Java 1.7 encoding:UTF-8

Crawlers need to be run from the openlab cluster. You will need an ICS login to ssh into openlab.

Guidelines:

1. **Use robots.txt**. Please abide by any rules set forth in the robots.txt files. Some of the crawlers got stuck on dynamically generated pages that they should not have been crawling (or perhaps could not get away from).
2. **Use** [**openlab**](http://www.ics.uci.edu/computing/linux/hosts.php)**.** Please run your webcrawlers from computers in the openlab cluster only. Any ICS student may ssh directly into openlab.ics.uci.edu. We are more likely to block off campus traffic.
3. **Go slow with an interval > 500ms**. We're going to block clients with the UCI webcrawler string that have too many connections per seconds. We have set the limit to \*500ms\* per request. Going faster than that rate will result in them getting blocked.
4. **Tip for Long Running Processes**
   1. **Use the Sun Grid Engine (SGE)**
      1. Students may run jobs on the [ICS Sun Grid Engine (SGE)](https://swiki.ics.uci.edu/doku.php/services:sun_grid_engine:tips_and_tricks).
      2. Example job submission:
      3. qsub -q 15day.q -M luser@ics.uci.edu -m beas -o ~/crawler.$JOB\_ID.out -e ~/crawler.$JOB\_ID.err /home/luser/coursework/web-crawler.sh
   2. Please visit the [ICS Sun Grid Engine (SGE)](https://swiki.ics.uci.edu/doku.php/services:sun_grid_engine:tips_and_tricks) entry for more information on using the SGE queues.
5. **Window Manager Multiplexer**
   1. Multiplexers like screen or tmux can let you disconnect and reconnect from sessions running on openlab servers. Click [here](https://swiki.ics.uci.edu/doku.php/commands:screen) for information on [screen](https://swiki.ics.uci.edu/doku.php/commands:screen).
   2. Also see the local man pages for tmux and screen
   3. % man screen
   4. % man tmux

Main Class

* Done
  + Add methods to report everything needed in report
* TODO
  + Add crawler method to traverse each file
  + 1. For each URL in frontier, need to save it to a file, or read it
  + 2. Call Storage class to rip all tokens out of file
  + 3. At the end of calling all URL with crawler, need to summarise report

Storage Class

* Done
  + Added objects for each necessary part of project
* TODO
  + Improve code to:
    - Count longest page
    - Process subdomains
    - Process word frequenicies
    - Process 3-grams
  + Add file writer object to print to file
    - Needed for subdomains
    - Needed for answers.txt
  + Keep full vocabuluary
  + How to store each document? large files?
    - How to parse?
* Anchor text?

TextProcessor

* Done:
  + Results from each file are now cumulative
* TODO:
  + Ignore stop words when processing token list
    - Do this before computing word frequency and 3 grams
  + Modify token list method to also count tokens (required for page length)

Indexer

* Doc id to file indexer
  + How to read and write to a storage file
* Token to file indexer
  + Refers to document id
  + Very basic for now, just each token stores its own list of doc id