The Effects of Perceived Health Events on the Language of Twitter Users

Ruth Chen
Senior Project
Spring 2016

The Effects of Perceived Health Events on the Language of Twitter Users

Ruth Chen – Senior Project

The purpose of this study is to research the effects of perceived health events on the language of Twitter users. Data collection will occur on Twitter, using health-related keywords to query for tweets and analyzing users' timelines before and after a health event as perceived on Twitter.

The following documentation describes the program used for data collection.

This program is written by Ruth Chen (ruthchen@wustl.edu), a psychology and computer science double major in the Washington University Class of 2016, under the advising of Ron Cytron (cytron@wustl.edu), the Associate Department Chair & Professor of Computer Science at Washington University.

This program is written in Java and builds on Twitter4J (http://twitter4j.org/en/index.html) by Yusuke Yamamoto (yusuke@mac.com).

This program is written for the Personality and Measurement Development Lab in the Psychological & Brain Sciences Department at Washington University. This project was overseen by Sara Weston (weston.sara@gmail.com) and Joshua Jackson (j.jackson@wustl.edu).

Table of Contents

Getting Started

To run this program, you will need to install Java 8 and Eclipse. Watch Ron Cytron's video tutorial: http://www.cs.wustl.edu/~jp/cse131/Cal/Videos/Instructions/installeclipse.mp4 or continue with the following steps.

Installing Java 8

1. Go on the Oracle Downloads page: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
2. In the first table on the page (titled "Java SE Development Kit 8u91"), find your operating system. Then, select "Accept License Agreement" at the top of the table.

Java SE Development Kit 8u91

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

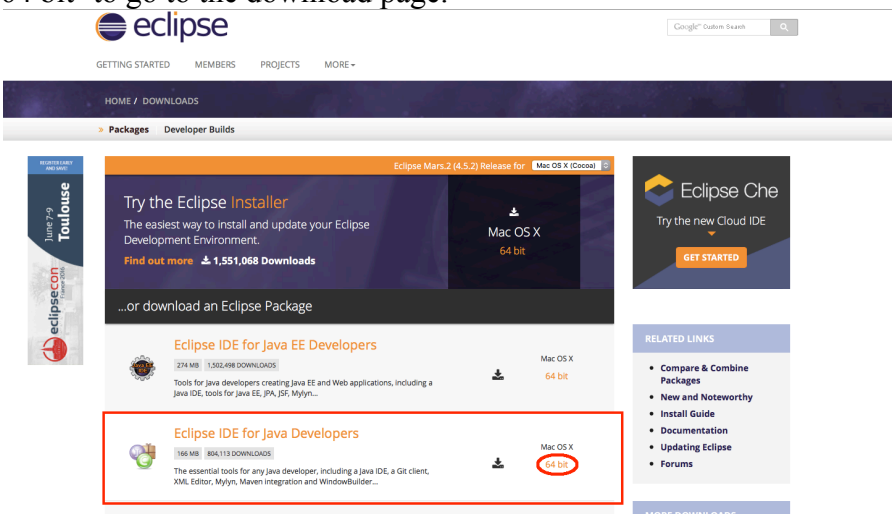
○ Accept License Agreement ◉ Decline License Agreement

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.72 MB	jdk-8u91-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.69 MB	jdk-8u91-linux-arm64-vfp-hflt.tar.gz
Linux x86	154.74 MB	jdk-8u91-linux-i586.rpm
Linux x86	174.92 MB	jdk-8u91-linux-i586.tar.gz
Linux x64	152.74 MB	jdk-8u91-linux-x64.rpm
Linux x64	172.97 MB	jdk-8u91-linux-x64.tar.gz
Mac OS X	227.29 MB	jdk-8u91-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.59 MB	jdk-8u91-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.95 MB	jdk-8u91-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.29 MB	jdk-8u91-solaris-x64.tar.Z
Solaris x64	96.78 MB	jdk-8u91-solaris-x64.tar.gz
Windows x86	182.11 MB	jdk-8u91-windows-i586.exe
Windows x64	187.41 MB	jdk-8u91-windows-x64.exe

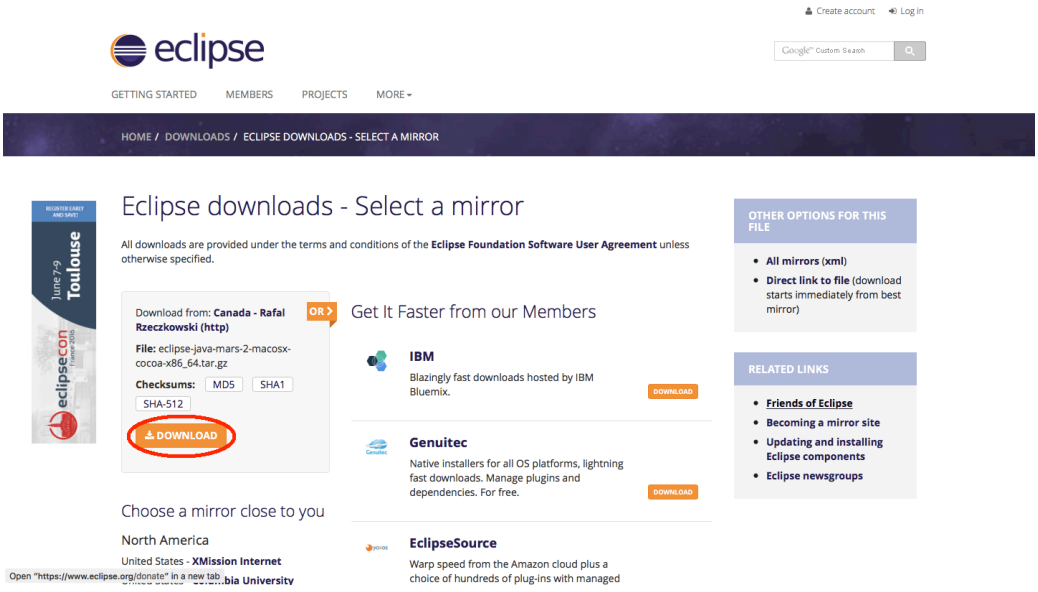
3. Click on the download link for your operating system. Follow the instructions in the program to finish installation.

Installing Eclipse

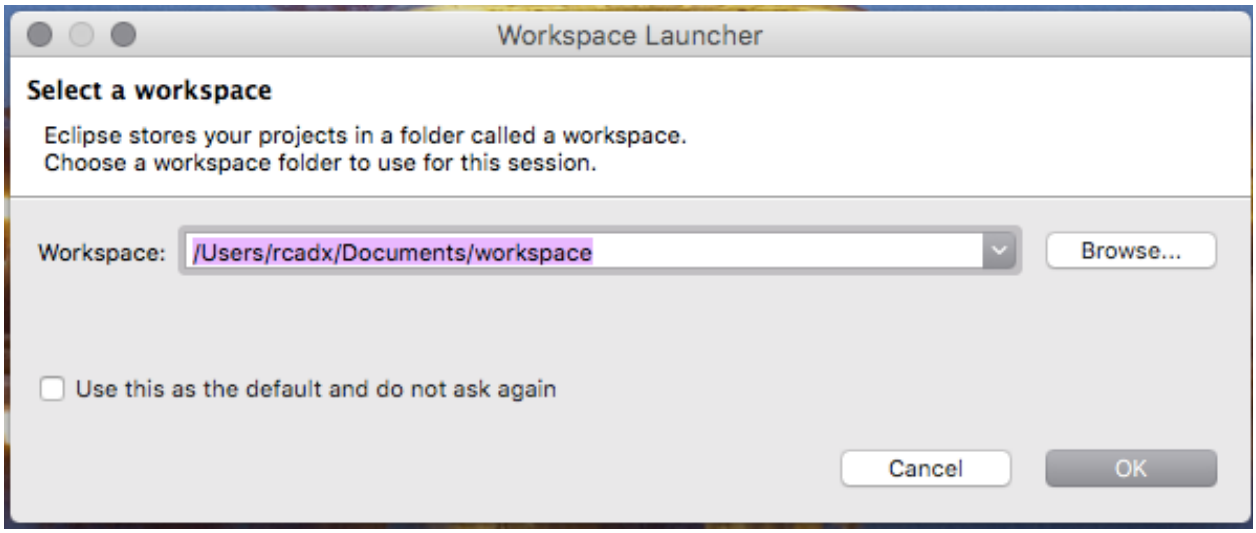
1. Go to the Eclipse Downloads page: https://eclipse.org/downloads/
2. Download the Eclipse IDE for Java Developers. It should be the 2nd option on the page. Click on '64 bit' to go to the download page.



3. Click on the ‘Download’ button on the left side of the page.



4. Open the downloaded file, then follow the instructions in the program to finish installation.
5. Once you’ve installed Eclipse, double click the Eclipse icon to run it. A popup box will appear asking you to select a workspace. The textbox by ‘Workspace’ should already be filled in. Click the ‘OK’ button.

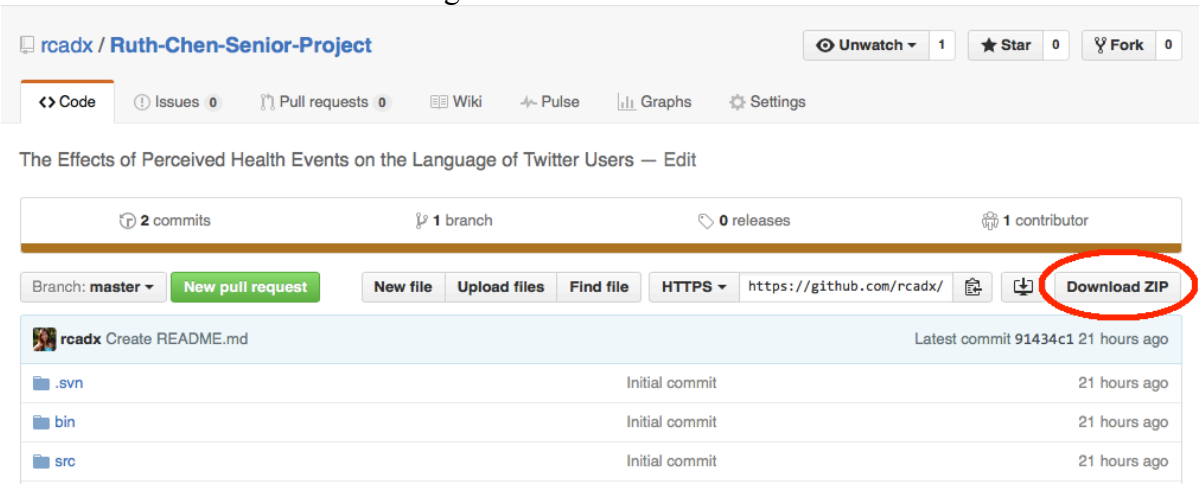


6. Exit the ‘Welcome’ tab in Eclipse to finish!

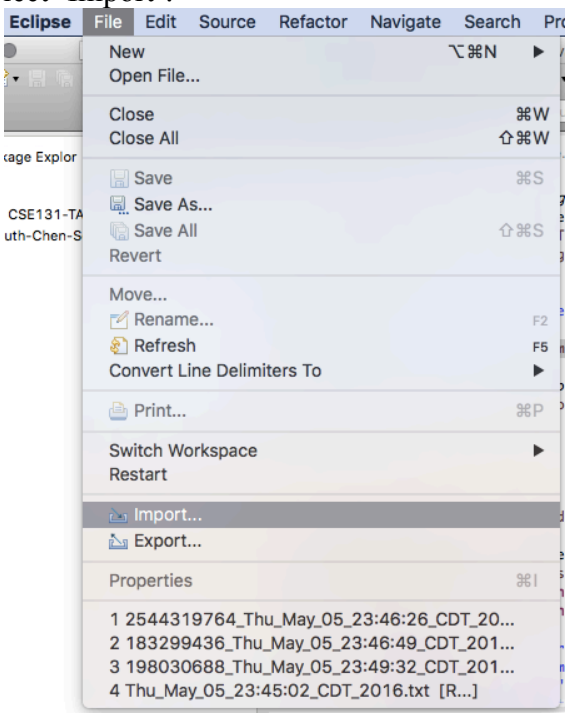
Setting up the Code

Now that you have Java 8 and Eclipse installed, the next step is to copy the program code into Eclipse.

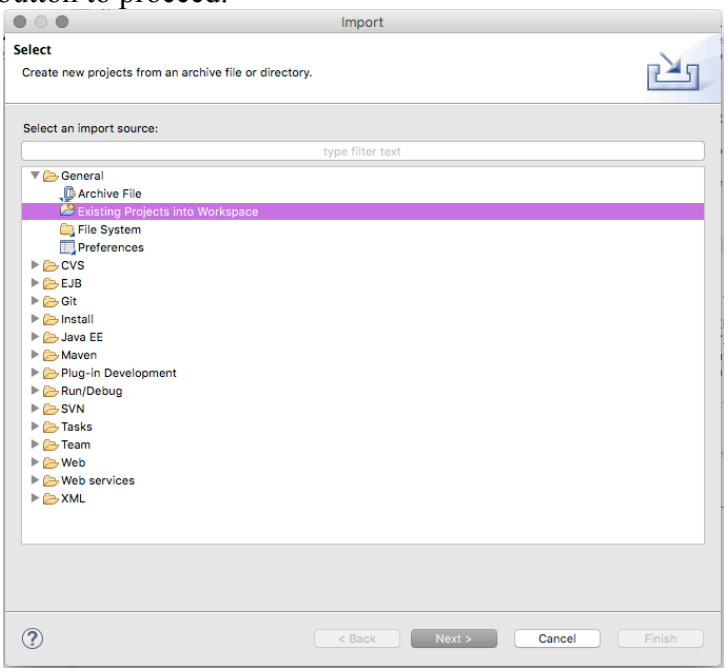
1. Go on Github and find the repository Ruth-Chen-Senior-Project by user rcadx: https://github.com/rcadx/Ruth-Chen-Senior-Project
2. Click on "Download ZIP" in the right corner.



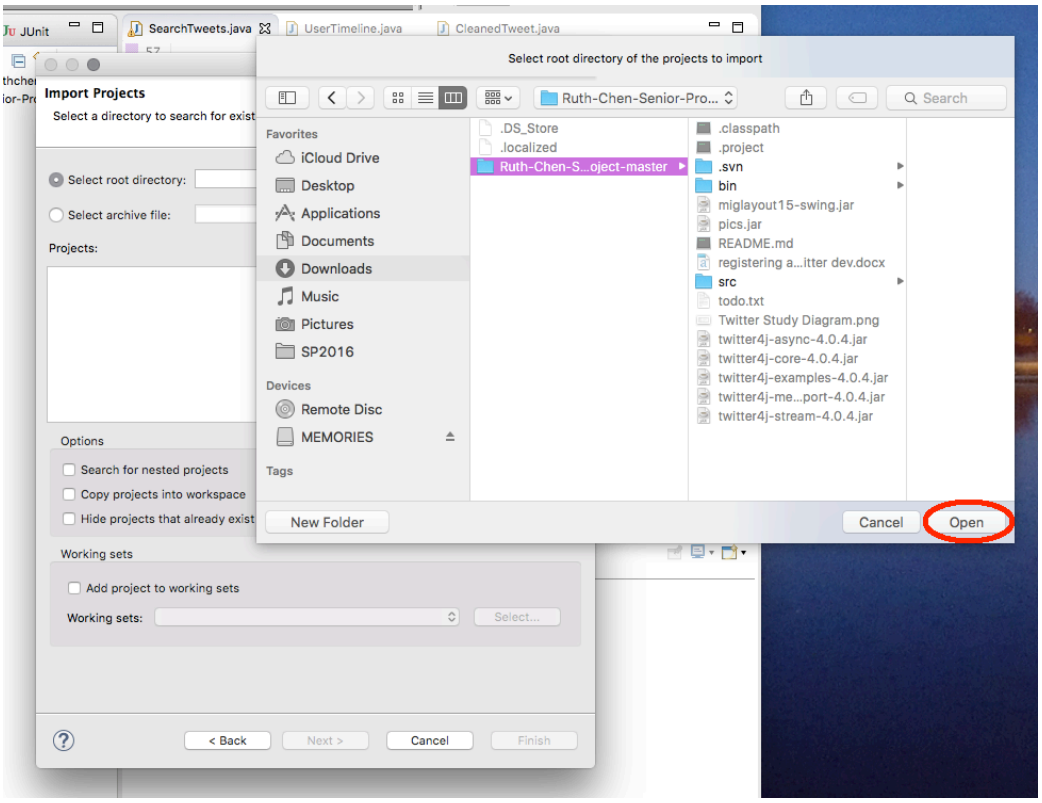
3. Go to your 'Downloads' folder, then copy the 'Ruth-Chen-Senior-Project-master' folder to Documents → workspace.
4. Open Eclipse.
5. Click on 'File', then select 'Import'.



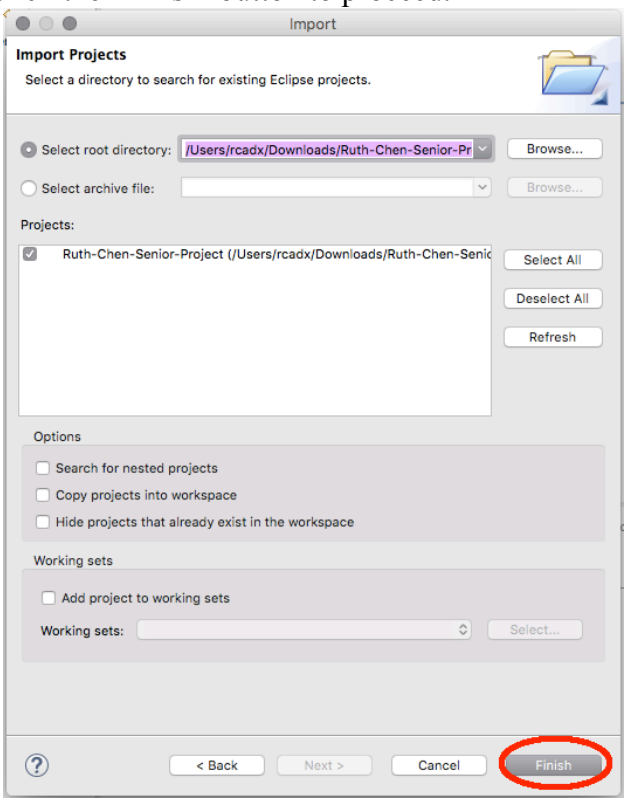
6. In the popup window that appears, double click on ‘General’. You’ll see a list of options underneath the ‘General’ folder. Click on ‘Existing Projects into Workspace’. Be sure to click the ‘Next’ button to proceed.



7. Find the ‘Select root directory’ box. Click the ‘Browse’ button. Navigate to your Downloads folder, then select ‘Ruth-Chen-Senior-Project-master”. Click the ‘Open’ button.

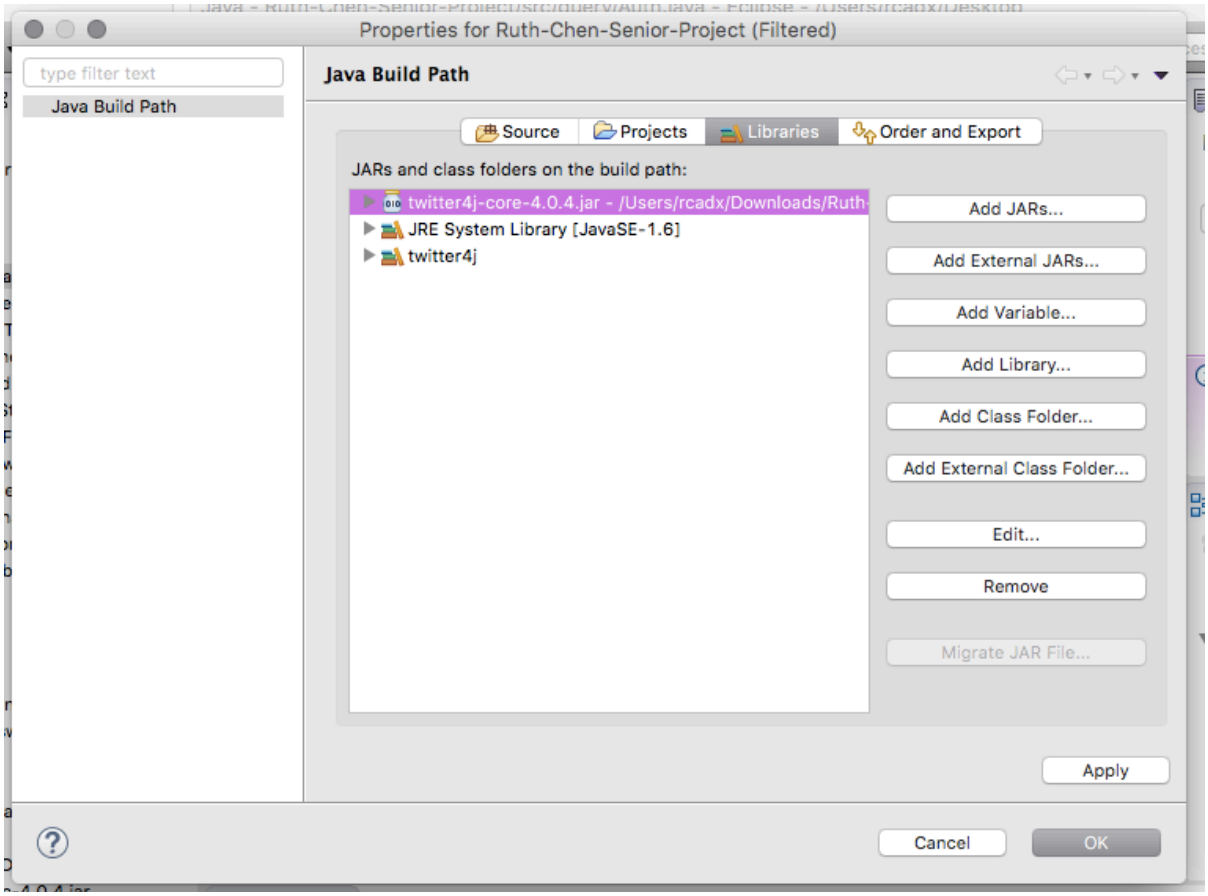


8. On the next screen, click the ‘Finish’ button to proceed.



9. If you see red Xs, go to the Potential Errors section on page 15.

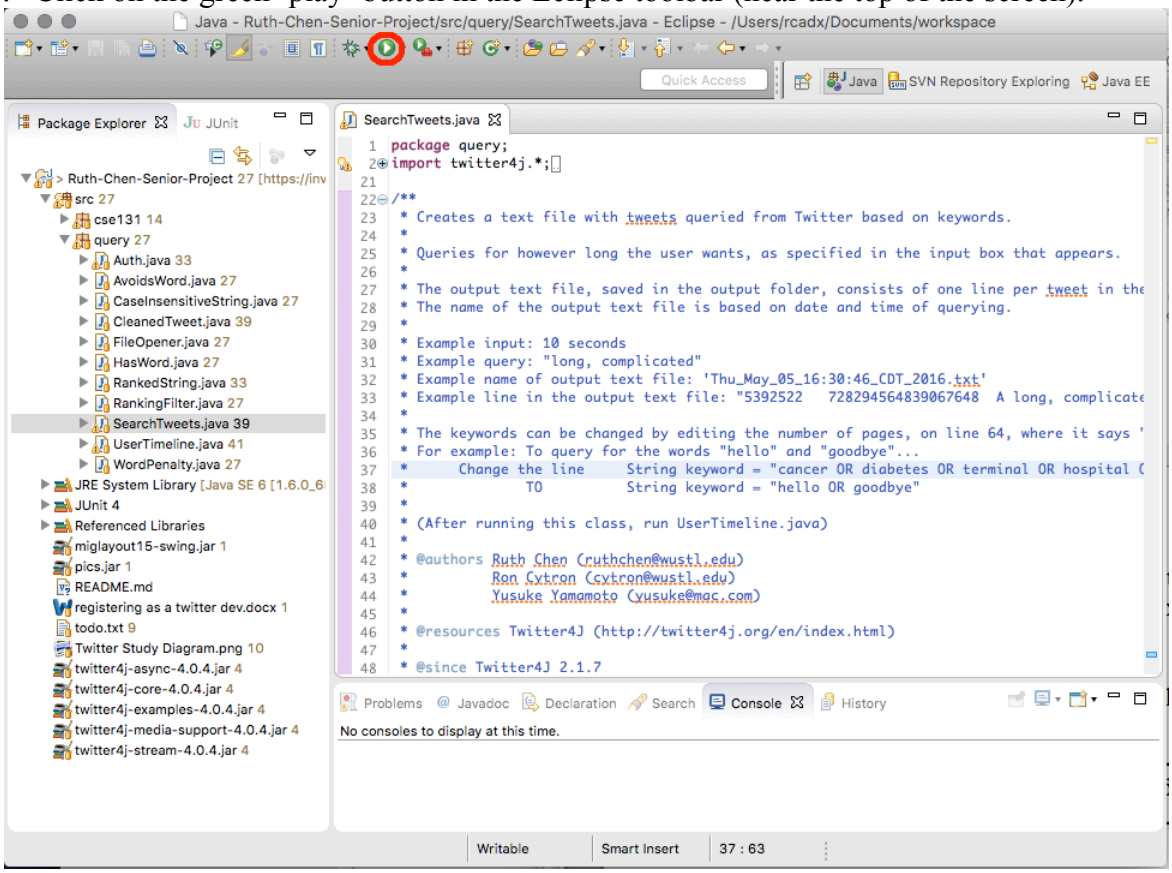
10. Your files should now be successfully loaded and ready to run! If you still have errors, please email ruthchen@wustl.edu or rchen2016@gmail.com with questions.


Properties for Ruth-Chen-Senior-Project (Filtered)
type filter text
Java Build Path
Java Build Path
Source
Projects
Libraries
Order and Export
JARs and class folders on the build path:
twitter4j-core-4.0.4.jar - /Users/rcadx/Downloads/Ruth
JRE System Library [JavaSE-1.6]
twitter4j
Add JARs...
Add External JARs...
Add Variable...
Add Library...
Add Class Folder...
Add External Class Folder...
Edit...
Remove
Migrate JAR File...
Apply
Cancel
OK

Running Queries

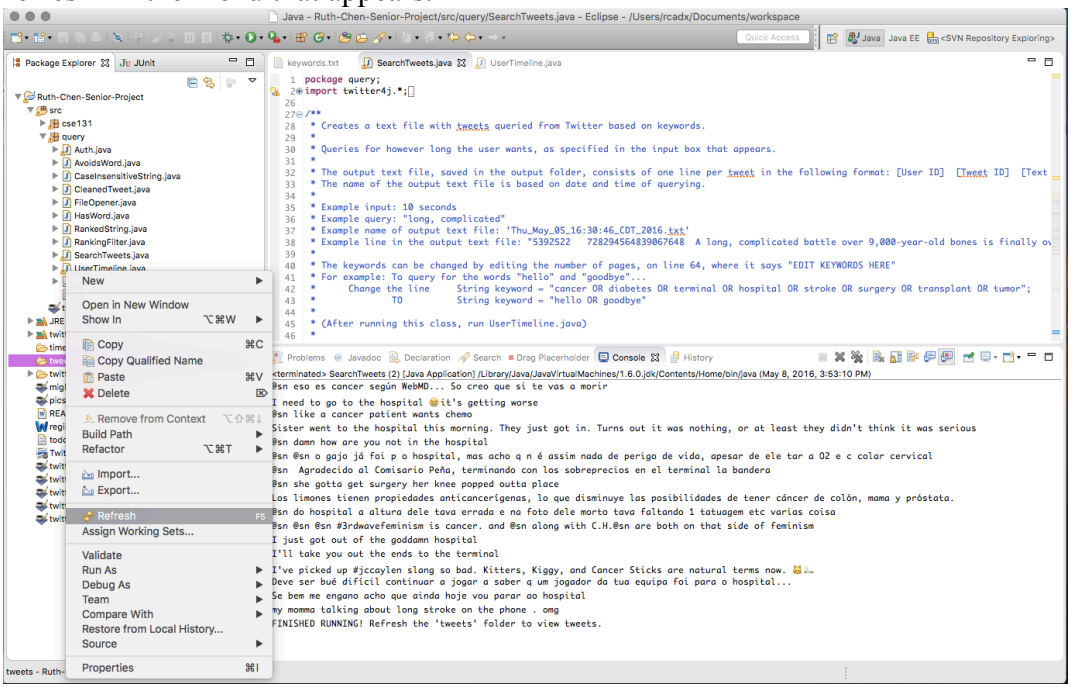
To collect the initial set of tweets using health-related keywords, we'll be running queries in Eclipse. The file we use in this section is 'SearchTweets.java'.

1. Open Eclipse.
2. In the Package Explorer on the left side of Eclipse, double click on 'Ruth-Chen-Senior-Project' to expand the folder.
3. Double click on the 'src' folder. You'll see two folders: 'cse131' and 'query'.
4. Double click on the 'query' folder. You'll see a list of files that end in '.java'.
5. Double click on 'SearchTweets.java'. The code will load in the main screen of Eclipse.
6. Click on the green 'play' button in the Eclipse toolbar (near the top of the screen).

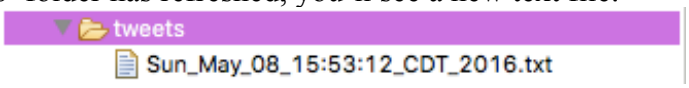


7. A popup box will appear, asking "How many seconds?" Enter however many seconds you want the program to run for. For example, if you don't have much time but want to run a quick query, you could run it for 30 seconds. If you'll be with your laptop for a while, you could run it for 3,000 seconds.
8. While the program is running, you'll see tweets as they appear in the 'Console' tab of Eclipse. This is just for your entertainment—you'll get a text file of these tweets too!
9. Once the program ends, you'll see the words "FINISHED RUNNING! Refresh the 'tweets' folder to view tweets" in the 'Console' tab.

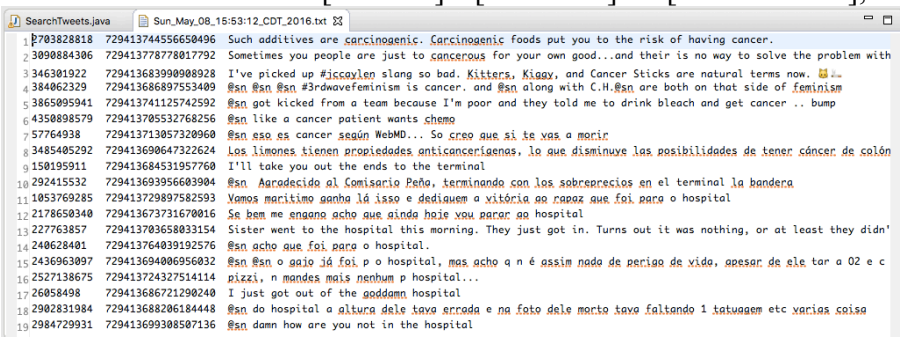
10. Right click the ‘tweets’ folder in the Package Explorer (left panel of Eclipse). Click on ‘Refresh’ in the menu that appears.



11. Once the ‘tweets’ folder has refreshed, you’ll see a new text file:



12. Double click on the text file to open it and view your tweets! Each line of the file is one tweet. The format of each line is [User ID] [Tweet ID] [Text of tweet], as follows:



NOTE #1: You don’t have to worry about the user ID or tweet ID. They will be processed by the computer in the next step!

NOTE #2: The tweets in this file are sorted by relevance. The tweets at the top are the ones that are more likely to be relevant to what we want. This is determined by ranking filters. You can change how each word gets weighted—we’ll talk about this in the Customization section later!

Getting User Timelines

Now that we have the initial set of tweets related to our keywords, we can pull timelines from specific users to analyze their tweets in depth. The file we use in this section is 'UserTimeline.java'.

You must run the queries, as described in the previous section, before running this file. This is because we use the text file that we get in the previous section to identify users to analyze in this section.

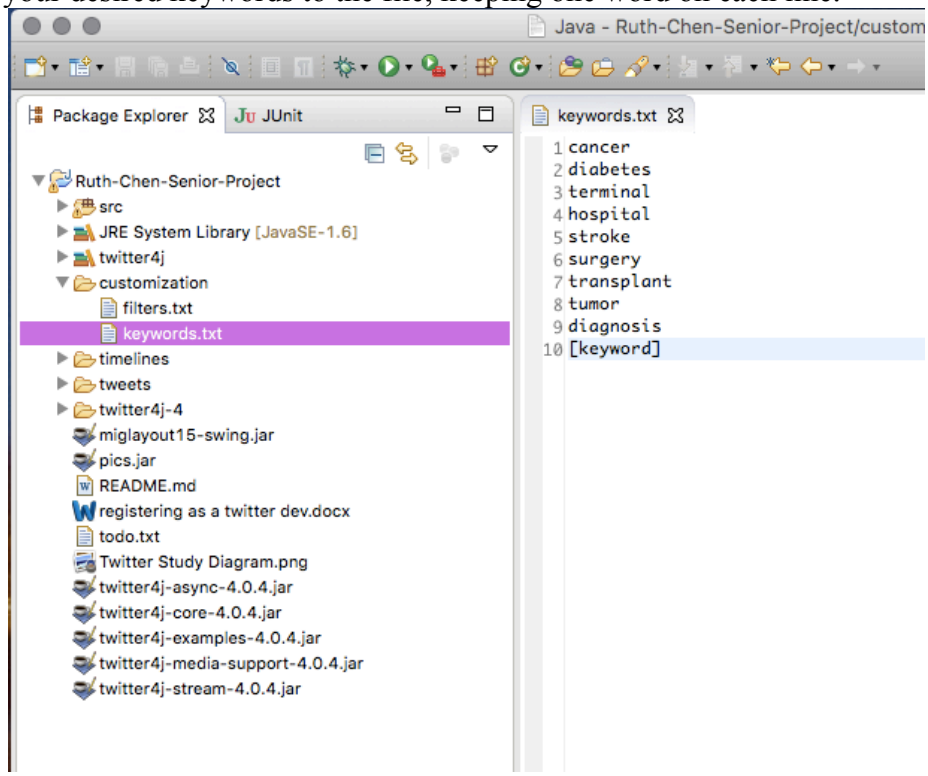
1. Open Eclipse.
2. In the Package Explorer on the left side of Eclipse, double click on 'Ruth-Chen-Senior-Project' to expand the folder.
3. Double click on the 'src' folder. You'll see two folders: 'cse131' and 'query'.
4. Double click on the 'query' folder. You'll see a list of files that end in '.java'.
5. Double click on 'UserTimeline.java'. The code will load in the main screen of Eclipse.
6. Click on the green 'play' button in the Eclipse toolbar (near the top of the screen).
7. A popup box will appear, asking you to select a file from the file browser.
8. Click into Documents → workspace → Ruth-Chen-Senior-Project → tweets, then select the file (from querying) you want to run.
9. The program will run until it finishes saving every user's timeline into a text file. The current user whose timeline is being saved is displayed in the 'Console' tab: "Now grabbing tweets from user XXXXXXXXXX".
10. Once the program ends, you'll see the words "FINISHED RUNNING! Refresh the 'timelines' folder to view tweets" in the 'Console' tab.
11. Right click the 'timelines' folder in the Package Explorer (left panel of Eclipse). Click on 'Refresh' in the menu that appears.
12. Once the 'timelines' folder has refreshed, you'll see a new text file for every user whose tweet was in the text file you got from querying (in the previous section). For example, if you queried for "hospital" and got a file with 10 tweets, you'd now have 10 new files. Each of these files contains around 3,000 tweets for each user. These tweets appear in order of most recent to least recent.

Customization

There are three things that can be customized: (1) keywords to search for, (2) ranking filters, and (3) how many tweets to grab from each user's timeline.

Customizing keywords

1. Open Eclipse.
2. In the Package Explorer on the left side of Eclipse, double click on 'Ruth-Chen-Senior-Project' to expand the folder.
3. Double click on the 'customization' folder.
4. Double click on the 'keywords.txt' file.
5. Add your desired keywords to the file, keeping one word on each line.



6. Press the 'Control' + 's' keys to save the file.

Customizing ranking filters

1. Open Eclipse.
2. In the Package Explorer on the left side of Eclipse, double click on 'Ruth-Chen-Senior-Project' to expand the folder.
3. Double click on the 'customization' folder.
4. Double click on the 'filters.txt' file.

5. On each line, you'll see a word, a tab, then a number. The word is a word you want to filter your text for and the number is how much weight that word carries. For example, if you wanted to avoid tweets that say "donation", you'd add a new line to this file:

   donation -100

   The number -100 means if a tweet has the word "donation", its ranking will be brought down by 100.

   If tweets with the word "cancer" are more likely to be relevant to what you want, you'd add this new line:

   cancer 100

   NOTE #1: Be sure to add a tab, NOT a space between the word and the number rating. Keep each word on a single line and don't add extra line breaks!

   NOTE #2: The ratings of each word are proportional to others. It's completely up to you how you want to weigh each word.

Customizing number of tweets from a user's timeline

1. Open Eclipse.
2. In the Package Explorer on the left side of Eclipse, double click on 'Ruth-Chen-Senior-Project' to expand the folder.
3. Double click on the 'src' folder. You'll see two folders: 'cse131' and 'query'.
4. Double click on the 'query' folder. You'll see a list of files that end in '.java'.
5. Double click on 'UserTimeline.java'. The code will load in the main screen of Eclipse.
6. Go to line 87 and look for the line *for(int pageNum = 1; pageNum < 21; pageNum++)*.

```
81              PrintStream ps = new PrintStream(fos);
82
83              Twitter twitter = Auth.auth(); //authentication
84
85              int tweetNumber = 1; //counter
86
87              for(int pageNum = 1; pageNum < 21; pageNum++){
88                  //gets tweets AFTER specified tweet and prints them to the file
89                  Paging afterPage = new Paging(pageNum, 500).sinceId(tweetID); //gets tweets after the specified tweet
90                  List<Status> afterTweets = twitter.getUserTimeline(userID, afterPage); //list of tweets
91
92                  for (Status tweet : afterTweets) {
93
94                      if(!tweet.isRetweet()){
```

7. Change the part that says *pageNum < 21* to *pageNum < [the number of pages of a user's timeline you want + 1]*. For example, if you wanted to get 30 pages, you would change this line to say *pageNum < 31*.

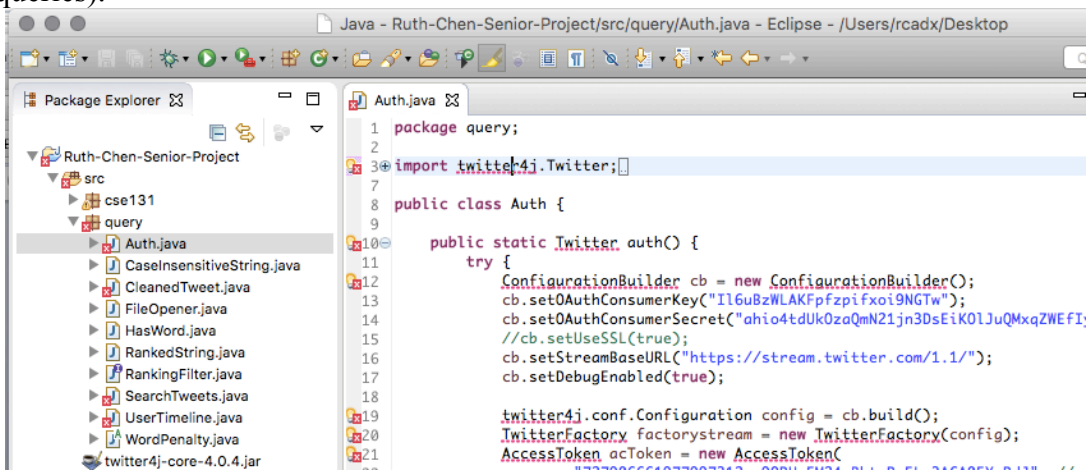
```
for(int pageNum = 1; pageNum < 31; pageNum++){
    //gets tweets AFTER specified tweet and prints th
```

Potential Errors

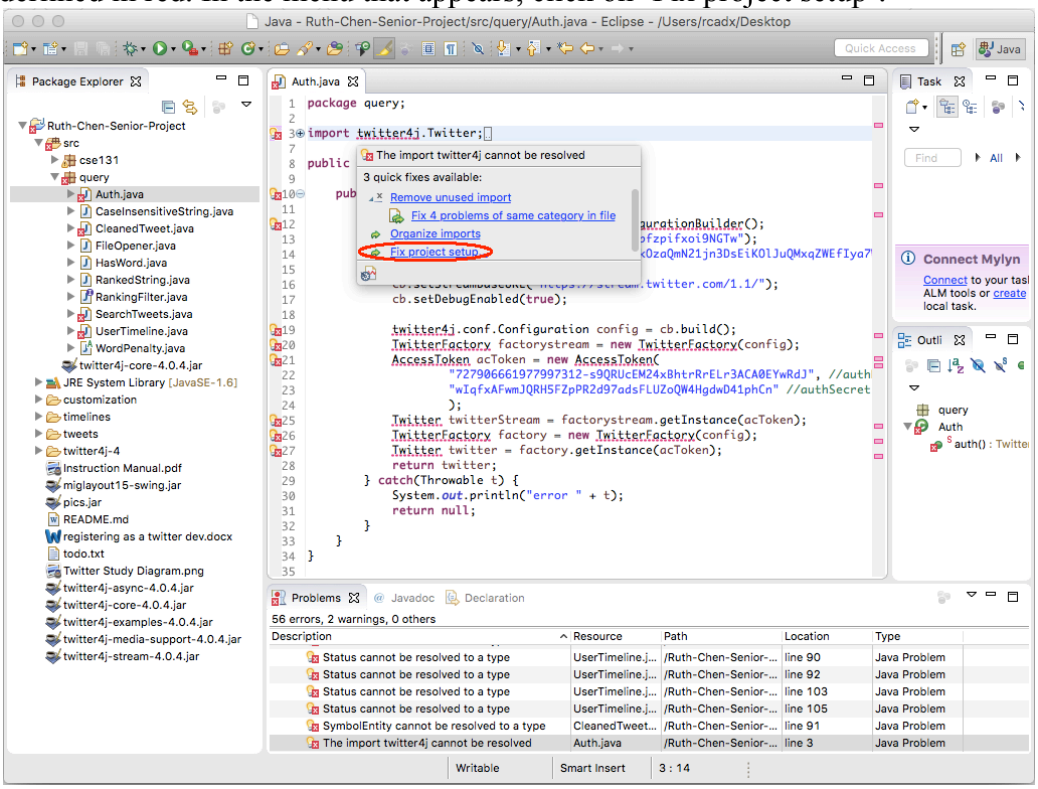
Setting up the Code – Error

If you got errors (red Xs) in your code while trying to import to Eclipse, continue reading to fix it.

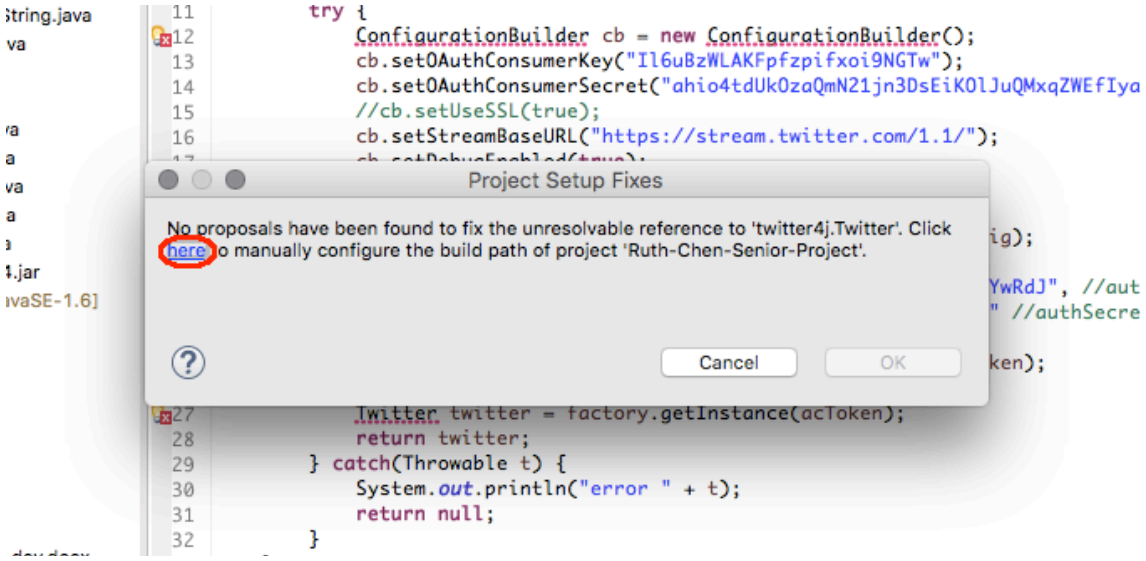
1. There are red Xs because we haven't loaded the twitter4j code (the code that runs Twitter queries).



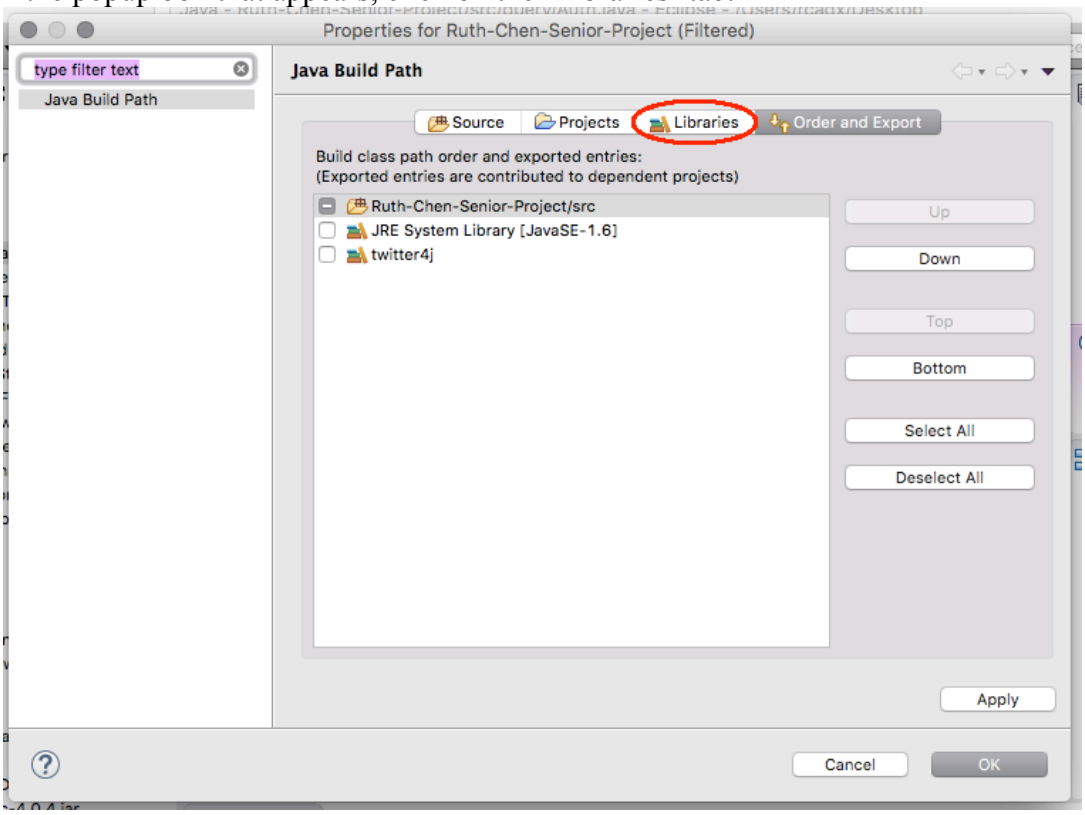
2. Click on the 'src' folder, then the 'query' folder. Double click the file 'Auth.java'. Hover your mouse over the first red X, which should be on line 3 where "twitter4j" is underlined in red. In the menu that appears, click on 'Fix project setup'.



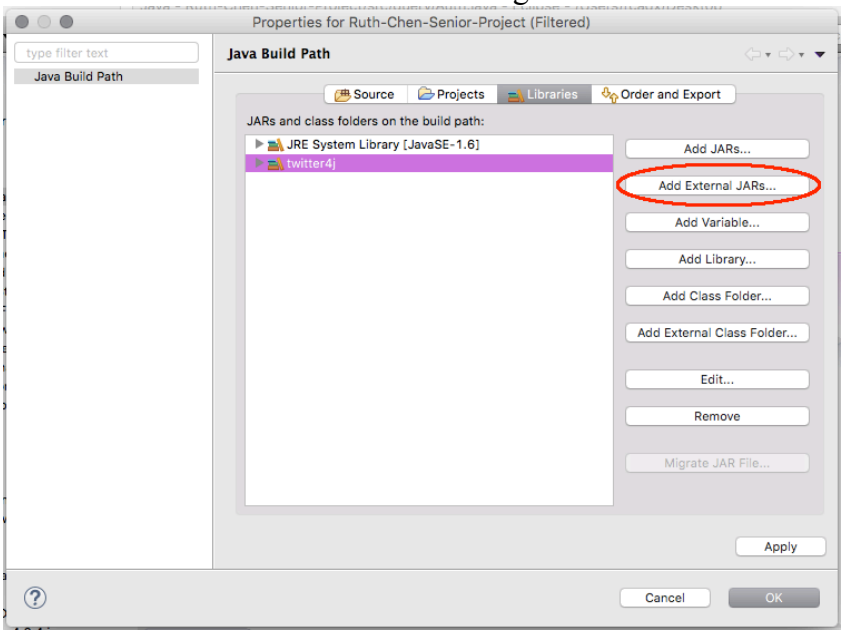
3. A popup box will appear with the words “No proposals have been found to fix the unresolvable reference…” Click on the blue link “here”.



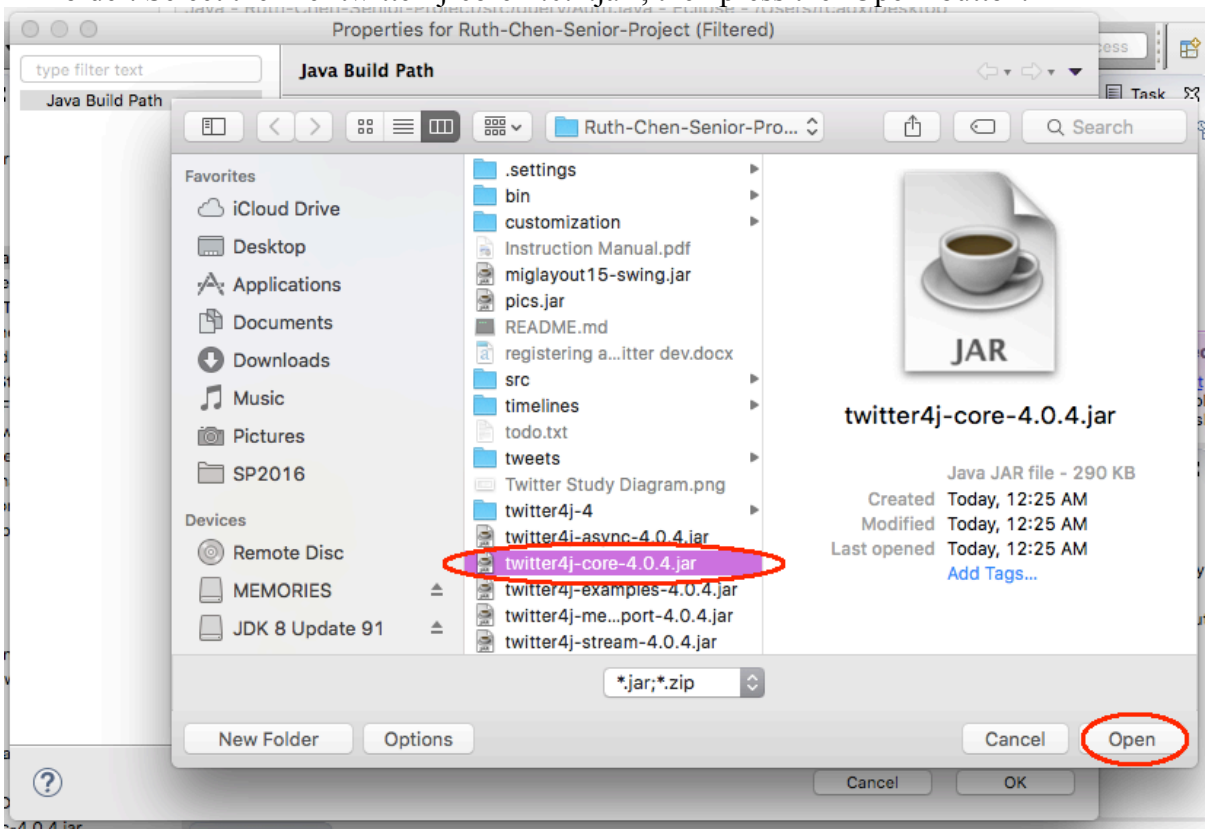
4. In the popup box that appears, click on the ‘Libraries’ tab.



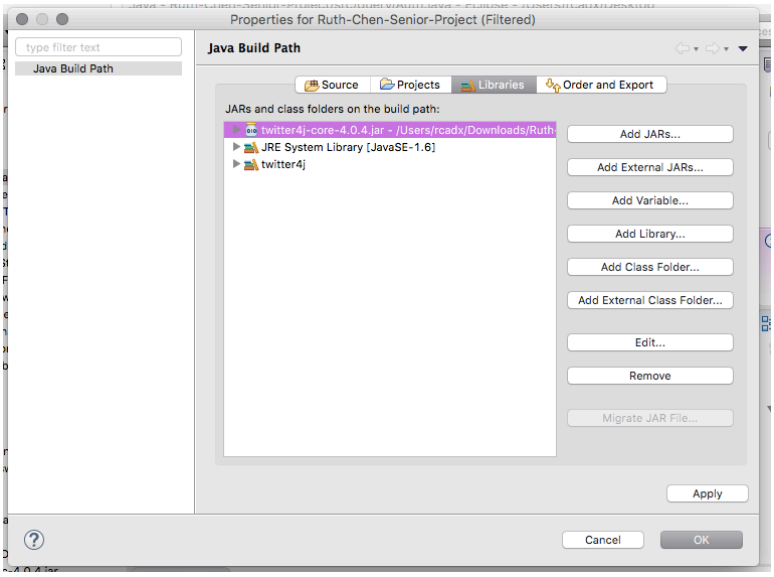
5. Click on the ‘Add External JARs” button on the right side of the screen.



6. A popup box will appear, asking you to choose a file to open. It should take you to the ‘Ruth-Chen-Senior-Project-master’ folder, but if not, navigate to it from the Downloads folder. Select the file ‘twitter4j-core-4.0.4.jar’, then press the ‘Open’ button.



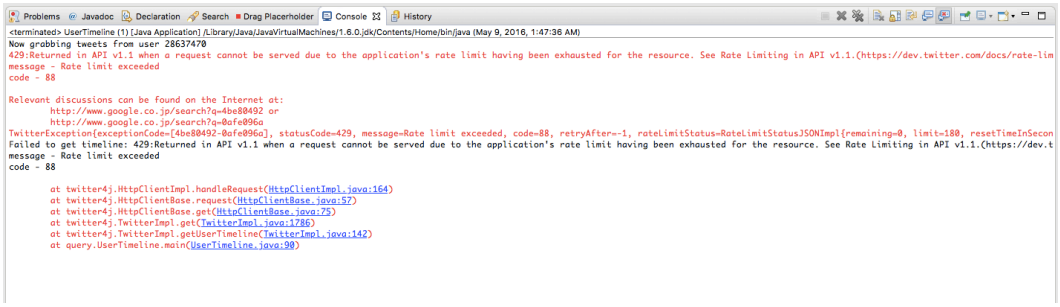
7. You'll be taken back to the previous popup box. Click the 'OK' button at the bottom to finish.



UserTimeline.java – Error

While running UserTimeline.java, you might run into this error:

Failed to get timeline: 429:Returned in API v1.1 when a request cannot be served due to the application's rate limit having been exhausted for the resource. See Rate Limiting in API v1.1.(https://dev.twitter.com/docs/rate-limiting/1.1)
message - Rate limit exceeded
code – 88



Twitter has a limit of how many tweets you can get at a time (300 requests per 15 minutes for getting user timelines).

If you get this error, this just means that too many tweets were received in too short of a timeframe. Wait 15 minutes before running the program again OR edit UserTimeline.java.

1. Open Eclipse.
2. Go to UserTimeline.java.
3. Find line 113, where it says *sleep(12000);*
4. Increase the number inside the parentheses. This will take longer for the program to run, but this means that the program will wait longer between requests. (This number is in milliseconds.)

Questions?

Email me at ruthchen@wustl.edu or rchen2016@gmail.com.