CMPE 275 Section 2, Fall 2016

Lab 1 - Aspect Oriented Programming

*Last updated: 10/07/2016*

In this lab, you implement the retry and stats concerns to a tweeting service through Aspect Oriented Programming (AOP). ~~For ease of submission and grading, we wrap the two concerns into a single aspect,~~ *~~RetryAndDoStats.java~~*~~.~~ Please note this is an individual assignment.

The tweet service is defined as follows:

*package edu.sjsu.cmpe275.aop;*

*import java.io.IOException;*

*public interface TweetService {*

*/\*\**

*\* @throws IllegalArgumentException if the message is more than 140 characters as measured by string length.*

*\* @throws IOException if there is a network failure*

*\*/*

*void tweet(String user, String message) throws IllegalArgumentException, IOException;*

*/\*\**

*\* @throws IOException if there is a network failure*

*\*/*

*void follow(String follower, String followee) throws IOException;*

*}*

Since network failure happens relatively frequently, you are asked to add the feature to automatically retry for up to two times for a network failure (indicated by an IOException). (Please note the two retries are in addition to the original failed invocation.) You are also asked to implement the following TweetStats service:

*package edu.sjsu.cmpe275.aop;*

*public interface TweetStats {*

*/\*\**

*\* reset all the three measurements.*

*\*/*

*void resetStats();*

*/\*\**

*\* @return the length of longest message successfully tweeted since the beginning or last reset. If no messages were successfully tweeted, return 0.*

*\*/*

*int getLengthOfLongestTweet();*

*/\*\**

*\* @return the user who has attempted to follow the biggest number of different users since*

*\* the beginning or last reset. If there is a tie, return the 1st of such users based on*

*\* alphabetical order. Even if the follow action did not succeed, it still counts toward the stats.*

*\* If no users attempted to follow anybody, return null.*

*\*/*

*String getMostActiveFollower();*

*/\*\**

*\* The most productive user is determined by the total length of all the messages successfully tweeted since the beginning*

*\* or last reset. If there is a tie, return the 1st of such users based on alphabetical order. If no users successfully tweeted, return null.*

*\* @return the most productive user.*

*\*/*

*String getMostProductiveUser();*

*}*

*TweetStatsImpl.java*

*package edu.sjsu.cmpe275.aop;*

*public class TweetStatsImpl implements TweetStats {*

*//...*

*}*

Your implementation of the two concerns need to be done in the two files: *RetryAspect.java and StatsAspect.java.* For example, *RetryAspect.java should look like the following:*

*package edu.sjsu.cmpe275.aop.aspect;*

*import org.aspectj.lang.annotation.Aspect; // if needed*

*import org.aspectj.lang.annotation.Before; // ifneeded*

*@Aspect*

*public class RetryAspect {*

*...*

*}*

You do not need to worry about multi-threading; i.e., you can assume invocations on the tweet service and stats service will come from only one thread.

For your testing purpose, you need to provide your own implementation of *TweetServiceImpl.java,* and simulate failures, but you do not need to submit this file, as the TA will use his own implementation(s) for grading purpose.

## Project Setup

You can refer to the tutorial [here](http://www.journaldev.com/2583/spring-aop-example-tutorial-aspect-advice-pointcut-joinpoint-annotations) on how to start the lab. A sample project with build file with dependencies, application context, and Java files is [here (version 1-1, last updated on 11:38 pm 10/8/2016)](https://drive.google.com/a/sjsu.edu/file/d/0B-l8bNupzZKhQXlnbFlOU0tpWFU/view?usp=sharing) for your reference and testing.

## Example Stats

The following examples are assuming stats are reset() before running every single example. Additional test cases will be used for grading.

1. Tweet message as tweet(“foo”,”barbar”). Then getLengthOfLongestTweet() returns 6.
2. Alice follows Bob, Bob follows Charlie (but fails to do so), and Bob follows Alice. getMostActiveFollower() returns “Bob”.
3. Successfully tweet a message ("Alice","[any message <= 140 chars]"), then getMostProductiveUser() returns “Alice”.

## Submission

Please submit through Canvas, only the **three** java files, *RetryAspect.java, StatsAspect.java, and TweetStatsImpl.java.* The code you submit must compile with the given project setup (to be provided). Your three java files CANNOT include any additional classes or packages, except those under java.util or those already provided in the given build dependencies. If your code does not compile with the TA’s code because of extra inclusion or dependency, you automatically lose most of your correctness points.

## Due date

Pleaser refer to Canvas.

## Grading:

This lab has a total point of 6, with 5 points for the correctness of your implementation of the retry and stats concerns. Code structure and Java documentation are worth 1 point.