```
/*************************************************************************\
 * TwitterClient.java                                                    *
 * @author Hawk Weisman                                                  *
 * PLEDGE:                                                               *
 *                                                                       *
 * A basic command-line Twitter client.                                  *
 * Contains code modified from Professor Gregory Kapfhammer's.           *
 *                                                                       *
 *  +-----------------------------------------------------+              *
 *  | Command-line arguments:                             |              *
 *  |                                                     |              *
 *  | timeline: prints the twitter home timeline          |              *
 *  | mentions: prints your mentions timeline             |              *
 *  | tweet: prompts the user for a tweet, and posts it   |              *
 *  | tweet-txt: posts all tweets contained in tweets.txt |              *
 *  | random: posts a randomly generated tweet            |              *
 *  | randomTest: debugs the randomizer                   |              *
 *  +-----------------------------------------------------+              *
\*************************************************************************/

import java.util.Random;
import java.util.Scanner;
import java.util.ArrayList;
import java.io.File;
import java.io.FileNotFoundException;
import java.util.List;
import twitter4j.*;

public class TwitterClient {

    /**
     * randomTest
     * @author Hawk Weisman
     */
    public static void randomTest () {

        Tweet randomTweet = new Tweet();
        Random random = new Random();
        String randomTweetText = new String();

        int firstPart, secondPart, thirdPart;

        // assign each tweetPart to a random integer between 1 and 4
        firstPart = random.nextInt(4)+1;
        secondPart = random.nextInt(4)+1;
        thirdPart = random.nextInt(4)+1;

        System.out.println (" + Debugging the randomizer." + "\n" +
                            " + firstPart = " + firstPart + "\n" +
                            " + secondPart = " + secondPart + "\n" +
                            " + thirdPart = " + thirdPart + "\n");
    }

    /**
     * randomTweet
     * @author Hawk Weisman
```

```
     * @return a randomly-generated tweet.
     */
    public static Tweet randomTweet () {

        Tweet randomTweet = new Tweet();
        Random random = new Random();
        String randomTweetText = new String();

        int firstPart, secondPart, thirdPart;

        // assign each tweetPart to a random integer between 1 and 4
        firstPart = random.nextInt(4)+1;
        secondPart = random.nextInt(4)+1;
        thirdPart = random.nextInt(4)+1;

        switch (firstPart) {
            case 1:
                randomTweetText = "This tweet was ";
                break;
            case 2:
                randomTweetText = "The following message was ";
                break;
            case 3:
                randomTweetText = "You are reading a tweet that was ";
                break;
            case 4:
                randomTweetText = "This message was ";
                break;
            default:
                break;
        }

        switch (secondPart) {
            case 1:
                randomTweetText = randomTweetText + "randomly generated by ";
                break;
            case 2:
                randomTweetText = randomTweetText + "created randomly using ";
                break;
            case 3:
                randomTweetText = randomTweetText + "made randomly thanks to ";
                break;
            case 4:
                randomTweetText = randomTweetText + "pulled from the ether by
                    means of ";
                break;
            default:
                break;
        }

        switch (thirdPart) {
            case 1:
                randomTweetText = randomTweetText + "a highly advanced computer
                    program.";
                break;
            case 2:
```

```
                randomTweetText = randomTweetText + "an in-house software tool.";
                break;
            case 3:
                randomTweetText = randomTweetText + "cutting-edge algorithmic
                    techniques.";
                break;
            case 4:
                randomTweetText = randomTweetText + "black magic.";
                break;
            default:
                break;
        }

        randomTweet.setMessage(randomTweetText);
        return randomTweet;
    }

    /**
     * parseLang
     * @author Hawk Weisman
     * @param lang A lang string from Twitter4J's getLang() method
     * @return A person-readable language
     */
    public static String parseLang (String lang) {

        switch (lang) {
            case "en":
                return "English";

            case "es":
                return "Spanish";

            case "it":
                return "Italian";

            case "pt":
                return "Portugese";

            case "tr":
                return "Turkish";

            case "fr":
                return "French";

            case "ko":
                return "Korean";

            case "jp":
                return "Japanese";

            case "ru":
                return "Russian";

            case "de":
                return "German";
```

```
        default:
            return "null";
    }
}

/**
 * main
 * @author Hawk Weisman
 */
public static void main (String[] argv) {

    ArrayList<Tweet> validTweets = new ArrayList<Tweet>(0);     // stores all
        valid tweets
    ArrayList<String> invalidTweets = new ArrayList<String>(0);  // stores
        all invalid tweets

    File tweets;    // represents the tweets.txt file to be read from
    Scanner scan;   // scanner to read tweets from tweets.txt

    Twitter twitter;                // Twitter object from twitter4j to be
        initialized later
    User user;                      // User object from twitter4j to be
        initialized later
    List<Status> timeline;          // A List of already posted statuses
        pulled from twitter
    List<Status> mentions;          // A List of all the statuses mentioning
        the user

    final int MAX_LENGTH = 140;     // maximum length for a valid tweet
    final int MIN_LENGTH = 1;       // minimum length for a valid tweet

    String currentLine = new String();  // holds the string being sorted

    System.out.print("\n");

    // if there are no command-line arguments, print a help file detailing
        accepted CLIs
    if (argv.length == 0) {
       System.out.println ("TwitterClient: Please enter a command-line
           argument. \n"
                           + "Accepted command-line arguments: \n"
                           + " + timeline: prints your twitter home timeline
                               \n"
                           + " + mentions: prints your mentions timeline \n"
                           + " + @<username>: prints information on user
                               <username>, followed by their latest tweet \n"
                           + " + tweet: prompts the user for a tweet, and
                               posts it \n"
                           + " + tweet-txt: posts all tweets contained in
                               Tweet.txt \n"
                           + " + random: posts a randomly generated tweet
                               (warning: probably stupid)");
    }

    // if there are one or more command-line arguments, decide what to do
        next
```

```
        else if (argv.length > 0) {
            // loop through argv
            for (String arg : argv) {

                // if one of the entered argument is a user...
                if (arg.substring(0,1).equals("@")) {
                    try {

                        // gets Twitter instance with default credentials
                        twitter = new TwitterFactory().getInstance();
                        user = twitter.verifyCredentials();

                        // grab the user to be viewed
                        if (twitter.searchUsers(arg,1).size() > 0) {
                            // make a User object to store the user
                            User viewUser = twitter.searchUsers(arg,1).get(0);
                            // tale their last status and put it in a Status
                                object
                            Status theirStatus = viewUser.getStatus();

                            // print the user's information
                            System.out.println ("User " + arg + " has " +
                                viewUser.getFollowersCount() + " followers, "
                                                + viewUser.getFriendsCount() + "
                                                    friends, and has posted " +
                                                    viewUser.getStatusesCount() +
                                                    " statuses.");
                            System.out.println ("They live in " + viewUser.
                                getLocation() + " and speak " + parseLang
                                (viewUser.getLang()) + ".");
                            System.out.println ("Description:" + "\n" + viewUser.
                                getDescription());
                            System.out.println ("Last status:" + "\n" +
                                theirStatus.getText());

                        } else {
                            System.out.println ("No user named " + arg + " could
                                be found.");
                        }

                        System.out.print("\n");

                    }

                    // catch any thrown TwitterExceptions
                    catch (TwitterException te) {
                        te.printStackTrace();
                        System.out.println("Failed to get user " + arg +": " + te
                            .getMessage());
                    }

                } else {
                    // decide what to do based on the current arg
                    switch (arg) {

                        case "timeline":
```

```
                // print the home timeline to the console
                try {

                    // gets Twitter instance with default credentials
                    twitter = new TwitterFactory().getInstance();
                    user = twitter.verifyCredentials();

                    // grab statuses from twitter and put them in
                        timeline
                    timeline = twitter.getHomeTimeline();

                    // print all the statuses in timeline
                    System.out.println("Showing @" + user.
                        getScreenName() + "'s home timeline.");
                    for (Status status : timeline) {
                        System.out.println("@" + status.getUser().
                            getScreenName() + " - " + status.getText
                            ());
                    }

                    // if there are no statuses in the timeline, say
                        that.
                    if (timeline.size() == 0)
                        System.out.println ("You have no statuses in
                            your timeline.");

                    System.out.print("\n");
                }

                // catch TwitterExceptions thrown while connecting to
                    Twitter
                catch (TwitterException te) {
                    te.printStackTrace();
                    System.out.println("Failed to get timeline: " +
                        te.getMessage());
                }
            break;

        case "mentions":
            // print the home timeline to the console
            try {

                // gets Twitter instance with default credentials
                twitter = new TwitterFactory().getInstance();
                user = twitter.verifyCredentials();

                // grab statuses from the mentions timeline and put
                    them in mentions
                mentions = twitter.getMentionsTimeline();

                // print all the statuses in timeline
                System.out.println("Showing @" + user.getScreenName()
                    + "'s mentions timeline.");
                for (Status status : mentions) {
                    System.out.println("@" + status.getUser().
                        getScreenName() + " - " + status.getText());
```

```
            }

            // if there are no mentions, say that.
            if (mentions.size() == 0)
                System.out.println ("You have no statuses in your
                    mentions timeline.");

            System.out.print("\n");
        }

        // catch TwitterExceptions thrown while connecting to
            Twitter
        catch (TwitterException te) {
            te.printStackTrace();
            System.out.println("Failed to get timeline: " + te.
                getMessage());
        }
        break;

        case "tweet":
            // set up the scanner to scan from the command line
            scan = new Scanner(System.in);

            // prompt the user for a tweet
            System.out.println ("Please enter a tweet and press
                return.");
            currentLine = scan.nextLine();
            System.out.print("\n");

            // if the tweet is valid, put it in the validTweets
                ArrayList
            if (Tweet.isValidMessage(currentLine) == true){
                Tweet tweet = new Tweet();
                tweet.setMessage(currentLine);
                validTweets.add(tweet);

                // otherwise, put it in invalidTweets
            } else {
                invalidTweets.add(currentLine);
            }
        break;

        case "tweets-txt":
            try {
                // set up the scanner to read from tweets.txt
                tweets = new File ("tweets.txt");
                scan = new Scanner(tweets);

                // loop through all the lines in tweets.txt and
                    sort them into ArrayLists.
                while (scan.hasNextLine()) {
                    currentLine = scan.nextLine();

                    // if the tweet is valid, put it in the
                        validTweets ArrayList
                    if (Tweet.isValidMessage(currentLine) == true
```

```
                                    ){
                                    Tweet tweet = new Tweet();
                                    tweet.setMessage(currentLine);
                                    validTweets.add(tweet);

                                // otherwise, put it in invalidTweets
                                } else {
                                    invalidTweets.add(currentLine);
                                }
                            }
                        }

                        // catch any exceptions thrown while reading in
                            Tweets.txt
                        catch (FileNotFoundException e){
                            e.printStackTrace(System.err);
                        }
                    break;

                case "random":
                        // add a random tweet to the random tweets ArrayList.
                        validTweets.add(randomTweet());
                    break;

                case "randomTest":
                    randomTest();
                    break;

                default:    // the default case should never trigger.
                    break;
                }
            }

        // print out the valid tweets
        if (validTweets.size() > 0) {
            System.out.println("Valid tweets:");
            for (Tweet tweet : validTweets)
                System.out.println(tweet.toString());
            System.out.print("\n");
        }

        // print out the invalid tweets
        if (invalidTweets.size() > 0) {
            System.out.println("Invalid tweets:");
            for (String invalidTweet : invalidTweets)
                System.out.println(invalidTweet);
            System.out.print("\n");
        }


        // try to post all valid tweets to the timeline
        if (validTweets.size() > 0) {
            try {
                // connect to Twitter
                twitter = new TwitterFactory().getInstance();
                user = twitter.verifyCredentials();
```

```
                timeline = twitter.getHomeTimeline();

                // loop through validTweets and post each tweet to Twitter
                for (Tweet tweet : validTweets)
                    twitter.updateStatus(tweet.getMessage());
                }

            // catch any TwitterExceptions thrown while posting valid tweets
            catch (TwitterException te) {
                te.printStackTrace();
                System.out.println("Failed to post tweet " + te.getMessage());
            }
        }
            }
        }
    }
}
```