Object Oriented Modelling

Activities

## Installing Eclipse

1. Download zip
2. Open with compressed (zipped) folders.
3. Produces a file explorer
4. Click on Extract all files change to suitable place (careful > 300MB).
5. Skip all files that require a password.
6. Open the extracted folder and double click on the eclipse executable.
7. Select a workspace (where you will hold your projects).
8. Go to the workbench (right hand icon).

## Installing the Hotel Case Study

1. Download the Hotel project and place it into the workspace you selected above.
2. In Eclipse use menu File -> New -> Java Project then create project from existing source.
3. Navigate to the hotel project you placed in the workspace.
4. Name the project hotel
5. Finish.
6. The project should appear in the file browser of Eclipse.
7. Now open up the Hotel project. In src you will find all the source code.

## Running the Hotel Project

1. Open the src then default package and select Main.java
2. Select Run menu and Run Configurations
3. Select Java Applet and then the left hand icon for creating a new launch configuration
4. The Hotel application will run and the output appears in the console.
5. Look at Main.java to see what the program is doing. You will see that it creates a booking system and then performs some tasks. After a few tasks the system prints out a snapshot of the system in XML format. This is a great way to see what is happening in a system and to debug it. Make sure you understand the XML format and how it relates to the UML class diagram.
6. From now on you can use the launch configuration you just created to run the application by double clicking on it (select Run menu first).

## Running StarUML

1. The project contains a StarUML model called Hotel.uml.
2. Using a Windows file browser, navigate to this file and double click on it.
3. If Eclipse has installed correctly you should also be able to double click on Hotel.uml in Eclipse to start StarUML.

## Activities

1. Customer 3 does not currently check-in. Modify the program in Main.java so that customer 3 checks-in, uses the spa, buys a drink, buys a meal and then checks-out. Check the XML output to ensure that this is working correctly.
2. Some hotels offer a Gym as an additional item. Modify the UML model to show how you would introduce this new feature.
3. Breakfast is a special type of meal. Modify the UML model to show how you would introduce Breakfast.
4. A Gold Status Customer is one who has stayed in any of the hotels managed by the Booking System more than 10 times. A Gold Status Customer gets 10% off their bill whenever they stay in a hotel. Modify the UML model to reflect the information necessary to manage Gold Status Customers.
5. The implementation of payment assumes that the funds are available in cash. Modify the UML model to add a banking system that manages customer accounts. The unique identifier for a customer account in the banking system is their name and address. When payment is made in the BookingSystem, the balance on the account is reduced by the appropriate amount (if available).
6. Booking is currently too simple. A customer should book a room for a given date and for a number of nights. If a customer does not check-in on the booked date then an invoice is generated (and stored in the system) and sent to their home address. Modify the UML model to implement this extension.
7. Booking does not check for an existing booking for a room on the date. What checks would need to be performed?
8. Modify the Java to introduce the Gym that you added to the model earlier. Modify Main.Java so that one of the existing customers uses the Gym. The Gym should appear in the XML output.
9. Modify the Java to introduce Breakfast that you added to the model earlier. Modify Main.java so that one of the existing customers has Breakfast.
10. Modify the Java to implement Gold Status Customers. Modify Main.java so that a customer books in and checks-out 11 times in succession and show in the XML output that the bill is reduced by 10%.
11. Modify the Java to implement payment via a banking system.
12. Modify the Java to implement the more sophisticated booking process defined in 6.