Design

# Basic design ideas:

Github provides a very detailed API. The documentation is perfect, and the subordinate classification of resources is particularly good. Therefore, I use the REST API v3 provided by Github to implement my program.

If the username exists, https://api.github.com/username will return a list of JSON formats for the details of the user. By accessing the URL, you can get the number of public repositories owed by the given user.

If the username exists, https://api.github.com/users/username/repos will return a list of JSON formats for all public repositories of the username in alphabetical order by default. The default type is owner. 30 repositories can be displayed in one page by default.

Therefore, I use URLConnection to access the above two URLs and read the page information into a string by line. Then I import the JSON package and convert the information from string to JSON object. Finally, I get the number of public repositories and the name of all public repositories using jsonObject.get(key).

Since the exception would be thrown when the URL is not found, or there is no network, the exception can be caught. Instead of print the exceptions I print some prompt messages. log4j package is used to record the exception details in the log file.

# Detailed design:

URL

Main

loadJson

searchJson

String json

String json

Repository count or

Repository name

Call method flow chart

First, create a log object. The log object reads the configuration information of the configuration file log4j.properties and creates an Exception.log file in the execution directory.

## Main method design:

Set a do-while loop. The loop condition is that flag equals to y. The program will ask user to input a value for flag. When the input is y, the main method is executed repeatedly. When the input is any other key, the program will be closed.

According to the GitHub official website, the username may only contain alphanumeric characters or single hyphens. Therefore, it should be verified that whether the input username is a valid username. If not, print the prompt message "Username is illegal. Please enter again:" and re-read the username.

If the entered username is legal, pass the URL "https://api.github.com/users/username" to the loadJson method to get the returned JSON format list for the URL.

Pass the obtained JSON format list in the String type and the key value “public\_repos” to the searchJson method. Then return the number of the public repositories of the username.

Divide the number of public repositories by 30 and round up and get the number of pages. Then get a loop and pass the URLs "https://api.github.com/users/"+username+"/repos?page=i" to the loadJson method. Get the JSON format list of the String type of each page and pass the JSON format list and the key “name” of each page to the searchJson method. Finally obtain the repositories’ name of each page, and output.

Catch the running exception and record the exception details to the log file.

After each run, output the prompt message "Do you want to query another username? y: Yes or any key: No". Then read the input and save it to the flag variable.

## loadJson method design:

Pass in the URL and return a list of JSON formats of String type.

Use URLConnection to connect the URL which is passed in. Read the contents of the URL into BufferedReader in a loop by line. Store the contents of the read into StringBuilder.

If the username does not exist, a FileNotFoundException or a MalformedURLException will be thrown when the URL content is read. Catch the exception and print message "Username does not exist! Please check your username." Record the exception details in the log file.

If the network connection fails, it will throw a NoRouteToHostException or UnknownHostException. Catch the exception and print prompt message "Please check your network connection.". Record the exception details in the log file.

If the URL connection times out, it will throw a ConnectException. Catch the exception and print message "Connection time out.". Record the exception details in the log file.

Catch other IOExceptions and record the exception details in the log file.

## searchJson method design:

Pass in the JSON format list of String type and return an ArrayList of String type.

Convert the JSON format list of String type to a JSONArray.

Traverse the jsonArray array and convert each object into a JSON object. Get the value corresponding to the key in each JSON object. Then add the obtained value to an ArrayList of String type. Finally return the ArrayList.

Catch JSONException and record the exception details in the log file.

# Reference connection:

<https://developer.github.com/v3/>

<https://developer.github.com/v3/guides/getting-started/>

<https://segmentfault.com/a/1190000015144126?utmsource=tag-newest>