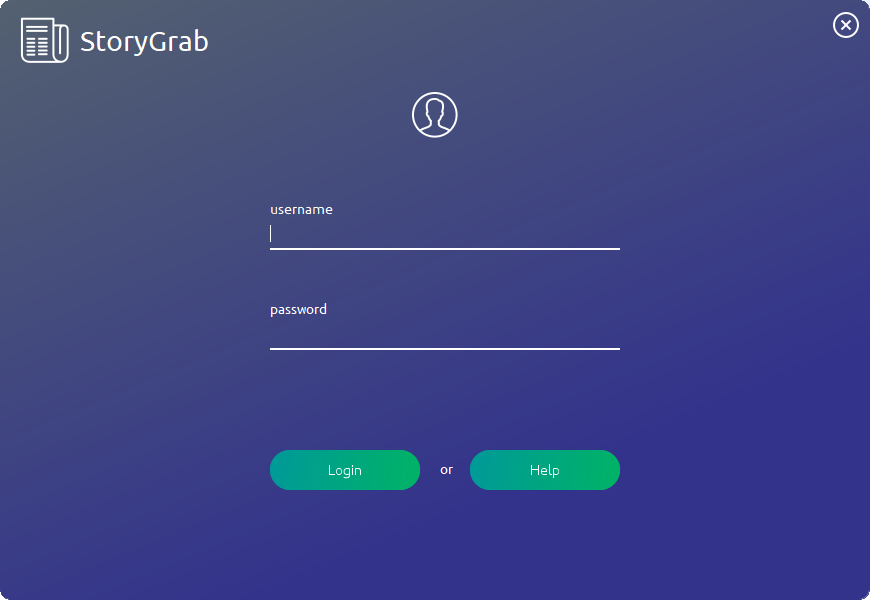
Criterion C: Development

List of Techniques

This program uses all the following techniques:

* Graphical user interface
* Global and local variables
* Methods
* Recursion
* Search algorithms
* Merging data structures
* Polymorphism
* Inheritance
* Encapsulation
* Parsing a text file
* Use of additional libraries
* Inserting data into an ordered sequential file
* Abstract data type

Graphical User Interface

The GUI, as requested by the client, shows a visual representation of what is going on in the program.

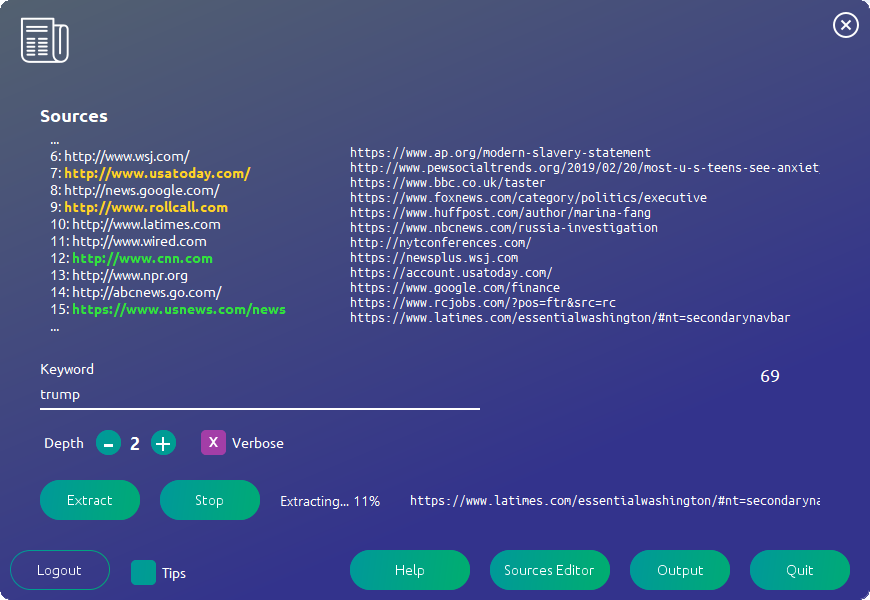
Close button

Help button

Login button

Password field

Username field



Progress label

Tips button

Logout button

Stop button

Extract button

Quit button

Output button

Source button

Help button

Depth controls

# of hits

Output line

Search field

Close button

Output pane

Sources pane

The first code piece is from the method doInBackground() in the class BackgroundRunner. It is placed here to illustrate the use of publish(List<String> chunks) to update the output panel. publish(List<String> chunks) sends chunks to process(List<String> chunks), which is reviewed in the second piece of code. This is because SwingWorker is a special class with reserved methods to be implemented. chunks is parsed and formatted to be more readable and then sent to the 3 output labels, referenced as longout, outputlbl, and hitslbl. hitslbl contains the number of search results currently found. outputlbl contains one line, the most recent message, and longout contains the 12 most recent messages.

**for**(int i = 0;i<extractors.size();i++){

Extractor t=extractors.get(i);

**if**(!t.isAlive()&&!used.contains(t)){

used.add(t);

leftToDo--;

*//set tmp vars*

double s = extractors.size();

double l = leftToDo;

double sm = s-l;

findSource(t.getWebpage()).setlinkDone(**true**);

System.out.println();

System.out.println("Done: ");

**for**(Extractor e : used){

System.out.println("\t"+e);

}

System.out.println();

percent=(int) ((sm/s)\*100);

statusLblRef.setText("Extracting... "+percent+"%"); *//set label*

}

publish(t.toBG);*//publish output to process*

**if**(!t.errorMsgs.equals("")){

publish("<font color=FFD126>"+t.errorMsgs+"</font>");

findSource(t.getWebpage()).setError(t.errorCount);

}

**if**(shouldStop){

publish("Cancelled by user - syncing threads");

**for**(Extractor ts : extractors){

publish("Extractor termination on "+ts.getWebpage());

ts.setStop(**true**);

}

**for**(Thread tt : extractors){

tt.join();

}

publish("Saving to database");

shouldStop = **false**; *//so loader can be used again*

isRunning = **false**; *//update this*

**return** **false**;

}

}

process(List<String> chunks) is a **protected** method that is meant to be modified as part of SwingWorker. Note that outputlbl, longout, and hitslbl are all of type JLabel and thus use the method setText().

**protected** void process(List<String> chunks) {

*// Get Info*

String currentOut = "";

currentOut = chunks.get(chunks.size()-1);

outputlbl.setText("<html>"+currentOut+"</html>");

int st = chunks.size()-OUTSIZE;

**if**(st<0){

st=0;

}

**if**(verbose){

String bo; *//no scroll*

bo = "<html><br>";

**for**(int i = st; i<chunks.size();i++){

String se = chunks.get(i).replaceAll("Searching ", "");

bo=bo+se+"<br>";

}

bo+="</html>";

longout.setText(bo);

}

**else**{

longout.setText("");

}

hitslbl.setText(*/\*"Hits: "+\*/*countAll()+"");

**if**(dynamic){Tools.saveToHTML(hitLinks, finalHTML, searchFor);}

}

Variables

Both global variables and private variables are used in most every class of the program, so I will not list every one out here. However, I will list ones that have