CS4233-D16 Programming Assignment 1

Gary Pollice

CS4233-D16 Programming Assignment 1

Overview

In this assignment, you will complete the Tollbooth tollgate system that we began developing in the lectures. I have made some changes to make your solution follow some standards to help make the testing easy. The UML diagram in this document was created by reverse engineering the code. It shows the main classes that you will work with.

Starting code

You are given an Eclipse project to start with. You need to import this project from the zipped archive into your Eclipse. When you are done, you will export it and submit it to the course pages in myWPI. The project name is **Assignment1D16**. You will change the name of the assignment to **Assignment1D16-XXX**, where you will replace "XXX" with your initials.

Notice that one of the tests provided fails. This is because you need to create an implementation of the SimpleLogger interface and use it to perform the logging. You will also need to create additional test gate controllers or modify the TestGateController that is provided to "program" it to fail in certain ways.

Important: Pay careful attention to the text of the log messages and the messages in the exceptions where they are specified in the *Confirmation* sections of the user stories below. Our test cases, and yours as well, should test to make sure that the text of the messages is correct.

Your job

You will implement and test the code for the four user stories in the *User Stories* section of this document.

You are expected to have readable, understandable code. Use meaningful names. Have correct javadoc comments on all public methods (putting these on all methods is a good habit to get into). The code should be well-formatted and consistently formatted.

You are expected to write your code and test it thoroughly. Testing it thoroughly means that you should write good unit tests to test all aspects of your code. Your code should be well-written. We will run the CodePro audit rules against your code in the **src** folder and points will be taken off for any red or yellow violation (we do not worry about blue violations). The red are high and the yellow are medium severity.

You should also achieve 90% code coverage of all classes (not interfaces) in the **src** folder with the exception of the **TollboothException** class. We will use EclEmma to get the coverage statistics.

Grading

Your submission will be graded according to the following rubric:

Category	Max points	Process
Correctness	30	30 – (# of my tests your code fails) *This can be a negative number.*
Readability/understandability	5	You will receive 0, 2, 4, or 5 points based upon the grader's evaluation of how readable and understandable your code is. This includes good, consistent formatting.
Code audit	10	10 – (2 * #red violations + #yellow violations) *This can be a negative number*
Coverage	10	10 pts. if you have 90% or greater coverage when running your tests. If your coverage is less than 90%, -1 pt. for every % less than 90%. *This can be a negative number.*

CS4233-D16 Programming Assignment 1

Submitting your work

When you have tested your code and are ready to deliver it, you need to export your project to a zipped archive file. This is easy to do from Eclipse. If you don't know how to import or export projects into Eclipse, ask one of the staff members or use the discussion forum on myWPI.

You will submit the project into the assignment entry on my WPI.

Important: NO LATE SUBMISSIONS WILL BE ACCEPTED. Don't wait until the last minute to do this assignment. It may seem trivial but will probably take at least twice as long as you think to get it right.

General User Stories

Open the Gate.

Name	Value
Name	Open the Gate.
Description	The gate receives an "open" request. The gate opens. If there is a malfunction, a log message is produced and N retries are attempted. If the malfunction still exists the gate is put in a "will not respond" state.
Status	Pending
Conversation	Conversation
ID	US001

Conversation

■ Open the gate.
■ If there is a malfunction:
• *Log the malfunction*
• *Retry N (to be specified) times*
• *If the malfunction still exists, set the gate to refuse any open/close requests.*
■ If the gate is already in the state to refuse requests, simply log the request and do nothing.
■ Throw a TollboothException after doing the proper logging if the gate cannot be opened (i.e., it ends in the will not respond state).

Confirmation

	Gate is closed. Receive the "open" message. No malfunction occurs. The gate is in the open state. The number of opens increases by 1.
	Gate is open. Receive the "open" message. No change in state or statistics occurs.
	Gate is closed. Receive the "open" message. A hardware malfunction occurs (exception thrown). A log message is produced. Attempt to open again and no malfunction. A log message is produced and the gate is in the open state. The first log message says "open: malfunction" and includes the exception thrown by the system. The second log message says "open: successful". The number of opens increases by 1.
	Gate is closed. Receive the "open" message. A hardware malfunction occurs (exception

	thrown). A log message is produced. This log message says "open: malfunction" and includes the exception thrown by the system. After N=2 additional attempts (three attempts in total), the malfunction still occurs. Two more malfunction log messages are produced, identical to the first one. A final log message is produced and the gate is put in the "will not respond" state. The final log message says "open: unrecoverable malfunction" and includes the last exception thrown by the system. If any of the retries are successful, then a log message saying "open: successful" is produced and the gate is in the open state. The number of opens is increased by 1 as long as the gate is finally opened, regardless of the number of attempts. If the gate is not opened then the statistics are not changed. If there is an unrecoverable malfunction and the gate is in the "will not respond state", then throw a TollboothGate exception with the message "open: unrecoverable malfunction" as the message and the last exception thrown by the gate controller as the cause.
	Gate is in the "will not respond" state and receives the "open" message. A log message is produced. No change made to the gate state. The log message says "open: will not respond" and no change is made to the statistics.

Close the Gate.

Name	Value
Name	Close the Gate.
Description	The gate receives a "close" request. The gate closes. If there is a malfunction, a log message is produced and N retries are attempted. If the malfunction still exists the gate is put in a "will not respond" state.
Status	Pending
Conversation	Conversation
ID	US002

Conversation

■ Close the gate
■ If there is a malfunction:
• *Log the malfunction*
• *Retry N (to be specified) times*
• *If the malfunction still exists set the gate to refuse any open/close requests.*
■ If the gate is already in the stat to refuse requests, simply log the request and do nothing.
■ Throw a TollboothException after doing the proper logging if the gate cannot be opened (i.e., it ends in the will not respond state).

Confirmation

	Gate is open. Receive the "close" message. No malfunction occurs. The gate is in the closed state. The number of opens increases by 1.
	Gate is closed. Receive the "close" message. No change in state or statistics occurs.

	Gate is open. Receive the "close" message. A hardware malfunction occurs (exception thrown). A log message is produced. Attempt to close again and no malfunction. A log message is produced and the gate is in the closed state. The first log message says "close: malfunction" and includes the exception thrown by the system. The second log message says "close: successful". The number of opens increases by 1.
	Gate is open. Receive the "close" message. A hardware malfunction occurs (exception thrown). A log message is produced. This log message says "close: malfunction" and includes the exception thrown by the system. After N=2 additional attempts (three attempts in total), the malfunction still occurs. Up to two more malfunction log messages are produced, identical to the first one. A final log message is produced and the gate is put in the "will not respond" state. The final log message says "close: unrecoverable malfunction" and includes the last exception thrown by the system. If any of the retries are successful, then a log message saying "close: successful" is produced and the gate is in the closed state. The number of closes is increased by 1 as long as the gate is finally closed, regardless of the number of attempts. If the gate is not closed then the statistics are not changed. If there is an unrecoverable malfunction and the gate is in the "will not respond state", then throw a TollboothGate exception with the message "close: unrecoverable malfunction" as the message and the last exception thrown by the gate controller as the cause.
	Gate is in the "will not respond state" and receives the "close" message. A log message is produced. No change is made to the gate state. The log message says "close: will not respond" and no change is made to the statistics.

Get the Gate Statistics

Name	Value
Name	Get the Gate Statistics
Description	Provide the number of times the gate has been opened and closed since it came online.
Status	Pending
Conversation	Conversation
ID	US004

Conversation

■ Simply a record of the number of times the gate was opened successfully and closed successfully since the gate came online.
■ Confirmation is included in the Open and Close user stories.

Reset the toll gate.

Name	Value
Name	Reset the toll gate.
Description	Set the state of the gate to be the same as when the gate object was created, with the exception of the statistics, which will not

	change.
Status	Pending
Conversation	Conversation
ID	US006

Conversation

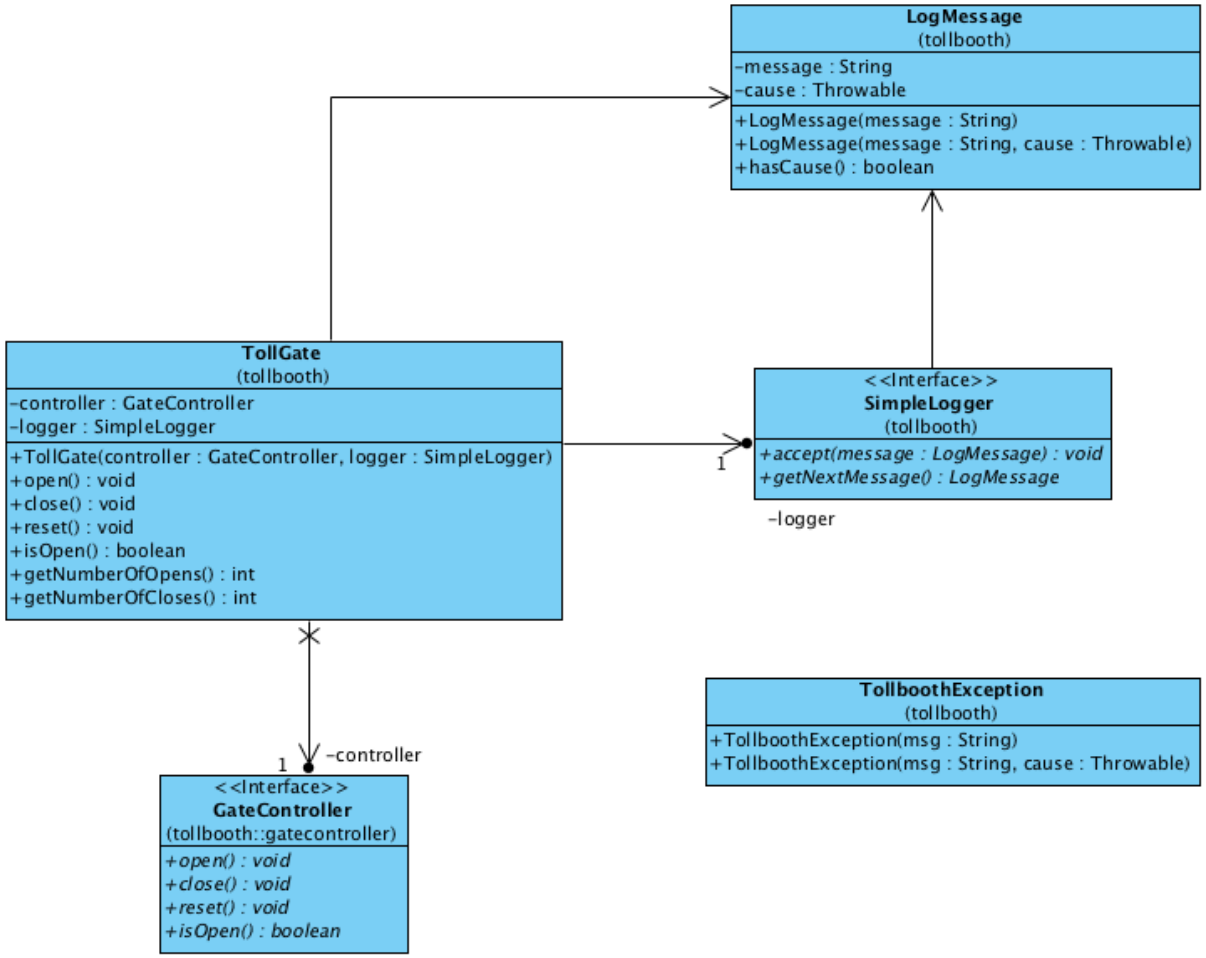
■ Gate is placed in the closed state.
■ Statistics are not changed.

Confirmation

	Gate is in the open state and receives the "reset" message. The gate needs to be put in the closed state. Do exactly what is done for the Close Gate user story (US002) when the gate is in the open state. As long as the gate is closed, an additional log message (It may be the only one) saying "reset: successful" is produced. No change is made to the statistics.
	Gate is in the closed state and receives the "reset" message. Gate is in the closed state. The statistics (# of opens and closes) are the same after the operation as before the operation. (i.e., nothing changes). A log message saying "reset: successful" is produced.
	Gate is in the will not respond state and receives the "reset" message. Do exactly what is done for the Close Gate user story (US002) when the gate is in the open state, changing “close” to “reset” in the messages. As long as the gate is closed, an additional log message saying "reset: successful" is produced. No change is made to the statistics.
	Gate is in the open or closed state and receives the "reset" message. The gate needs to be put in the closed state. Do exactly what is done for the Close Gate user story (US002) when the gate is in the open or closed state respectively, changing “close” to “reset” in the messages. As long as the gate is closed, an additional log message saying "reset: successful" is produced. No change is made to the statistics.

Class Diagram

Assignment1-D16



Summary

Name	Description
LogMessage	The LogMessage is a simple data structure that contains information about any system malfunction. @version Feb 18, 2016
TollGate	The TollGate contains everything about a tollgate in a tollbooth. @version Feb 3, 2016
SimpleLogger	The TollboothLogger implements a FIFO queue that only allows a message to be added and removed from the queue. The implementation of the logger may vary depending upon the system requirements. @version Feb 18, 2016

TollboothException	Description @version Feb 3, 2016
GateController	This interface defines the behavior expected of any tollgate controller. @version Feb 3, 2016