School Dance Ticket Software

Table of Contents

Completion Check

❓ Methods (add() recursive method)
❓ Loops (insert() while loop,)
❓ Decision structures

1. allow student buy early bird tickets for cheap price;
2. allow student buy more than one tickets for guests;
3. allow student buy door price tickets;
4. allow student refund his/her tickets;
5. allow display all students information on a table in the order of student id;
6. if student does NOT exist while refund, the message will be displayed on status bar at the bottom.
7. main window shall display party start time, duration, location and sponsor information
8. it is better to have help document for user.
9. Basic operation check: add, save, load, insert, remove ...
10. ❓💡 What is the difference between add() and insert() method in BinaryTree?

✔ UML - class, attributes, method... 👇See below
❓ Original ideas and design complexity
✔ Data Structure - applicable, Generic [See Node.java and BinaryTree.java]
✔ Functionality - Ticket(early bird, door price, refund), Find(one.update, all), Help

✔ Use of Tree data structure - BinaryTree, Node

✔ Naming conventions (Camel style, upercase for class, lower case for fields and methods)

🔨 Comments/Docs

✔🔨 Code readability (fields and methods naming, function single responsibility)

✔ User Interface (MainFram, MainPanel, DetailDialog, RefundDialog, ...)

❓ UML Layout/presentation

✔ File I/O (see Student.save(), Student.saveAll(), Student.loadAll())

✔❓ Data Structure (ArrayList, LinkedList, Queue) 😢May not be used🚧

✔❓ Basic operations (implemented size, contains, add, insert, find, but only used add, find)😢May not be used🚧

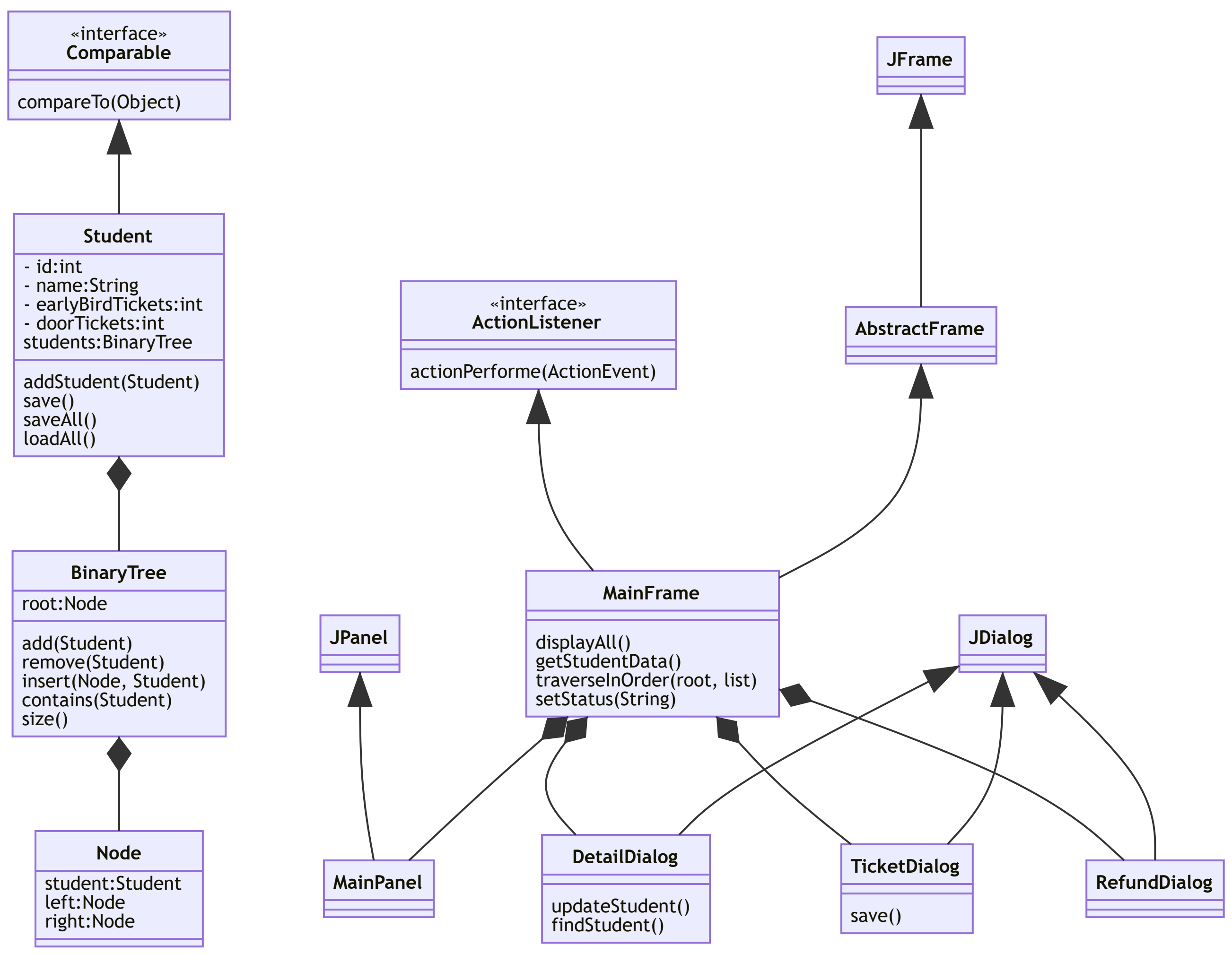
✔ Advanced operation - remove() (BinaryTree.remove()==>refund ticket)

✔ Use of a Tree data structure (BinaryTree.java)

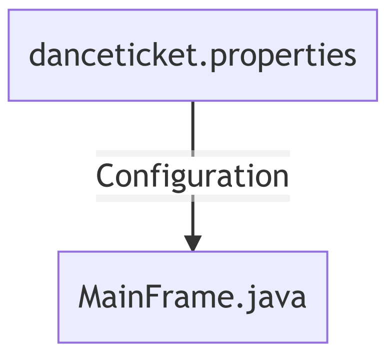
✔ Use file I/O to load/save configuration (danceticket.properties) and data (tickets.csv)

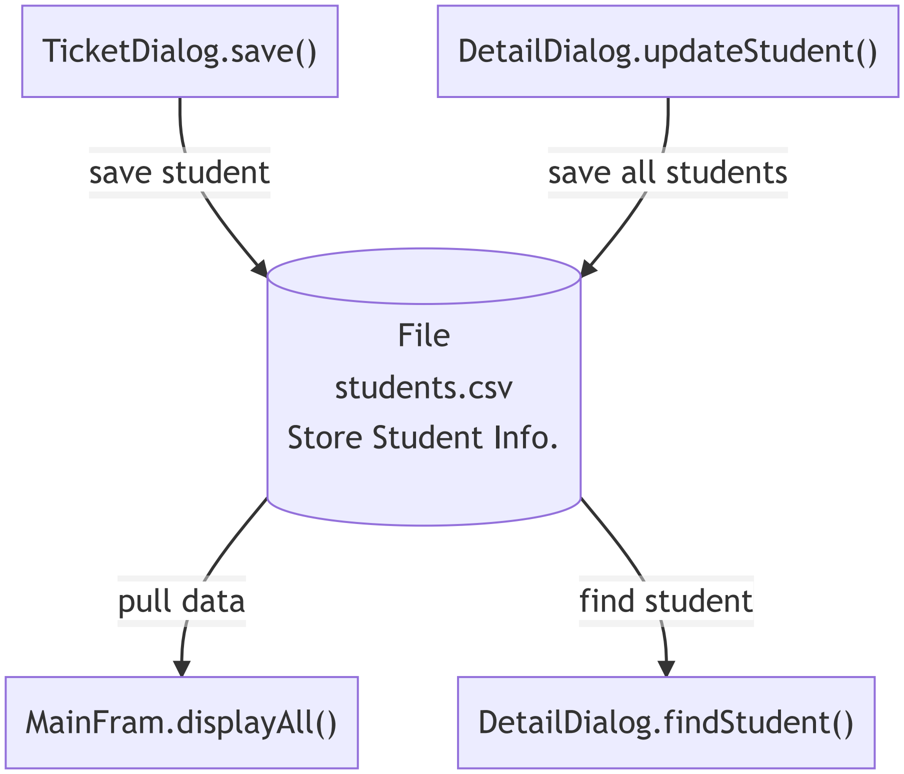
✔ must incorporate your own data structure(s) (BinaryTree, Node, BinaryTree.find, BinaryTree.remove)

Class Diagram



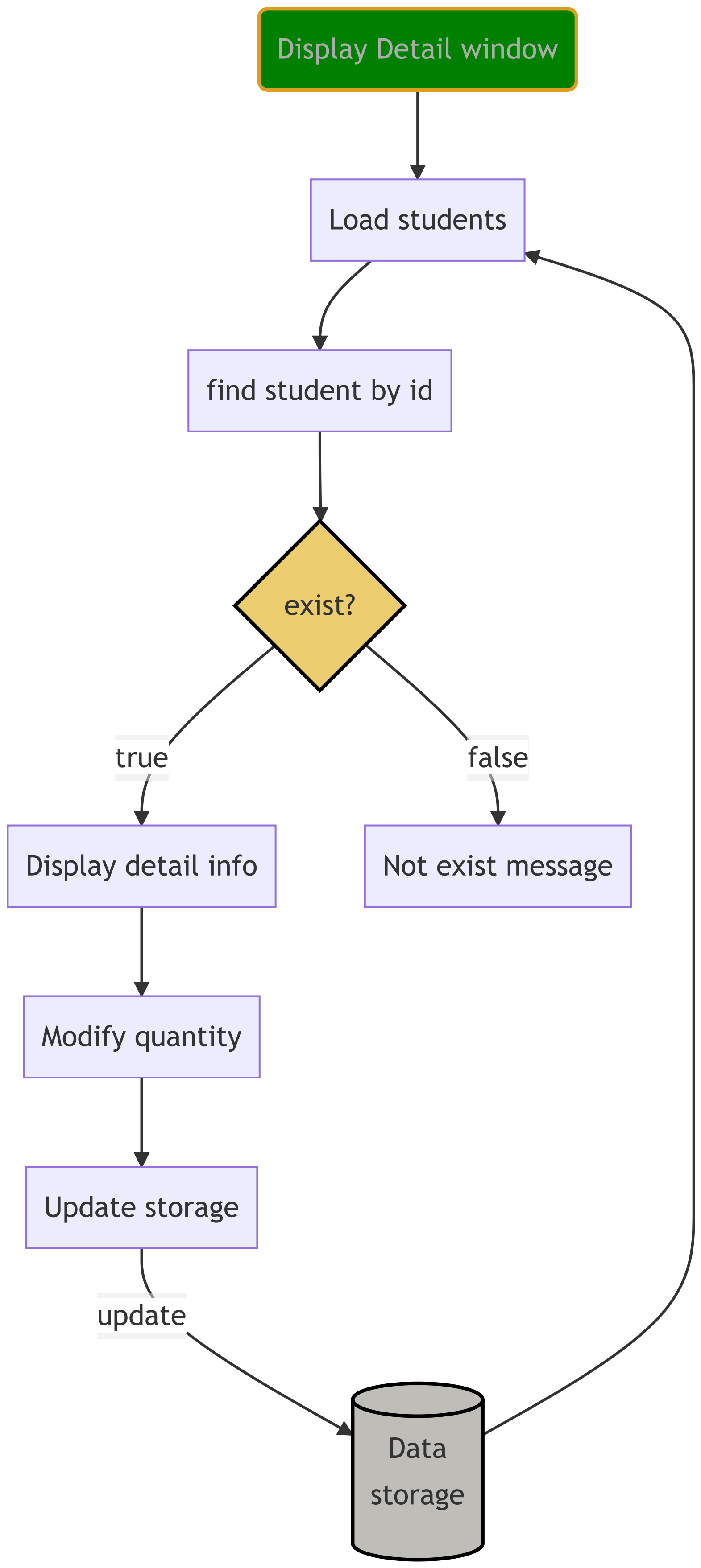
Configuration & Data store files



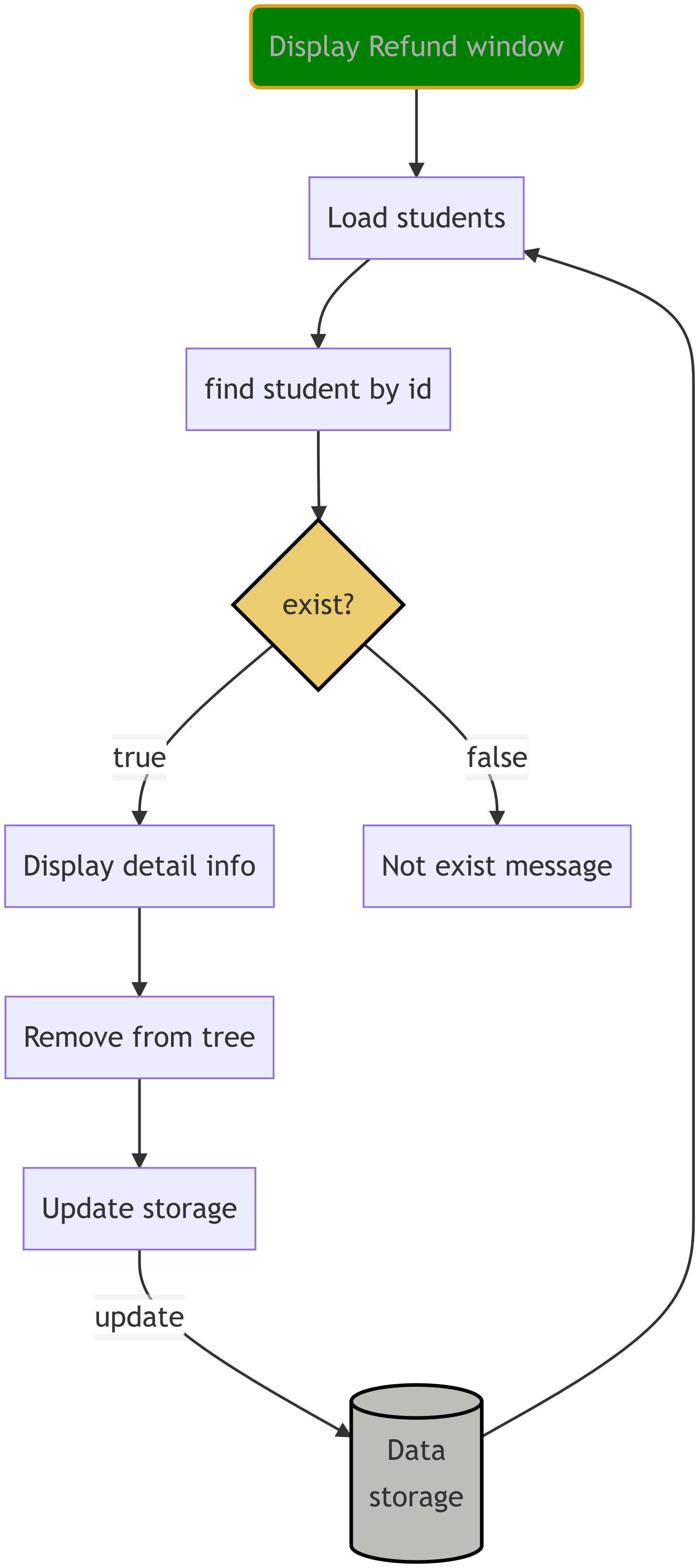


Flowchart

- Detail Operation Logic



- Refund Operation Logic


Display Refund window
Load students
find student by id
exist?
true
false
Display detail info
Not exist message
Remove from tree
Update storage
update
Data storage

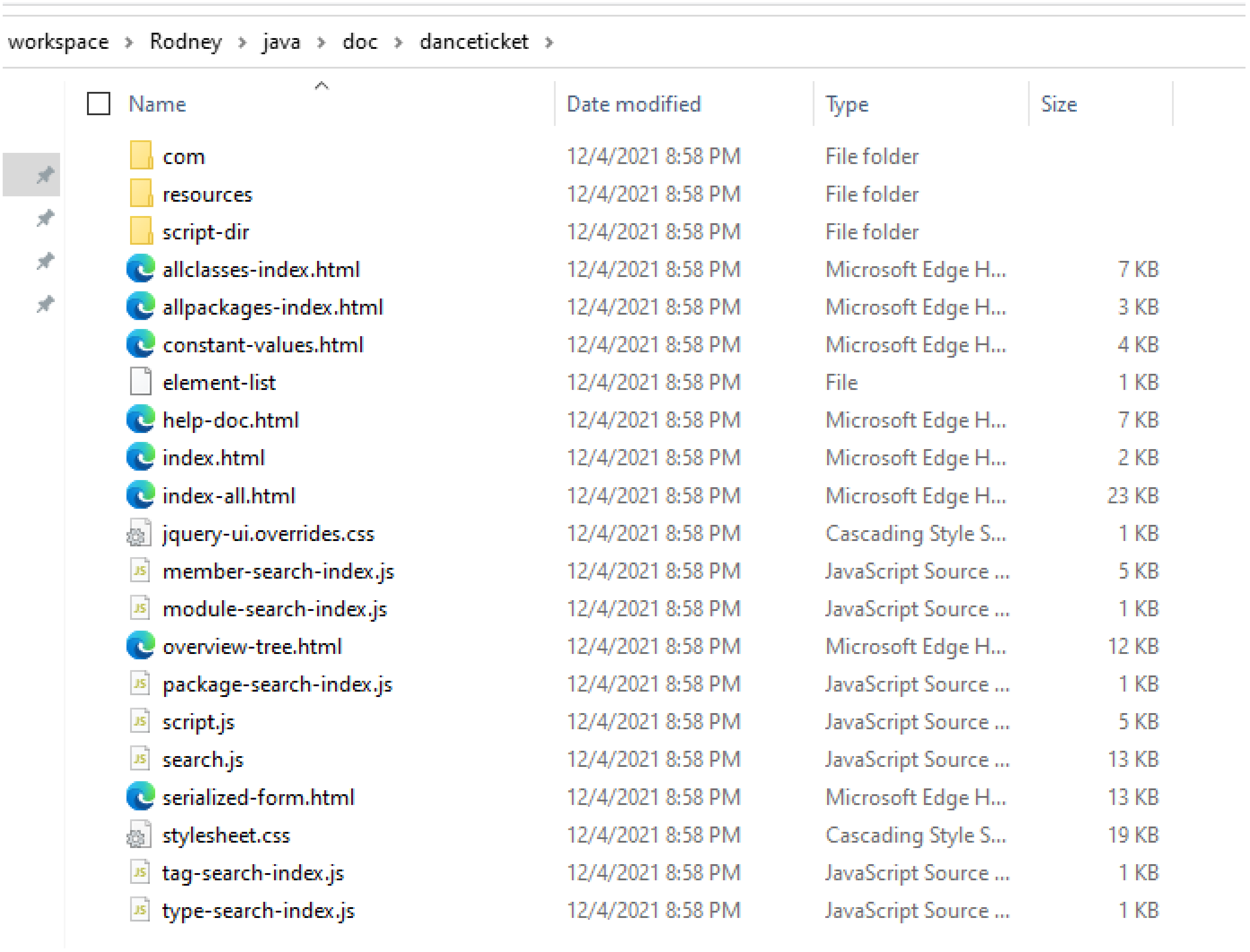
Test

test BinaryTree add(), loadAll(), saveAll(), Find(), Insert(), Remove()

also test configuration properties load(), Date format.

Java Docs

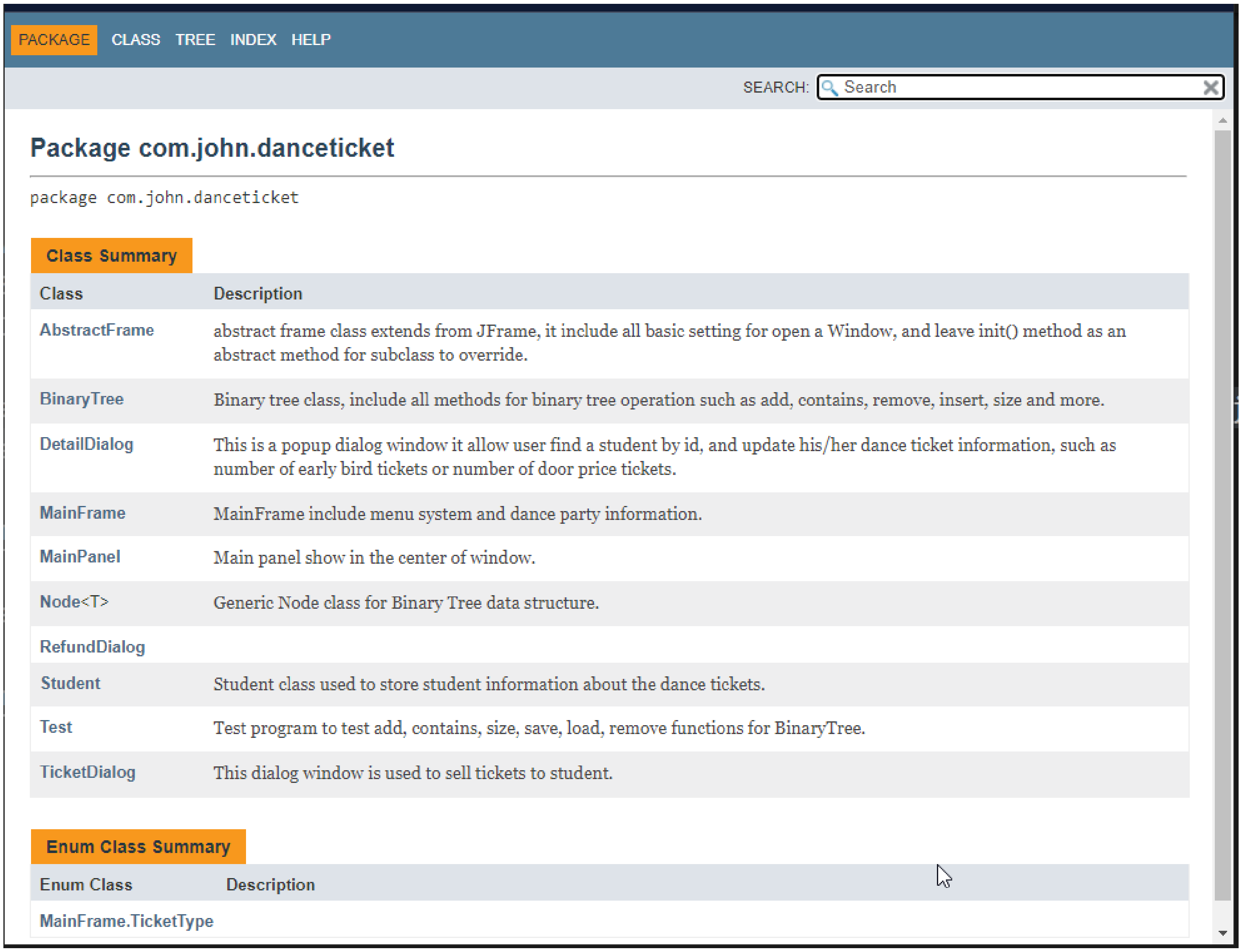
```
Open DOS command window, cd to the folder below, type in javadoc command

C:\Users\12818\workspace\Rodney\java\doc\danceticket>javadoc -sourcepath ../../john/src -subpackages com.john.dar
```



workspace › Rodney › java › doc › danceticket ›

Name	Date modified	Type	Size
com	12/4/2021 8:58 PM	File folder	
resources	12/4/2021 8:58 PM	File folder	
script-dir	12/4/2021 8:58 PM	File folder	
allclasses-index.html	12/4/2021 8:58 PM	Microsoft Edge H...	7 KB
allpackages-index.html	12/4/2021 8:58 PM	Microsoft Edge H...	3 KB
constant-values.html	12/4/2021 8:58 PM	Microsoft Edge H...	4 KB
element-list	12/4/2021 8:58 PM	File	1 KB
help-doc.html	12/4/2021 8:58 PM	Microsoft Edge H...	7 KB
index.html	12/4/2021 8:58 PM	Microsoft Edge H...	2 KB
index-all.html	12/4/2021 8:58 PM	Microsoft Edge H...	23 KB
jquery-ui.overrides.css	12/4/2021 8:58 PM	Cascading Style S...	1 KB
member-search-index.js	12/4/2021 8:58 PM	JavaScript Source ...	5 KB
module-search-index.js	12/4/2021 8:58 PM	JavaScript Source ...	1 KB
overview-tree.html	12/4/2021 8:58 PM	Microsoft Edge H...	12 KB
package-search-index.js	12/4/2021 8:58 PM	JavaScript Source ...	1 KB
script.js	12/4/2021 8:58 PM	JavaScript Source ...	5 KB
search.js	12/4/2021 8:58 PM	JavaScript Source ...	13 KB
serialized-form.html	12/4/2021 8:58 PM	Microsoft Edge H...	13 KB
stylesheet.css	12/4/2021 8:58 PM	Cascading Style S...	19 KB
tag-search-index.js	12/4/2021 8:58 PM	JavaScript Source ...	1 KB
type-search-index.js	12/4/2021 8:58 PM	JavaScript Source ...	1 KB

Double click index.html file name



Package com.john.danceticket

`package com.john.danceticket`

Class Summary

Class	Description
AbstractFrame	abstract frame class extends from JFrame, it include all basic setting for open a Window, and leave init() method as an abstract method for subclass to override.
BinaryTree	Binary tree class, include all methods for binary tree operation such as add, contains, remove, insert, size and more.
DetailDialog	This is a popup dialog window it allow user find a student by id, and update his/her dance ticket information, such as number of early bird tickets or number of door price tickets.
MainFrame	MainFrame include menu system and dance party information.
MainPanel	Main panel show in the center of window.
Node<T>	Generic Node class for Binary Tree data structure.
RefundDialog	
Student	Student class used to store student information about the dance tickets.
Test	Test program to test add, contains, size, save, load, remove functions for BinaryTree.
TicketDialog	This dialog window is used to sell tickets to student.

Enum Class Summary

Enum Class	Description
MainFrame.TicketType	

Deployment (jar)

1. Ant build
2. target: dist
3. Generate danceticket.jar

```
<project root>
    ├── 🔨build.xml
    |
    ├── 🔥build/
    |       ├── com
    |       ├── resources
    |       └── logging.properties
    └── 🎯dist
            └── lib
                 └── danceticket.jar
```

to Run the jar file

```
cd <danceticket.jar folder>
java -jar danceticket.jar
```

❗⚡💡 Very important:

1. In order to run the jar file, put all configuration and resource files (logging.properties, dance.jpg, danceticket.properties) in classpath.
2. use InputStream to read file, instead of FileReader.

```
                String imageFile = parent.getProp().getProperty(IMAGE_FILE_PROP);
        // load image file by using InputStream
                InputStream imgIn = MainPanel.class.getClassLoader().getResourceAsStream(imageFi
                try {
                        img = ImageIO.read(imgIn);
                } catch (IOException e) {
                        e.printStackTrace();
                }


        private void loadProperties() {
                logger.info("loadProperties() ...");
                prop = new Properties();
        // load resources/danceticket.properties file using InputStream
                InputStream in = MainFrame.class.getClassLoader().getResourceAsStream(PROP_FILE)
                try {
                        // load a properties file
                        prop.load(in);
                } catch (IOException ex) {
                        logger.severe(ex.getMessage());
                }
        }
```

Run danceticket.jar on DOS command window.

```
C:\Users\12818\workspace\Rodney\java\john\dist\lib>java -jar danceticket.jar
file:/C:/Users/12818/workspace/Rodney/java/john/dist/lib/danceticket.jar!/logging.properties
```

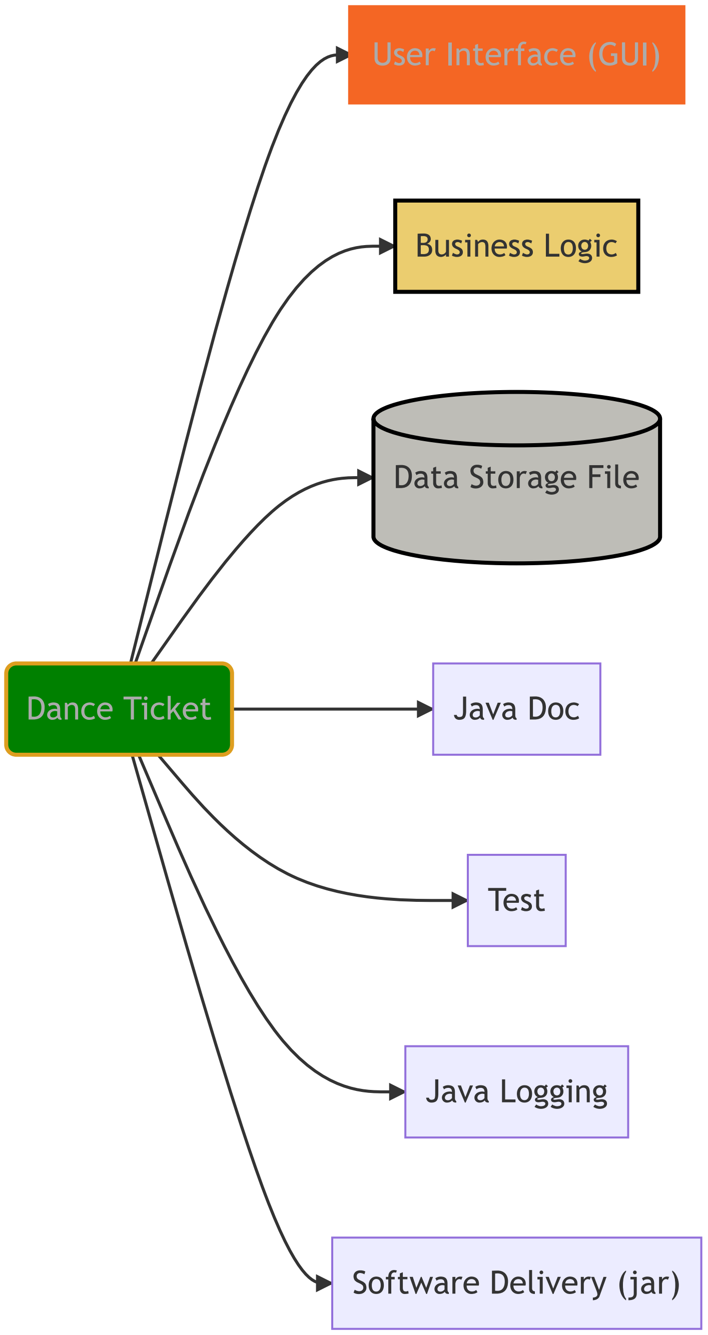
Logging

1. Logging configuration file: logging.properties
2. logging output file: /workspace/Rodney/java/mylogs%u.log

3. Create logger

```
        static {
//              System.setProperty("java.util.logging.config.file", "/Users/12818/workspace/Rodr
                String path = MainFrame.class.getClassLoader().getResource("logging.properties")
                System.setProperty("java.util.logging.config.file", path);
//              System.setProperty("java.util.logging.ConsoleHandler.level", "java.util.logging.
        }
        static Logger logger = Logger.getLogger("DANCE_TICKET");
```

Software include


User Interface (GUI)
Business Logic
Data Storage File
Dance Ticket
Java Doc
Test
Java Logging
Software Delivery (jar)