Hotel Reservation Framework

CS 321 Spring 2016

Han Tsung Liu

Last Modified: Mar 22nd, 2016

Hotel Reservation System (HRS)

- Design, Coding and Testing assignments for
  - Server side of the Hotel Reservation System
- Implement HRS as one Java program
- HRS Framework assists with your implementation
- You will be provided with two .class files:
  - Framework.class
  - Parser.class
- Your team will then implement the rest of the system based on the assignment description

Hotel Reservation System Framework

- Framework provides
  - Access to system parameters
  - Functionality for reading instructions from input files
    - Correspond to inputs from
      - Customer
      - Front-desk Clerk
      - Manager
      - Timer
  - Functionality for
    - Storing, retrieving, updating, and deleting
    - Reservation and Customer data

Overview of Framework

- 1) Take in input files with sequence of instructions (6 sub-types)
- 2) Maintain the hotel reservation system for a total of 31 days in January.
- 3) Output messages for execution per instruction (3 types)
- 4) System is going to be built with Java

High level Scope of System

Parser.class

Input File Instruction
(Handle by Parser)
1) Make Reservation
2) Check In
3) Check Out
4) Management Report
5) Next Day Signal
6) 6 PM Signal

Framework.class

Hotel Reservation System
Framework
1) Database for storing customer and reservation objects

Still Need
1) Managing different types of instructions (keep track of date, room availability, store customer and reservation information)

Output Messages

Still Need
1) Execution trace
2) Management Report
3) Error Messages

Parser.class (similar to an Iterator)

Framework Method	Description
void **init**(String *filename*)	Initializes the framework to retrieve instructions from the specified input file. If the file does not exist, or if the file cannot be opened, the framework will throw an IOException.
boolean **hasNextInstruction**()	Returns true if the file contains more instructions, otherwise returns false.
String[] **nextInstruction**()	Returns the next instruction as an array of strings. The length and content of the array depend on the instruction.

Example Usage

```
public static void main(String [] args){
    try{
        Framework.init(args[0]);
        while(Framework.hasNextInstruction()){
            String [] instructions = Framework.nextInstruction();
            executeInstruction(instructions);
            // displays all the instructions
            for(int i = 0; i < instructions.length ; i ++){
                System.out.println(instructions[i]);
            }
        }
    }
    catch(Exception IOException){
        // catches exception
    }
}
public static void executeInstruction(String [] instructions){
    // instruction type is stored in the first String
    int instructionType = Integer.parseInt(instructions[0]);
}
```

Standard Input File Format

- 1. Make a Reservation
- 2. Check in
- 3. Check out
- 4. Print Management Report
- 5. Day Change (move to next day)
- 6. 6pm alarm (automatic cancellation for non-guaranteed)

Make A Reservation "@1"

String	Content	Range or Example
0	1	1
1	Name	George Mason
2	Address	4400 University Drive, Fairfax,VA 22030
3	Check In Date	(1-31)
4	Check Out Date (1-31)	(1-31)
5	Room Type (1 or 2)	(1 or 2)
6	Number of Occupants (1-4)	(1-4)
7	Guaranteed	(0=No, 1=Yes)
8	Credit Card Info	(Visa, MasterCard...etc)
9	Credit Card Expiration Date	3/2016
10	Credit Card Number	1234 1234 1234 1234 12

Check In "@2"

Line	Content	Range/Example
0	2	2
1	Name	George Mason
2	Update Credit Card Information	MasterCard
3	Credit Card Expiration Date	3/2016
4	Credit Card Number	1234 1234 1234 1234 12

Check Out "@3"

Line	Content	Type/(Range/Example)
0	3	3
1	Name	George Mason

Print Management Report "@4"

Line	Content	Example
0	4	4

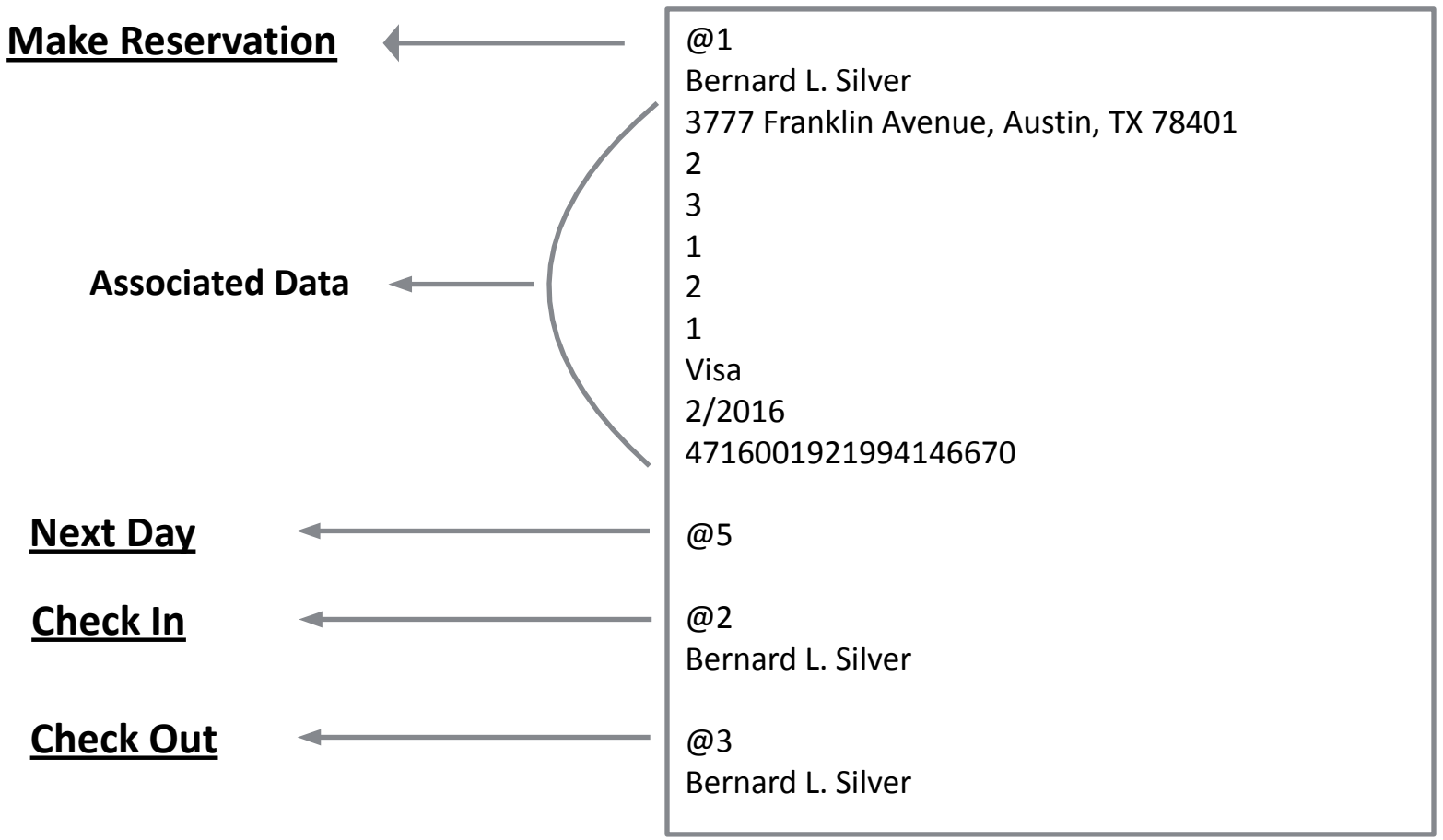
Day Change "@5"

Line	Content	Example
0	5	5

6PM Signal "@6"

Line	Content	Example
0	6	6

Example Input File



More Information

- Each set of instructions starts with “@#”
- Each line is separate by “\n”
- There can be multiple “\n” between two instruction sets.
- The system you build should validate user input if required by your professor.

Framework.class

- Manage customer and reservation data

(Customer and Reservation Objects)

- Update 2016: Data Encapsulation and Validation

----- Getting Started -----

- First step is to create a Customer and Reservation Java Class.

(Code all Instance Variables and Define “Setters” and “Getters”)

Framework Class Variables - FINAL

Parameter	Description
NUM_SINGLE_ROOMS	The number of single rooms available in the hotel
NUM_DOUBLE_ROOMS	The number of double rooms available in the hotel
NUM_DAYS	The number of days that the hotel reservation system must support
SINGLE_RATE	The nightly rate for a single room (in dollars)
DOUBLE_RATE	The nightly rate for a double room (in dollars)
STATUS_RESERVED	The status code for a reserved reservation
STATUS_CHECKED_IN	The status code for a checked in reservation
STATUS_CHECKED_OUT	The status code for a checked out reservation
STATUS_NO_SHOW	The status code for a no show reservation
STATUS_MUST_PAY	The status code for a must pay reservation

Customer Methods in Framework

Methods
int **storeCustomer**(Customer *cus*)
boolean **deleteCustomer**(int *customerID*)
Customer **getCustomerByID**(int *customerID*)
Customer **getCustomerByName**(String *name*)
boolean **modifyCustomer**(int *customerID*, Customer *cus*)

Reservation Methods in Framework

Methods
int **storeReservation**(Reservation *res*)
boolean **deleteReservation**(int *reservationID*)
Reservation **getReservationByID**(int *reservationID*)
Reservation **getReservationByCID**(int *customerID*)
boolean **modifyReservation**(int *reservationID*, Reservation *res*)

Step 2: Class Instance Variables

Reservation	Customer
reservationID - INTEGER	customerID - INTEGER
status - INTEGER (1-5)	name - STRING
startDate – INTEGER (1-31)	Address – STRING
endDate – INTEGER (1-31)	ccType – STRING
roomType – INTEGER (1-2)	ccNumber – STRNIG
numOccupants – INTEGER (1-4)	ccExpiration – STRING
guaranteed – INTEGER (0-1)	
roomNum – INTEGER (10 rooms total)	
customerID - INTEGER	

Customer: Setters and Getters

Setters	Getters
setCustomerID(int id): void	getCustomerID(): int
setName(String name): void	getName(): String
setAddress(String address): void	getAddress(): String
setCCType(String ccType): void	getCCType(): String
setCCNumber(String ccNumber): void	getCCNumber(): String
setCCExpiration(String ccExpiration): void	getCCExpiration(): String

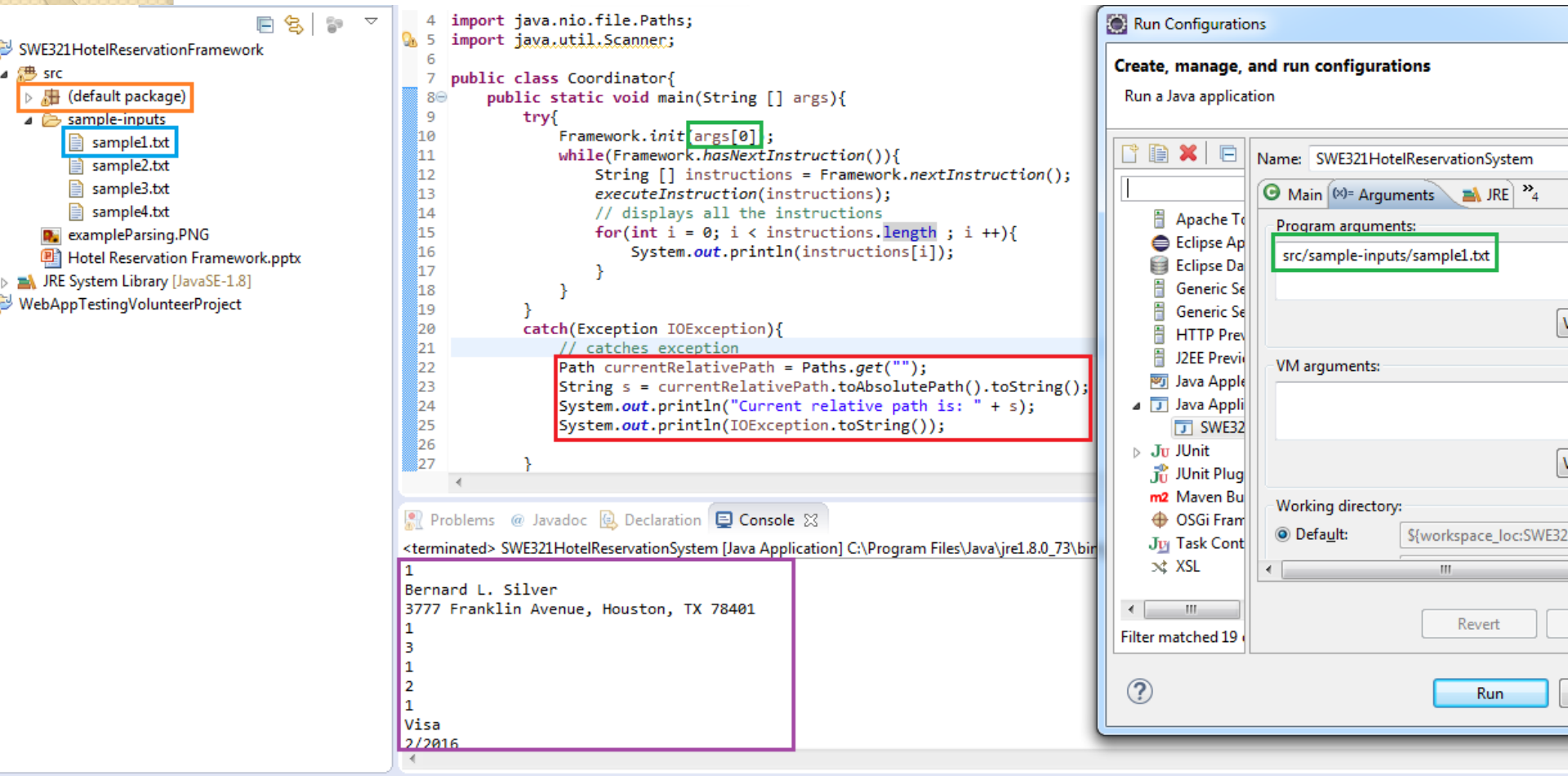
Reservation: Setters and Getters

Setters	Getters
setReservationID(int reservationID): void	getReservationID(): int
setStatus(int status): void	getStatus(): int
setStartDate(int startDate): void	getStartDate(): int
setEndDate(int endDate): void	getEndDate(): int
setRoomType(int roomType): void	getRoomType(): int
setNumOccupants(int numOccupants): void	getNumOccupants(): int
setGuaranteed(int guaranteed): void	getGuaranteed(): int
setRoomNumber(int roomNum): void	getRoomNumber(): int
setCustomerID(int customerID): void	getCustomerID(): int

System Setup Environment

- Github/Blackboard: download a copy of framework.class and parser.class
- CMD: Setup environment variables to include java sdk.
- Eclipse: create new project and copy over framework.class and parser.class

Eclipse Input File Example



General Tips

- Follow user guide instructions
- **Correct setter and getter names!**
- Keep your own list of reservationIDs for search and retrieve
- Framework "**deep copies**" all data
- Get started early and come to my office hours for any questions

(better before than after grading!)

Contact Me

- Name: Han Liu
- Email: hliu10@gmu.edu
- Subject Line: SWE321-section00X_**YourFullName**_TeamNumber
- Tuesday 1:30 – 3:30pm or by appointment

Questions?