CS 350 Software Design / SE 310 Software Architecture I

Lab 2: Build a Simple Maze Game

Goal:

1. Get familiar with basic UML modeling
    a. Understand the given UML diagrams
    b. Understand Java code according to the UML diagram
2. Practice basic OO techniques using Java, including:
    a. Using classes and methods
    b. Basic I/O
    c. Java compilation and configuration

Requirements:

1. Compile the given program to show an empty maze game
2. Modify methods in the given code to add rooms in the maze
3. Compile the modified program to show a simple maze with at least two rooms, and the rooms are connected by walls or doors,
4. Modify the program again to read from input files that specify maze structures.
5. Compile the modified program to show maze game as specified in the input files

Instructions

This lab has three stages:

Stage 1: Make the program run.

1. Download the `mazeLab.zip` provided for this lab (see Bb Learn)
2. Extract the contents of `mazeLab.zip`. This will create a folder labeled `lab1`.
3. Open an Eclipse and choose `Open Projects from File System...`
4. In the dialog window click on the `Directory...` button and browse until you find the `lab1` folder. Select it, click `Open`, then `Finish`.
5. In Eclipse, right click "`JRE System Library`" and select `Build Path -> Configure Build Path`. In the new window, select "`Add External JARs`".
6. Add the `maze-ui.jar` file in the project path.
7. Attempt to run the project. If successful the message "The maze does not have any `rooms yet!`" will be displayed.

Stage 2: Build a maze with rooms

1. Fill in the `CreateMaze` function in the `SimpleMazeGame` class to create a maze with at least two rooms. **Note you have to set the current room to one of the two rooms you create.**
2. Compile and run the new program to show a maze game with rooms

Stage 3: Load a maze from a given file.

1. Two maze input files, `large.maze` and `small.maze` are included in the given `createMaze.zip`
2. Put these two files into a directory.
3. Fill in the `loadMaze` function in the `SimpleMazeGame` class to read a maze from a given file
4. Change the `main()` function so that
    a. If the input path file is given as a parameter, then a corresponding maze should be produced;
    b. If no parameter is given, then a maze should be created as you did in stage 2.
5. Compile the new program and set the runtime parameters in `Run -> Run Configurations -> Java Application -> Arguments` tab.
6. Run the new program.

After Stage 3, pack the final program into a **`zip file`** and submit it through Blackboard.

Late Policy

- Assignments submitted 1 hour to 1 week late will receive a 15% penalty.
- Assignments submitted 1 to 2 weeks late will receive an additional 10% penalty.
- Assignments submitted more than 2 weeks late will be subject to an additional 5% penalty for each week.

Appendix: Basic Maze UML Class Diagram