INSTRUCTIONS:

Environment Setup:

Computer information:

Cpu: intel core i7-6700HQ

Operating system: windows 10 professional

Development Environment:

Eclipse IDE for Java Developers

Version: 2018-09 (4.9.0)

Build id: 20180917-1800

OS: Windows 10, v.10.0, x86_64 / win32

Java version: 1.8.0_131

Visual Studio 2017

Backend folder:

airLineServer : please import to eclipse implement using java socket and threading in eclipse

Frontend Folder:

airlineclientside: import into VS2017 and implement using winform and C#

Java information:(very important, please use java version 1.8+)

java version "1.8.0_131"

Java(TM) SE Runtime Environment (build 1.8.0_131-b11)

Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)

Database information:

In the server folder, there is a database: airline_db.accdb, I use Access 2010, there is a folder in the attachment:**" libs for server to connect db"** Under this folder, there are all of the java jars for connecting Access database using JAVA.

One thing to note is: in Java 8+ , it doesn't support jdbc-odbc bridge now, so I can use and old java native method to connect, instead I have to use that third party library.

When you import the java server project into the eclipse, you can import the jar files too.

Another way I do is

Put all of the jars into your windows java install directory

For example, in my computer my java installed in:

E:\java

Under this directory there are two folder: jdk and jre

So you should put all of the jars into jre/lib/ext

In my computer is:

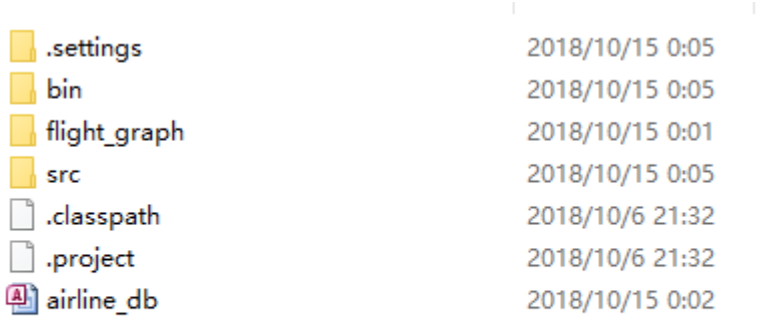
E:\java\jre\lib\ext

After you done this , you don't need to import jars in eclipse any more

PROJECT STRUCTURE:

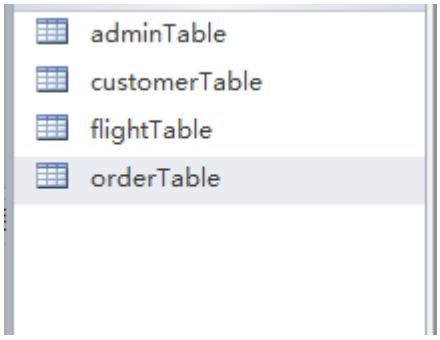
In order to make things easy

First talk about the server part:



So src folder just store all of the server souce file: .java

ariline_db is the Access database I mentioned, it has 4 tables:



adminTable: store the admin information

customerTable:store all of information for customer: and customer login the system using realID as the password, so do not be confused when you see the table item, when login, use username and realID to fill in the form

flightTable:store the information of flights

orderTable: store customer's booking for a flight along with seat type and seat number

There is also a folder: flight_graph

This is a tricky one, and this one is a folder of all of the xml files for each flight

When a admin add the flight information ,it will generate a xml file

Name like: (could change)flightID.xml.

The format of each xml file is like

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<flight>
    <firstclass>
        <seat Number="1">1</seat>
    </firstclass>
    <ecoclass>
        <seat Number="2">1</seat>
        <seat Number="3">0</seat>
    </ecoclass>
</flight>
```

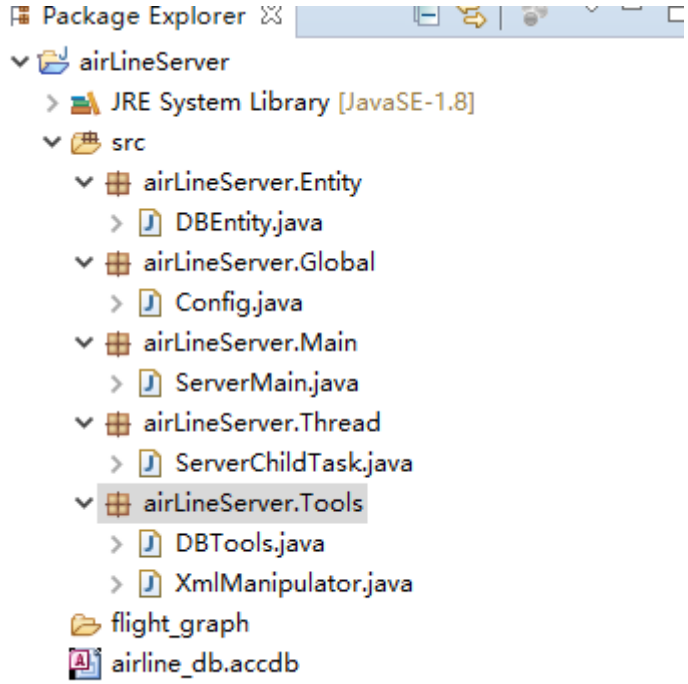
It actually reflect the current plane seat situation for each flight

The attribute for seat Number is the seat number and the value of them is 1 or 0, 1 represent the seat already been taken by others while 0 means still empty

This is the data client get when they choose the seat and make order

The server program work flow:

For the server I use java socket and thread pool to implement



ServerMain is the entry point for whole program , the main thread just wait for client socket and when it comes a client socket, it will put into a threadpool and run ServerChildTask

In ServerChildTask, it will first parse the information(request) from client and create a DBEntity for business logic.

For DBEntity there is a method which is the core of back end logic. It take a great part of manipulate DB and XML and get result for user

Then after the result is obtained, the ServerChildTask will send back to client

The format of request information or the protocol is

Role: admin/customer \t

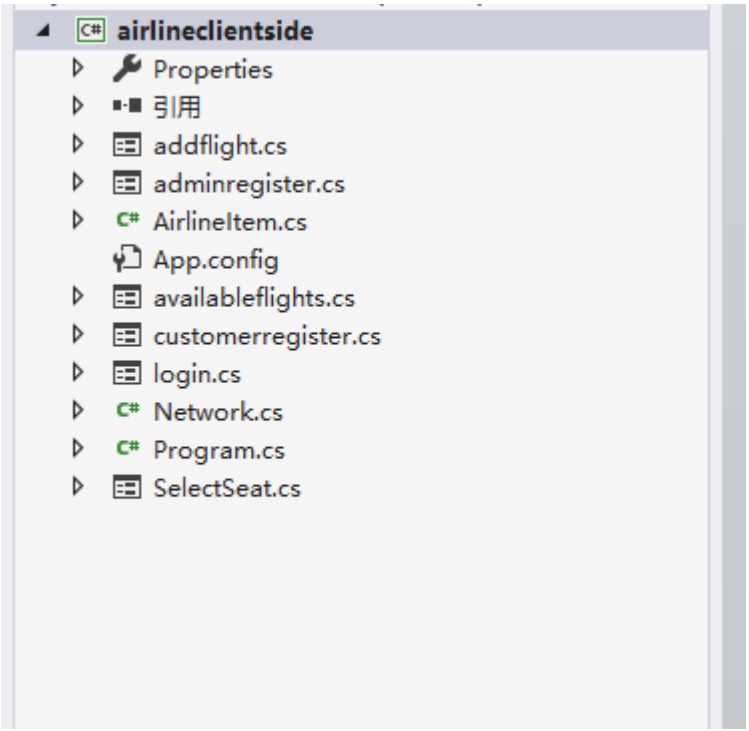
Action:……many different actions \t

Content lines divided by \t, they are the request paramaters

A simulation of http protocol

Client part:

For the client part



Basically there are 6 winforms:

The first one you will meet is login.cs, it has two button:

One is register another is login

For the new user, first you need to register either an admin or customer

After registered, the action flow is different for customer and admin:

For admin:

Login then add flight information

For customer:

Login then you can see available flights

Then you can choose one in the listview and press select seat button

After that you can select the seat and make order

The rule I made is every customer can only buy one seat.

This is the end of the document, because I work full-time now, and this program is finished in 3 days, and I just have little experience using c# before, so I learned that winform and c# while writing codes

There must be a lot of bugs that I have no time to fix, I will continue when I have spare time

And also I am looking forward to hear from you about anything you confused

Thank you very much Chen Xu