2	1CI5c	張懸	Deserts Xuan	Taiwan	0988564256
<input type="checkbox"/>		編輯		複製		刪除	GE	2	3GE5c	盧廣仲	Crowd Lu	Taiwan	0988745689
<input type="checkbox"/>		編輯		複製		刪除	CI	0	9CI3	張忠謨	Chalie Puth	taipai	0988745632
<input type="checkbox"/>		編輯		複製		刪除	CI	0	9CI6d	周杰倫	Jay Chou	Taiwan Taipei	0988719738

☐ 全選

已選擇項目:

編輯

複製

刪除

匯出

+ 選項

←

→

## 乘客資訊

歡迎進入訂票系統！

請輸入出發地點：

Melbourne

請輸入抵達地點：

Charles de Gaulle

請輸入出發日期：(格式：2017-11-03)

2017-12-19

請輸入消費者姓名：

簡君聿

請輸入消費者護照英文名：

ALEX

請輸入消費者住址：

Taiwan

請輸入消費者電話：

0988719738

請輸入乘客姓名：

查理

請輸入乘客護照英文名：

Charlie

如要繼續增加請輸入1，沒有請輸入 0 離開

以下為你的航班資訊祝你有個愉快的旅程^^

定位代碼:9BR3b

消費者資訊：

姓名:簡君聿 外國名:ALEX 住址:Taiwan 行動電話:0988719738

乘客資訊：

[姓名:查理 外國名:Charlie]

訂購機票

飛機號碼:BR2

出發時間:2017-12-19 13:00 抵達時間:2017-12-20 13:00

出發地點:Melbourne 抵達地點:Charles de Gaulle

機票有效期限:11天

訂完機票後

訂完機票後

	name	flight_number	booking_id	customer_name	customer_foreign_name	customer_address	customer_phone_number
<input type="checkbox"/> 編輯 複製 刪除	CI	2	1CI5c	張懸	Deserts Xuan	Taiwan	0988564256
<input type="checkbox"/> 編輯 複製 刪除	GE	2	3GE5c	盧賓仲	Crowd Lu	Taiwan	0988745689
<input type="checkbox"/> 編輯 複製 刪除	BR	2	9BR3b	簡君聿	ALEX	Taiwan	0988719738
<input type="checkbox"/> 編輯 複製 刪除	CI	0	9CI3	張志謀	Charlie Puth	taipai	0988745632
<input type="checkbox"/> 編輯 複製 刪除	CI	0	9CI6d	周杰倫	Jay Chou	Taiwan Taipai	0988719738

☐ 全選
 已選擇項目:
 ☐ 編輯
 ☐ 複製
 ☐ 刪除
 ☐ 匯出

以連結至:jdbc:mysql://127.0.0.1/:flight\_booking:root:1234資料庫。

你好 ~ 此為顧客訂票查詢系統 !

輸入1為訂購機票 | 輸入2為查詢機票 | 輸入3取消機票 | 輸入4離開本系統

3

請輸入欲取消之訂單號碼:

9CI3ddd

查無此訂單!!

輸入1為訂購機票 | 輸入2為查詢機票 | 輸入3取消機票 | 輸入4離開本系統

3

請輸入欲取消之訂單號碼:

1CI5c

取消班機資訊成功!

取消機票

+ 選項

<input type="checkbox"/>				CI	2	1CI5c	張懸	Deserts Xuan	Taiwan	0988564256
<input type="checkbox"/>				GE	2	3GE5c	盧廣仲	Crowd Lu	Taiwan	0988745689
<input type="checkbox"/>				BR	2	9BR3b	簡君聿	ALEX	Taiwan	0988719738
<input type="checkbox"/>				CI	0	9CI3	張忠謨	Chalie Puth	taipai	0988745632
<input type="checkbox"/>				CI	0	9CI6d	周杰倫	Jay Chou	Taiwan Taipai	0988719738

取消前

已選擇項目: 編輯 複製 刪除 匯出

+ 選項

← T →

▼

		name	flight_number	booking_id	customer_name	customer_foreign_name	customer_address	customer_phone_number		
<input type="checkbox"/>				GE	2	3GE5c	盧廣仲	Crowd Lu	Taiwan	0988745689
<input type="checkbox"/>				BR	2	9BR3b	簡君聿	ALEX	Taiwan	0988719738
<input type="checkbox"/>				CI	0	9CI3	張忠謨	Chalie Puth	taipai	0988745632
<input type="checkbox"/>				CI	0	9CI6d	周杰倫	Jay Chou	Taiwan Taipai	0988719738

↑

☐ 全選

已選擇項目: 編輯 複製 刪除 匯出

取消後

## Booking Class

booking\_information package

```
package booking_information;
```

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

```
import java.util.Date;
```

```
/**
```

```
 * 主要負責存取機票預訂資訊。
```

```
 * 存:預定號碼、預定飛機時間、消費者資訊、乘客資訊
```

```
 * 取:預定號碼、預定飛機時間、消費者資訊、乘客資訊
```

```
 * @version 1.0 2017 年 10 月 28 日
```

```
 * @author ALEX-CHUN-YU
```

```
 */
```

```
public class Booking {
```

```
    /**
```

```
     * Booking Number.
```

```
     */
```

```
    private String number = null;
```

```
    /**
```

```
     * Booking Date.
```

```
     */
```

```
    private Date date = new Date();
```

```
    /**
```

```
     * Customer Information.
```

```
     */
```

```
    private Customer customer;
```

```
    /**
```

```
     * Passengers Information.
```

```
     */
```

```
    private ArrayList<Passenger> passengers;
```

```
    /**
```

```
     * Constructor About Booking Information.
```

```
     * @param customer customer
```



```

    * @param passengers passengers
    */
    public Booking(Customer customer, ArrayList<Passenger>
passengers) {
        this.setCustomer(customer);
        this.setPassengers(passengers);
    }

    /**
     * Set Booking Information About Booking Number.
     * @param number booking number.
     */
    public void setBookingNumber(String number) {
        this.number = number;
    }

    /**
     * Get Booking Information About Booking Number.
     */
    public String getBookingNumber() {
        return this.number;
    }

    /**
     * Set Booking Information About Booking Date.
     * @param date booking date
     */
    public void setBookingDate(Date date) {
        this.date = date;
    }

    /**
     * Get Booking Information About Booking Date.
     */
    public Date getBookingDate() {
        return this.date;
    }
}

```

```

/**
 * Set Booking Information About Customer.
 * @param customer booking customer
 */
private void setCustomer(Customer customer) {
    this.customer = customer;
}

/**
 * Get Booking Information About Customer.
 */
public Customer getCustomer() {
    return this.customer;
}

/**
 * Set Booking Information About Passengers.
 * @param passengers booking passenger
 */
private void setPassengers(ArrayList<Passenger> passengers) {
    this.passengers = passengers;
}

/**
 * Get Booking Information About Passengers.
 */
public ArrayList<Passenger> getPassengers() {
    return this.passengers;
}

@Override
public String toString() {
    return "定位代碼:" + this.number
        + "\n\n 消費者資訊:\n" + this.customer.toString()
        + "\n\n 乘客資訊:\n" + this.passengers.toString() +
"\n\n";
}

```

```

@Override
public boolean equals(Object obj) {
    if (obj.getClass().equals(this.getClass())) {
        Booking booking = (Booking) obj;
        if (booking.getBookingNumber().equals(this.number)
            && booking.getBookingDate().equals(this.date)
            && booking.getCustomer().equals(this.customer)
            &&
booking.getPassengers().equals(this.passengers)) {
            return true;
        }
    }
    return false;
}

@Override
public int hashCode() {
    return super.hashCode();
}
}

```

## Customer Class

booking\_information package

package booking\_information;

/\*\*

\* 主要負責存取購買機票之消費者資訊.

\* 存: 消費者名子、消費者外國名、地址、連絡電話

\* 取: 消費者名子、消費者外國名、地址、連絡電話

\* @version 1.0 2017 年 10 月 27 日

\* @author ALEX-CHUN-YU

\*/

public class Customer {

/\*\*

\* Customer Name.

\*/

private String name = null;

/\*\*

\* Customer Foreign Name.

\*/

private String foreignName = null;

/\*\*

\* Customer Address.

\*/

private String address = null;

/\*\*

\* Customer Phone Number.

\*/

private String phoneNumber = null;

/\*\*

\* Constructor About Customer Information.

```

    * @param name customer name
    * @param foreignName customer foreign name
    * @param address customer address
    * @param phoneNumber customer phone number
    */
    public Customer(String name, String foreignName, String
address, String phoneNumber) {
        this.setCustomerName(name);
        this.setCustomerForeignName(foreignName);
        this.setCustomerAddress(address);
        this.setCustomerPhoneNumber(phoneNumber);
    }

    /**
     * Set Booking Information About Customer Name.
     * @param name customer name
     */
    private void setCustomerName(String name) {
        this.name = name;
    }

    /**
     * Get Booking Information About Customer Name.
     * @return customer name
     */
    public String getCustomerName() {
        return this.name;
    }

    /**
     * Set Booking Information About Customer Foreign Name.
     * @param foreignName customer foreign name
     */
    private void setCustomerForeignName(String foreignName) {
        this.foreignName = foreignName;
    }

    /**

```

```

    * Get Booking Information About Customer Foreign Name.
    * @return costumer foreign name
    */
    public String getCustomerForeignName() {
        return this.foreignName;
    }

    /**
     * Set Booking Information About Customer Address.
     * @param address customer address
     */
    private void setCustomerAddress(String address) {
        this.address = address;
    }

    /**
     * Get Booking Information About Customer Address.
     * @return customer address
     */
    public String getCustomerAddress() {
        return this.address;
    }

    /**
     * Set Booking Information About Customer Phone Number.
     * @param phoneNumber customer phone number
     */
    private void setCustomerPhoneNumber(String phoneNumber) {
        this.phoneNumber = phoneNumber;
    }

    /**
     * Get Booking Information About Customer Phone Number.
     * @return customer phone number
     */
    public String getCustomerPhoneNumber() {
        return this.phoneNumber;
    }

```

```

    @Override
    public String toString() {
        return "姓名:" + this.name + " 外國名:" +
this.foreignName + " 住址:"
        + this.address + " 行動電話:" +
this.phoneNumber;
    }

    @Override
    public boolean equals(Object obj) {
        if (obj.getClass().equals(this.getClass())) {
            Customer customer = (Customer) obj;
            if (customer.getCustomerName().equals(this.name)
                &&
customer.getCustomerForeignName().equals(this.foreignName)
                &&
customer.getCustomerPhoneNumber().equals(this.phoneNumber)
                &&
customer.getCustomerAddress().equals(this.address)) {
                return true;
            }
        }
        return false;
    }

    @Override
    public int hashCode() {
        return super.hashCode();
    }
}

```

## Passenger Class

booking\_information package

```
package booking_information;
```

```
/**
```

```
 * 主要負責存取消費者所購買之乘客資訊。
```

```
 * 存：乘客名子、乘客外國名
```

```
 * 取：乘客名子、乘客外國名
```

```
 * @version 1.0 2017 年 10 月 27 日
```

```
 * @author ALEX-CHUN-YU
```

```
 */
```

```
public class Passenger {
```

```
    /**
```

```
     * Passenger Name.
```

```
    */
```

```
    private String name = null;
```

```
    /**
```

```
     * Passenger Foreign Name.
```

```
    */
```

```
    private String foreignName = null;
```

```
    /**
```

```
     * Constructor About Passenger Information.
```

```
     * @param name passenger name
```

```
     * @param foreignName passenger foreign name
```

```
    */
```

```
    public Passenger(String name, String foreignName) {
```

```
        this.setPassengerName(name);
```

```
        this.setPassengerForeignName(foreignName);
```

```
    }
```

```
    /**
```

```
     * Set Booking Information About Passenger Name.
```



```

    * @param name passenger name
    */
    private void setPassengerName(String name) {
        this.name = name;
    }

    /**
     * Get Booking Information About Passenger Name.
     * @return passenger name
     */
    public String getPassengerName() {
        return this.name;
    }

    /**
     * Set Booking Information About Passenger Foreign Name.
     * @param foreignName passenger foreign name
     */
    private void setPassengerForeignName(String foreignName) {
        this.foreignName = foreignName;
    }

    /**
     * Get Booking Information About Passenger Foreign Name.
     * @return passenger foreign name
     */
    public String getPassengerForeignName() {
        return this.foreignName;
    }

    @Override
    public String toString() {
        return "姓名:" + this.name + " 外國名:" +
this.foreignName;
    }

    @Override

```

```

    public boolean equals(Object obj) {
        if (obj.getClass().equals(this.getClass())) {
            Passenger passenger = (Passenger) obj;
            if (passenger.getPassengerName().equals(this.name)
                &&
                passenger.getPassengerForeignName().equals(this.foreignName)) {
                return true;
            }
        }
        return false;
    }

    @Override
    public int hashCode() {
        return super.hashCode();
    }
}

```

## AirlineCompany Class

flight\_information package

```
package flight_information;
```

```
import java.util.Collections;
```

```
import java.util.HashMap;
```

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

```
/**
```

```
 * 存取航空公司資訊.
```

```
 * 存: 公司名、增加飛機、減少飛機
```

```
 * 取: 公司名、所有飛機
```

```
 * @version 1.0 2017 年 10 月 28 日
```

```
 * @author ALEX-CHUN-YU
```

```
 */
```

```
public class AirlineCompany {
```

```
    /**
```

```
     * Airline Company Name.
```

```
     */
```

```
    private String name = null;
```

```
    /**
```

```
     * Airline Company Flights.
```

```
     */
```

```
    private final Map<Integer, GenericFlight> flights = new  
    HashMap<Integer, GenericFlight>();
```

```
    /**
```

```
     * Constructor for the AirlineCompany class which takes the  
    default name.
```

```
     * @param name Airline Company Name.
```

```
     */
```

```
    public AirlineCompany(final String name) {  
        this.setName(name);
```

$$\}$$

/\*\*

**\* Set Airline Company Information About Airline Company Name.**

**\* @param name company name.**

*\*/*

```
private void setName(final String name) {
```

```
this.name = name;
```

$$\}$$

/\*\*

**\* Get Company Information About Company Name.**

\* @return Airline Company Name.

\*/

```
public String getName() {
```

```
return this.name;
```

$$\}$$

/\*\*\*

**\* Register a flight with the airline company.**

\* @param flight flight

\* *@*return number.

\*/

```
public Integer registerFlight(final GenericFlight flight) {
```

```
Integer num = flight.getFlightNumber();
```

**boolean flag;**

```
if (num == 0) {
```

```
while (true) {
```

```
flag = false;
```

```
for (Integer flightNumber : flights.keySet()) {
```

```
if (num.equals(flightNumber)) {
```

```
flag = true;
```

}

}

```
if (flag) {
```

```
num += 1;
```

} else {

```

        break;
    }
}
flight.setFlightNumber(num);
}
this.flights.put(num, flight);
return num;
}

/**
 * Unregister Flight With The Airline Company.
 * @param flightNumber Flight Number.
 */
public void unregisterFlight(final Integer flightNumber) {
    if (this.flights.containsKey(flightNumber) &&
this.flights.get(flightNumber) != null) {
        this.flights.get(flightNumber).setFlightNumber(null);
        this.flights.put(flightNumber, null);

        System.out.println("刪除成功!!");
    } else {
        System.out.println("沒有此航班資訊!");
    }
}

/**
 * Get All Registered Flights For This Airline Company.
 * @return Unmodifiable list of registered flights.
 */
public Map<Integer, GenericFlight> getFlights() {
    return Collections.unmodifiableMap(this.flights);
}
}

```

## Airport Class

flight\_information package

package flight\_information;

/\*\*

\* 存取機場資訊.

\* @version 1.0 2017 年 10 月 28 日

\* @author ALEX-CHUN-YU

\*/

public class Airport {

/\*\*

\* Airport City.

\*/

private City city = null;

/\*\*

\* Constructor About Airport Information.

\* @param city city

\*/

public Airport(City city) {

this.setCity(city);

}

/\*\*

\* Set Airport Information About City.

\* @param city city

\*/

private void setCity(City city) {

this.city = city;

}

/\*\*

\* Get Airport Information About City.

\* @return city city

\*/

public City getCity() {

```

        return this.city;
    }

    @Override
    public String toString() {
        return this.city.toString() + " Airport";
    }

    @Override
    public boolean equals(Object obj) {
        if (obj.getClass().equals(this.getClass())) {
            Airport airport = (Airport) obj;
            if (airport.getCity().equals(this.city)) {
                return true;
            }
        }
        return false;
    }

    @Override
    public int hashCode() {
        return super.hashCode();
    }
}

```

## City Class

flight\_information package

```
package flight_information;
```

```
/**
```

```
 * 存取城市資訊.
```

```
 * @version 1.0 2017 年 10 月 28 日
```

```
 * @author ALEX-CHUN-YU
```

```
 */
```

```
public class City {
```

```
    /**
```

```
     * City Name.
```

```
    */
```

```
    private String name = null;
```

```
    /**
```

```
     * Constructor About City Information.
```

```
     * @param name city name
```

```
    */
```

```
    public City(String name) {
```

```
        this.setName(name);
```

```
    }
```

```
    /**
```

```
     * Set City Information About City Name.
```

```
     * @param name city name
```

```
    */
```

```
    private void setName(String name) {
```

```
        this.name = name;
```

```
    }
```

```
    /**
```

```
     * Get City Information About City Name.
```

```
     * @return city name
```

```
    */
```

```
    public String getName() {
```



```

        return this.name;
    }

    @Override
    public String toString() {
        return this.name;
    }

    @Override
    public boolean equals(Object obj) {
        if (obj.getClass().equals(this.getClass())) {
            City city = (City) obj;
            if (city.getName().equals(this.name)) {
                return true;
            }
        }
        return false;
    }

    @Override
    public int hashCode() {
        return super.hashCode();
    }
}

```

## Flight Class

flight\_information package

```
package flight_information;

import booking_information.Booking;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.List;

/**
 * 主要負責預定機票之類別.
 */
public class Flight extends GenericFlight {
    /**
     * Booking Identification.
     */
    private String bookingId = null;

    /**
     * Booking List.
     */
    private final List<Booking> bookings = new ArrayList<Booking>();

    /**
     * Constructor About Flight Information.
     * @param validityPeriod validity period
     * @param departureTime departure time
     * @param arrivalTime arrival time
     * @param departureAirport departure airport
     * @param arrivalAirport arrival airport
     */
    public Flight(final long validityPeriod,
                  final Date departureTime, final Date arrivalTime,
                  final Airport departureAirport, final Airport
arrivalAirport) {
```

```

        super(validityPeriod, departureTime, arrivalTime,
departureAirport, arrivalAirport);
    }

    /**
     * Open Booking On The Flight.
     * @param booking flight booking.
     */
    public void openBooking(final Booking booking) {
        // Set booking number
        this.bookings.add(booking);
    }

    /**
     * Close Booking For The Flight.
     * @param booking flight booking.
     */
    public void closeBooking(final Booking booking) {
        this.bookings.remove(booking);
    }

    /**
     * Get Flight Information About Flight Bookings.
     * @return booking result
     */
    public List<Booking> getBookings() {
        return Collections.unmodifiableList(this.bookings);
    }
}

```

## GenericFlight Class

flight\_information package

```
package flight_information;
```

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.List;
```

```
/**
```

- \* 此為通用於飛機航班抽象類別.
  - \* 飛機號碼、有效期限、出發時間、抵達時間
  - \* 出發地點、抵達地點、飛行時間、中途停留
  - \* @version 1.0 2017 年 10 月 28 日
  - \* @author ALEX-CHUN-YU
- ```
*/
```

```
public abstract class GenericFlight {
    /**
     * Flight Number.
     */
    private Integer flightNumber = 0;

    /**
     * Validity Period.
     */
    private long validityPeriod = 0;

    /**
     * Departure Time.
     */
    private Date departureTime = null;

    /**
```

```

    * Arrival Time.
    */
    private Date arrivalTime = null;

    /**
     * Departure Airport.
     */
    private Airport departureAirport = null;

    /**
     * Arrival Airport.
     */
    private Airport arrivalAirport = null;

    /**
     * Airport Stopover.
     */
    private final List<Stopover> stopovers = new
    ArrayList<Stopover>();

    /**
     * Constructor About Generic Flight Information.
     * @param validityPeriod validity period
     * @param departureTime departure time
     * @param arrivalTime arrival time
     * @param departureAirport departure airport
     * @param arrivalAirport arrival airport
     */
    public GenericFlight(final long validityPeriod,
                        final Date departureTime, final Date
arrivalTime,
                        final Airport departureAirport, final
Airport arrivalAirport) {
        this.setValidityPeriod(validityPeriod);
        this.setDepartureTime(departureTime);
        this.setArrivalTime(arrivalTime);
        this.setDepartureAirport(departureAirport);
        this.setArrivalAirport(arrivalAirport);
    }

```

```
}
```

```
/**
```

```
 * Set Generic Flight Information About Flight Number.
```

```
 * @param flightNumber flight number
```

```
 */
```

```
public void setFlightNumber(Integer flightNumber) {
```

```
    this.flightNumber = flightNumber;
```

```
}
```

```
/**
```

```
 * Get Generic Flight Information About Flight Number.
```

```
 * @return flightNumber flight number
```

```
 */
```

```
public Integer getFlightNumber() {
```

```
    return this.flightNumber;
```

```
}
```

```
/**
```

```
 * Set Generic Flight Information About Validity Period.
```

```
 * @param validityPeriod validity period
```

```
 */
```

```
private void setValidityPeriod(long validityPeriod) {
```

```
    this.validityPeriod = validityPeriod;
```

```
}
```

```
/**
```

```
 * Get Generic Flight Information About Validity Period.
```

```
 * @return validity period
```

```
 */
```

```
public long getValidityPeriod() {
```

```
    return this.validityPeriod;
```

```
}
```

```
/**
```

```
 * Set Generic Flight Information About Departure Time.
```

```
 * @param departureTime departure time
```

```
 */
```

```

private void setDepartureTime(Date departureTime) {
    this.departureTime = departureTime;
}

/**
 * Get Generic Flight Information About Departure Time.
 * @return departure time
 */
public Date getDepartureTime() {
    return this.departureTime;
}

/**
 * Set Generic Flight Information About Arrival Time.
 * @param arrivalTime arrival time
 */
private void setArrivalTime(Date arrivalTime) {
    if (arrivalTime != null) {
        if ((this.getDepartureTime() == null)
            || arrivalTime.after(this.getDepartureTime())) {
            this.arrivalTime = arrivalTime;
        } else {
            throw new NullPointerException();
        }
    } else {
        throw new NullPointerException();
    }
}

/**
 * Get Generic Flight Information About Arrival Time.
 * @return arrival time
 */
public Date getArrivalTime() {
    return this.arrivalTime;
}

/**

```

```

    * Set Generic Flight Information About Departure Airport.
    * @param departureAirport departure airport
    */
private void setDepartureAirport(Airport departureAirport) {
    this.departureAirport = departureAirport;
}

/**
 * Get Generic Flight Information About Departure Airport.
 * @return departure airport
 */
public Airport getDepartureAirport() {
    return this.departureAirport;
}

/**
 * Set Generic Flight Information About Arrival Airport.
 * @param arrivalAirport arrival airport
 */
private void setArrivalAirport(Airport arrivalAirport) {
    this.arrivalAirport = arrivalAirport;
}

/**
 * Get Generic Flight Information About Arrival Airport.
 * @return arrival airport
 */
public Airport getArrivalAirport() {
    return this.arrivalAirport;
}

/**
 * Get Stopover Information About Flight Time.
 * @return Flight length.
 */
public long getFlightTime() {
    if ((this.getArrivalTime() != null)
        && (this.getDepartureTime() != null)) {

```



```

        return (this.getArrivalTime().getTime()
                - this.getDepartureTime().getTime())
                / Integer.parseInt("3600000");
    } else {
        throw new NullPointerException();
    }
}

/**
 * Add a stopover to the flight. The stopover must be within the time
of the
 * entire flight.
 * @param stopover stopover
 */
public void addStopover(final Stopover stopover) {
    if (((stopover.getDepartureTime() != null) && (stopover
        .getDepartureTime().getTime() <
this.getArrivalTime().getTime()))
        && ((stopover.getArrivalTime() != null) &&
(stopover
        .getArrivalTime().getTime() >
this.getDepartureTime()
        .getTime())))) {
        this.stopovers.add(stopover);

    } else {
        throw new IllegalArgumentException();
    }
}

/**
 * Remove A Stopover Based On Its Values.
 * @param stopover stopover to remove.
 */
public void removeStopover(final Stopover stopover) {
    this.stopovers.remove(stopover);
}

```

```

/**
 * Getter For A Read-Only Version Of The Stopovers List.
 * @return stopovers list.
 */
public List<Stopover> getStopovers() {
    return Collections.unmodifiableList(this.stopovers);
}

@Override
public String toString() {
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm");

    return "出發時間:" + sdf.format(this.departureTime)

        + " 抵達時間:" + sdf.format(this.arrivalTime)

        + "\n 出發地點:" + this.departureAirport

        + " 抵達地點:" + this.arrivalAirport

        + "\n 總花費時間:" + this.getFlightTime() + "小時

\n";
}
}

```

## Stopover Class

flight\_information package

```
package flight_information;

import java.text.SimpleDateFormat;
import java.util.Date;

/**
 * 存取中途停留機場以及時間.
 *
 * @version 1.0 2017 年 10 月 29 日
 * @author ALEX-CHUN-YU
 */
public class Stopover {
    /**
     * Departure Time.
     */
    private Date departureTime = null;

    /**
     * Arrival Time.
     */
    private Date arrivalTime = null;

    /**
     * Airport Stopover .
     */
    private Airport airport = null;

    /**
     * Constructor About Stopover Information.
     * @param departureTime departureTime
     * @param arrivalTime arrivalTime
     * @param airport airport
     */
    public Stopover(final Date departureTime, final Date arrivalTime,
final Airport airport) {
```

```

        this.setDepartureTime(departureTime);
        this.setArrivalTime(arrivalTime);
        this.setAirport(airport);
    }

    /**
     * Set Stopover Information About Departure Time.
     * @param departureTime departure time.
     */
    private void setDepartureTime(final Date departureTime) {
        if (departureTime != null) {
            if ((this.getArrivalTime() == null)
                || departureTime.before(this.getArrivalTime())) {
                this.departureTime = departureTime;
            } else {
                throw new NullPointerException();
            }
        } else {
            throw new NullPointerException();
        }
    }

    /**
     * Get Stopover Information About Departure Time.
     * @return departure time.
     */
    public Date getDepartureTime() {
        return this.departureTime;
    }

    /**
     * Set Stopover Information About Arrival Time.
     * @param arrivalTime arrival time.
     */
    private void setArrivalTime(final Date arrivalTime) {
        // Check if the new the departure time is not null;
        if (arrivalTime != null) {
            if ((this.getDepartureTime() == null)

```

```

        || arrivalTime.after(this.getDepartureTime())) {
            this.arrivalTime = arrivalTime;
        } else {
            throw new NullPointerException();
        }
    } else {
        throw new NullPointerException();
    }
}

```

```

/**
 * Get Stopover Information About Arrival Time.
 * @return arrival time.
 */
public Date getArrivalTime() {
    return this.arrivalTime;
}

```

```

/**
 * Set Stopover Information About Airport Stopover.
 * @param airport stop airport
 */
private void setAirport(final Airport airport) {
    this.airport = airport;
}

```

```

/**
 * Get Stopover Information About Airport Stopover.
 * @return airport.
 */
public Airport getAirport() {
    return this.airport;
}

```

```

/**
 * Get Generic Flight Information About Flight Time.
 * @return Flight length.
 */

```

```

public long getStopTime() {
    if ((this.getArrivalTime() != null)
        && (this.getDepartureTime() != null)) {
        return (this.getArrivalTime().getTime()
            - this.getDepartureTime().getTime())
            / Integer.parseInt("3600000");
    } else {
        throw new NullPointerException();
    }
}

@Override
public String toString() {
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-
dd HH:mm");

    return "轉機的機場:" + this.airport + " | 抵達時間:" +
sdf.format(this.departureTime)
        + " | 出發時間:" + sdf.format(this.arrivalTime) + " |
休息時間:"
        + this.getStopTime() + "小時";
}

@Override
public boolean equals(final Object obj) {
    if (obj.getClass().equals(this.getClass())) {
        Stopover other = (Stopover) obj;
        return (other.getDepartureTime() ==
this.getDepartureTime())
            && (other.getArrivalTime() ==
this.getArrivalTime())
            && (other.getAirport() == this.getAirport());
    } else {
        return false;
    }
}

```

```
}

@Override
public int hashCode() {
    return super.hashCode();
}
}
```

# MysqlDatabaseController

database package

## Class

```
package database;

import java.io.FileInputStream;
import java.io.IOException;
import java.sql.*;
import java.util.Properties;

/**
 * Database Controller.
 *
 * @version 1.0 2017 年 10 月 23 日
 * @author Alex
 */
public class MysqlDatabaseController {
    /**
     * Database Host.
     */
    private String dbHost;

    /**
     * Database Name.
     */
    private String dbName;

    /**
     * Database User Name.
     */
    private String userName;

    /**
     * Database User Password.
     */
}
```



```

private String password;

/**
 * Database Connection.
 */
private Connection connection;

/**
 * Constructor.
 */
public MysqlDatabaseController() {
    initLoadJdbc();
    initConnection();
}

/**
 * Database Connection Configuration.
 */
private void initLoadJdbc() {
    Properties properties = new Properties();
    try {
        properties.load(new FileInputStream(
"src/main/resources/databaseConfiguration.properties"));
        dbHost = properties.getProperty("dbHost");
        dbName = properties.getProperty("dbName");
        userName = properties.getProperty("userName");
        password = properties.getProperty("password");
    } catch (IOException e) {
        e.printStackTrace();
    }
    try {
        Class.forName("com.mysql.cj.jdbc.Driver");
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
}

```

```

/**
 * Connect DB.
 */
private void initConnection() {
    try {
        connection = DriverManager.getConnection(dbHost +
dbName
        + "?useUnicode=true&characterEncoding=utf8"
        + "&useJDBCCompliantTimezoneShift=true"
        +
"&useLegacyDatetimeCode=false&serverTimezone=UTC", userName,
password);

        System.out.println("以連結至:" + dbHost + ":" + dbName

+ ":" + userName + ":" + password + "資料庫。");

        //statement=connection.createStatement();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

/**
 * INSERT INTO 資料表 (Column Name...) VALUES (Column
Values...).
 * @param tableName table name
 * @param obj sql obj
 */
public void execInsert(String tableName, SqlObject obj) {
    String sql="INSERT INTO " + tableName + " (" +
obj.getColumnNameString() + ")"
        + " VALUES (" + obj.getColumnValueString() + ");";
    try {
        PreparedStatement ps = connection.prepareStatement(sql);
        ps.execute();
    } catch (SQLException e) {
        System.out.println("The data has been loaded into db " +

```

```

tableName + " table.");
    }
}

/**
 * DELETE FROM 哪些資料表 WHERE 限制的條件.
 * @param tableName table name
 * @param condition condition
 */
public void execDelete(String tableName, String condition) {
    String sql = "DELETE FROM " + tableName + " WHERE " +
condition;
    try {
        PreparedStatement ps = connection.prepareStatement(sql);
        ps.execute();
    } catch (SQLException e) {
        System.out.println("The data has been loaded into db " +
tableName + " table.");
    }
}

/**
 * UPDATE Table SET Column = [value] WHERE Condition.
 * @param tableName table name
 * @param obj obj
 * @param condition condition
 */
public void execUpdate(String tableName, SqlObject obj, String
condition) {
    String sql = "UPDATE " + tableName + " SET " +
obj.getColumnNameValuePairString() + " " + condition + ";";
    try {
        PreparedStatement preparedStatement =
connection.prepareStatement(sql);
        preparedStatement.execute();
    } catch (SQLException e) {
        System.out.println("The data has been loaded into db " +

```

```
tableName + " table.");
```

```
}
```

```
}
```

```
/**
```

```
 * SELECT 所需求之欄位 FROM 哪些資料表 WHERE 限制
```

的條件.

```
 * @param column columns
```

```
 * @param tableName table name
```

```
 * @param condition condition
```

```
 * @return result set 存取所抓取之欄位
```

```
 */
```

```
public ResultSet execSelect(String column, String tableName, String  
condition) {
```

```
    String sql = "SELECT " + column + " FROM " + tableName +  
" WHERE " + condition;
```

```
    ResultSet resultSet = null;
```

```
    try {
```

```
        Statement statement = connection.createStatement();
```

```
        resultSet = statement.executeQuery(sql);
```

```
    } catch (SQLException e) {
```

```
        e.printStackTrace();
```

```
    }
```

```
    return resultSet;
```

```
}
```

```
}
```

## SqlObject Class

database package

```
package database;
```

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

```
/**
```

```
 * 根據不同 SQL 語法所設計出來的 SQL 物件.
```

```
 *
```

```
 * @version 1.0 2017 年 11 月 01 日
```

```
 * @author Alex
```

```
 *
```

```
 */
```

```
public class SqlObject {
```

```
    /**
```

```
     * Sql Object Size.
```

```
     */
```

```
    private int size;
```

```
    /**
```

```
     * Sql Object Column Name.
```

```
     */
```

```
    private ArrayList<String> columnName;
```

```
    /**
```

```
     * Sql Object Column Value.
```

```
     */
```

```
    private ArrayList<Object> columnValue;
```

```
    /**
```

```
     * Constructor.
```

```
     */
```

```
    public SqlObject() {
```

```
        columnName = new ArrayList<String>();
```

```
        columnValue = new ArrayList<Object>();
```

```
        size = 0;
```

```
}
```

```
/**
```

```
 * Add Sql Object (Format : Column, Value).
```

```
 * @param column column
```

```
 * @param value value
```

```
 */
```

```
public void addSqlObject(String column, Object value) {  
    if (columnName.contains(column)) {  
        columnValue.add(columnName.indexOf(column), value);  
        columnValue.remove(columnName.indexOf(column) + 1);  
    } else {  
        columnName.add(column);  
        columnValue.add(value);  
        size = columnName.size();  
    }  
}
```

```
/**
```

```
 * Get Sql Object Size.
```

```
 * @return size
```

```
 */
```

```
public int getSize() {  
    return size;  
}
```

```
/**
```

```
 * Get Sql Object column Name index of your number.
```

```
 * @param i your number
```

```
 * @return size
```

```
 */
```

```
public String getColumnNameIndexOf(int i) {  
    return columnName.get(i);  
}
```

```
/**
```

```
 * Get Every Column Name.
```

```
 * @param column column
```

```

    * @return value
    */
    public int getColumnNameIndex(String column) {
        if (columnName.contains(column)) {
            return columnName.lastIndexOf(column);
        } else {
            return -1;
        }
    }

    /**
     * Get Sql Object column Value index of your number.
     * @param i your number
     * @return value
     */
    public Object getColumnValueIndexOf(int i) {
        if (columnName.size() >= i) {
            return columnValue.get(i);
        }
        return null;
    }

    /**
     * Get All Column Name For Data Insert.
     * @return column String
     */
    public String getColumnNameString() {
        StringBuilder columnString = new StringBuilder();
        if (size != 0) {
            columnString.append(columnName.get(0));
            for (int i = 1; i < size; i++) {
                columnString.append(",");
                columnString.append(columnName.get(i));
            }
        }
        return columnString.toString();
    }
}

```

```

/**
 * Get All Column Value For Data Insert.
 * @return column Prepared Statement String
 */
public String getColumnValueString() {
    StringBuilder columnPreparedStatementString = new
StringBuilder();
    if (size != 0) {

columnPreparedStatementString.append(toSqlValue(columnValue.get(0))
);
        for (int i = 1; i < size; i++) {
            columnPreparedStatementString.append(",");

columnPreparedStatementString.append(toSqlValue(columnValue.get(i)))
;
        }
    }
    return columnPreparedStatementString.toString();
}

/**
 * Get Column Name Value Pair String For Data Update.
 * @return column Prepared Statement String
 */
public String getColumnNameValuePairString() {
    StringBuilder columnPreparedStatementString = new
StringBuilder();
    if (size != 0) {

columnPreparedStatementString.append(columnName.get(0));
        columnPreparedStatementString.append("=");

columnPreparedStatementString.append(toSqlValue(columnValue.get(0))
);
        for (int i = 1; i < size; i++) {
            columnPreparedStatementString.append(",");

```



```
columnPreparedStatementString.append(columnName.get(i));
        columnPreparedStatementString.append("=");
```

```
columnPreparedStatementString.append(columnValue.get(i));
    }
}
return columnPreparedStatementString.toString();
}
```

```
/**
```

```
 * 如果為 String 根據 sql 語法必須加上單引號.
```

```
 * @param obj obj string object or integer object
```

```
 * @return string
```

```
 */
```

```
private String toSqlValue(Object obj) {
    if (obj.getClass() == Integer.class) {
        return obj.toString();
    } else if (obj.getClass() == String.class) {
        return "\"" + obj.toString() + "\"";
    } else {
        return null;
    }
}
```

```
}
}
```

## ConstantField Class

constant\_field package

```
package constant_field;
```

```
/**
```

```
 * Constant Field.
```

```
 *
```

```
 * @version 1.0 2017 年 11 月 01 日
```

```
 * @author Alex
```

```
 *
```

```
 */
```

```
public class ConstantField {
```

```
    public final static String AIRLINE_COMPANY_TABLE =  
"airline_company";
```

```
    public final static String NAME = "name";
```

```
    public final static String FLIGHT_NUMBER = "flight_number";
```

```
    public final static String DEPARTURE_AIRPORT =  
"departure_airport";
```

```
    public final static String DEPARTURE_DATE = "departure_date";
```

```
    public final static String ARRIVAL_AIRPORT = "arrival_airport";
```

```
    public final static String ARRIVAL_DATE = "arrival_date";
```

```
    public final static String VALIDITY_PERIOD = "validity_period";
```

```
    public final static String CUSTOMER_BOOKING_ID_TABLE =  
"customer_flight_booking";
```

```
    public final static String BOOKING_ID = "booking_id";
```

```
    public final static String CUSTOMER_NAME = "customer_name";
```

```
    public final static String CUSTOMER_FOREIGN_NAME =  
"customer_foreign_name";
```

```
    public final static String CUSTOMER_ADDRESS =  
"customer_address";
```

```
    public final static String CUSTOMER_PHONE_NUMBER =  
"customer_phone_number";
```

```
    public final static String PASSENGERS_TABLE = "passengers";
```

```
    public final static String PASSENGER_NAME =  
"passenger_name";
```

```
    public final static String PASSENGER_FOREIGN_NAME =  
"passenger_foreign_name";}
```