

Usage Instructions

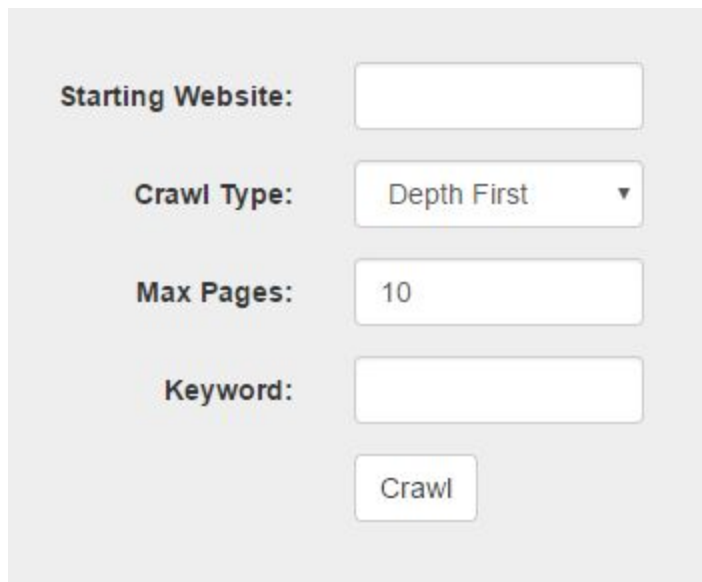
Visual Web Crawler App Usage Instructions

Navigate to the Web Page

The location of our hosted web app is: <http://54.201.94.228:8080/>

Enter the Crawl Settings

Underneath the saved cookie crawls HTML select field, you will find a form containing crawl options that looks as follows:

A screenshot of a web application interface for configuring a crawler. It features four input fields stacked vertically: 'Starting Website' (a text box), 'Crawl Type' (a dropdown menu currently showing 'Depth First'), 'Max Pages' (a text box containing the number '10'), and 'Keyword' (a text box). Below these fields is a 'Crawl' button. The entire form is set against a light gray background.

The crawler input fields are as follows:

- **Starting Website** - The URL that you would like to start the crawl at. This field is always required.
- **Crawl Type** - You may specify either a depth-first or breadth-first crawl with this options element. Note that depth-first search may end sooner than the number of pages that you specify if the crawler is unable to find any links on a given page.
- **Max Pages** - The number of pages other than the starting page that the crawler should attempt to crawl. The minimum number of pages is 0, and the maximum number of pages is 1000. Note that attempting to crawl pages that have not been cached in our server's database will take ~1-2 seconds per page, but previously crawled results usually load into the web app in less than 5 seconds.
- **Keyword** - This optional field specifies that the crawler will attempt to find a keyword of your choosing (without whitespace) in the string contents, the tag names, the tag IDs,

and the class names, and the string contents of an HTML page. Note that crawls performed with the keyword option will take longer than other crawls because the crawler cannot appeal to information in the datastore in order to find a keyword.

When you are ready to begin the crawl, press the “Crawl” button. When the crawl has completed, its data will be displayed as a tree in the SVG visualization, and a log of crawl results in the same form that the crawler returns will be displayed in the results text box below the visualization.

After a crawl has been completed, the crawl arguments as they are given to the crawler itself are stored in a select HTML statement directly above the input form we used before:



The screenshot shows a web interface with a label "Saved Crawls:" followed by a dropdown menu. The dropdown menu is open, showing the selected value "www.google.com -d -l 10". Below the dropdown menu is a button labeled "Crawl Again".

For more information on the arguments, please see the Crawler Usage Information subsection in the below section. This list of options represents crawl configurations that have been stored as a cookie on your browser. Once you have performed at least one crawl using our app (without deleting your browser cache), you can use the Saved Crawls field to reuse a crawl configuration that you have already performed. Simply press the “Crawl Again” button, and the crawler will obtain results that will be displayed in the visualization and the result log text box as before.

Crawler Standalone Usage Instructions

The following set of instructions applies to a Windows 8 installation of the crawler on a 64-bit machine:

Download and Install Python 2.7

- Download the latest version of Python 2.7 (2.7.13) from <https://www.python.org/downloads/> and follow the default installation instructions for the .msi file.

Download and Install MongoDB

- Download and install the latest version of MongoDB Windows Server 2008 R2 64-bit and later, with SSL support x64 available at:
 - <https://www.mongodb.com/download-center#community>
- Follow the installation instructions available at

- <https://docs.mongodb.com/v3.0/tutorial/install-mongodb-on-windows/#install-mongodb>

Download and Install pip

- Download and install the latest version of the pip Python package manager by following the instructions available at:
 - <https://pip.pypa.io/en/stable/installing/>

Download and Install lxml

- Download and install the latest version of the lxml by following the instructions given at:
 - <http://lxml.de/installation.html>

Install Crawler External Dependencies

Open a command window and type in the following commands:

- 'pip install requests' - Installs the Requests HTTP library for Python.
- 'pip install BeautifulSoup4' - Installs the BeautifulSoup HTML tree parser and data processor.
- 'python -m pip install pymongo' - Installs the PyMongo Python driver for MongoDB.

Start a Local MongoDB Instance

- Navigate to the directory containing MongoDB binary files (likely C:\Program Files\MongoDB\Server\3.4\bin unless a different directory was specified during installation).
- Open a command window within this directory.
- In the command window, type 'mongod'. This will start the local MongoDB instance. Leave this command window open for now.

Start the Crawler

- Download the zipped crawler and server files to a directory of your choosing.
- Navigate to the directory containing the crawler and server, and navigate to the 'crawler' folder.
- Open a command window within this directory.
- To receive verbose output for each crawled page, go to crawler/lib/constants.py and ensure that the 'DEBUG_LOG_MODE' constant is set to True.
- In the command window, type 'python crawler.py <starting_url> -d'. This will cause the crawler to attempt to perform a depth-first crawl of 10 pages (the default number of pages to crawl) starting at <starting_url>.

Crawler Standalone Output Format

The crawler provides log information on robots.txt file acquisition, downloading, and caching, as well as information about URLs that have been scraped from a particular page as well as the next page to be processed by the URL queue. This information is output via STDERR.

The crawler also provides its final result in the form of a log file identified by 'CRAWLER_OUTPUT_<process_id>' and prints this log file via STDOUT. Ultimately, this is the information that the web server passes along to the client browser at the end of the crawl.

Individual log records conform to the following JSON/BSON format:

```
{
  'parent' : '<parent_url>',
  'child'  : '<child_url>',
  'height' : <height_int>,
  'keyword_found' : <keyword_boolean>,
  'title'  : '<title_string>'
}
```

'parent' and 'child' denote a single link between vertices in the crawl tree, and 'height' indicates the tree depth from the root at which the child node can be found. The root node record is specified by setting 'height' to 0 and 'parent' to null.

The 'title' field represents the title of the child node. If a keyword search is not being performed, the 'keyword_found' attribute will be set to false for each node (in order to make it easier for the data visualizer to process records). If a keyword search is being performed and the keyword is not found, the 'keyword_found' attribute will be set to false as well. The 'keyword_found' attribute is set to true if the keyword has been found in the child node's HTML text contents, tag names, tag ids, or class names.

Crawler Usage Information

The command line arguments for the crawler are as follows:

```
python crawler.py -d | -b <tree_height> [-l <crawl_limit>] [-k
<keyword>]
```

Options:

-d : Use DFS as the crawl search algorithm

-b <tree_height> : Use BFS as the crawl search algorithm. <tree_height> specifies the maximum height of the BFS crawl tree. (Note: Implementation upcoming)

-l <crawl_limit> : (Optional) Specify a non-default limit for the number of pages to crawl. A default page limit is used otherwise (specified in crawler/lib/constants.py as CL_DEFAULT_PAGE_LIMIT.

-k <keyword> : (Optional) Specify a keyword that halts the crawl when found on a page. Note that keywords must not contain whitespace, and the keyword search is case insensitive.