crawler

* Authentication
  + AuthInfo
    - Stores user/pw logins into sites
    - Three types
      * Basic
      * Form
      * NT
  + Basic, Form, NT AuthInfo
* Exceptions
* <none>
  + Configurable
    - Abstract class for configurable stuff
  + **CrawlConfig**
    - *Stores many configuration settings for the crawler*
    - resumeCrawlingFromPrevious
    - crawlingDepth
      * unlimited = -1
    - maxPagesToFetch
      * unlimited =-1
    - userAgentString
      * set to student ID
    - politeness delay
      * set to 300
    - includes
      * http, binary, connections, timeouts, max download size, redirects
    - set crawl storage folder
  + **CrawlController**
    - *Creates crawl threads and manages the crawling session*
    - *There are existing implementations of crawlController: basic, image, multiple in the test directory*
    - ***This is the main class! Call this to run everything!***
    - Start
    - WaitUntilFinish
    - Sleep
    - addSeed
    - addSeenUrl – assign a document id to a url
    - robottxtserver methods
    - frontier methods
    - docId servers methods
    - is finished/shut down
  + Page
    - Data of a fetched page
      * url, redirectFlag, redirectUrl, status, contentType
    - load
      * loads the contents from a HttpEntity
  + WebCrawler
    - *Runnable thread executed by crawler, expected to be implemented*
    - *There are existing implementations of the WebCrawler in the test folder*
    - crawlerId, thread, parser, fetcher, robotstxtserver instance, docIdServer, frontier
    - init
    - onstart
      * setting up data before crawler starts
    - onBeforeExit
      * setting up data before crawler exits
    - handle pageStatusCode, UrlBeforeProces, Exceptions
    - getMyLocalData
      * called before terminating thread
      * passes local data to the controller
      * should be compiled into a list by controller for processing
    - visit a page
      * process content of fetched page
    - should visit
      * tell crawler if a URL should be visited
    - process page (url)

Definitions

* list of stop words
* list of starting URL
  + contains ics.uci.edu

Fetcher

* IdleConnectionMonitorThread
* PageFetchResult
  + Fetches content of page
  + Has results of the page
* PageFetcher
  + Fetches pages
  + Does authentication logins

Frontier

* Counters
  + Can increment value
  + Coulds scheduled vs processed pages
  + Used by Frontier to count stuff
* DocIDServer
  + Database of docID
    - Each seen URL has a docId
  + Can give new URL a new docId
* Frontier
  + Schedules URLs
  + Processes them
  + Used by CrawlController/WebCrawler
* InProcessPagesDB
  + DB for each processed page, removes the pages from DB environment
* WebURLTupleBinding
  + Binds objects to entries, used by workqueues
* WorkQueues
  + Used by Frontier
  + Gets url’s and does transactions on them

Parser

* Binary Parser Data
  + Parses HTML binary into encoding
* Extracted URL Anchor Pair
  + Href, anchor, tag
  + Used by parser
* HTML Content Handler
  + Handles HTML
* NotAllowedContentException
* Parser
  + Parses pages, used by WebCrawler
* TextParserData
  + Has text content and URL’s

RobotsTxt

* Has directives from host
  + What is allowed
  + What needs refetching
  + What is disallow
  + Last access time
* **Config**
  + Enabled
  + userAgentName // do I need to set this?
  + Cache size // number of hosts to cache
* Parser
  + Parses content of robot.txt
* Server
  + *Used by CrawlController/WebCrawler*
  + Tells if a page is allowed to be fetched
  + Get directives from a page

**Storage**

* Hold all the information we need to turn in assignment
* Must populate with data after parsing each page
* Needs file handling for storage stuff

**Text Processor**

* Generated token list and processes text

URL

* TLDList
  + Singleton that stores list of TLD
* URLCannicalizer
  + Does URL normalization
  + Creates URL and generates parameters
* URL Resolver
  + Resolves a URL against the base URL
  + Used by URL Cannoicalizer
  + Parses, indexes, stores URL properties (location, path, parameters, query)
* WebUrl
  + Container for URL
  + Doc id, url, depth, domain, subdomain, etc

Util

* IO
  + Delete folder, folder contents
  + Used by CrawlController if we are not resuming from previous session (deletes all prev session data)
* Net
  + Extracts set URLs from an input
    - Finds all url from chunk of text and adds to a list
* Util
  + Has stuff to convert binary to content

**Main**

* Runs the crawler
* Calls storage to document the data
* Prints out the data from storage