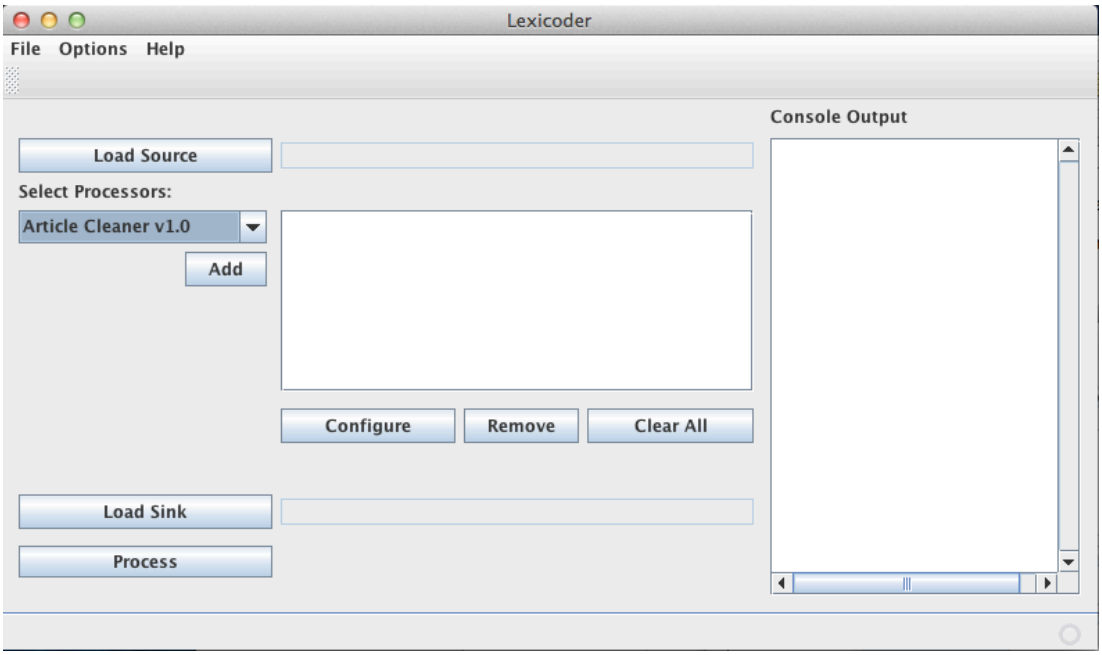
lexicoder v 2.0 User's Manual

Lexicoder is a Java-based, multi-platform tool for automated content analysis. To install Lexicoder on your computer (PC or Mac), you need only unzip the Lexicoder.zip file. It can be saved in any location on your computer. To start Lexicoder, you need to double-click on the Lexicoder.jar file.

Lexicoder will typically take a few moments to start. If it does not start, or reports an error, please be sure that you have the latest version of Java installed on your computer.

Once it's up and running, the Lexicoder interface is very simple:



There are four steps to analyze data in Lexicoder — each represented by one of the Button on the left of the interface: Load Source,
Select Processors [Add],
Load Sink, and
Process.

The Console Output window confirms each of these processing steps.

We'll describe how to use Lexicoder here first by describing briefly each of the four steps in the process, and then by walking step-by-step through a content analysis of an example dataset, using an example dictionary.

Section 1. The Four Steps

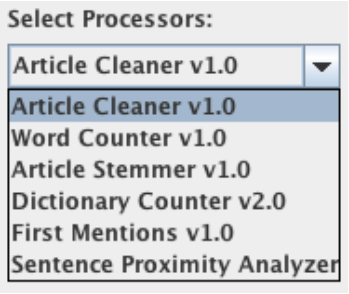
1. Load Source

Select the file that you wish to analyze. This file should be plain, tab-delimited text. It should have one column with case IDs (labelled "ID"). That ID will be saved alongside results, and you will need that ID in order to merge those results with your original datafile. It should also have one column labelled "Body" — this is the column containing the data that Lexicoder will analyze.

2. Select Processors

This is the stage at which you select the various processors you would like to use on your data. There are several options listed in the drop-down menu, and they can be added in any order you wish.

The current version of Lexicoder includes six processors, available through the drop-down window as follows:



2A. Article Cleaner v1.0

This is typically the first processor in any analysis. The cleaner goes through your dataset and, by replacing and re-shuffling punctuation marks, tries to reduce errors in subsequent analyses. (Commas, decimals and periods, question marks, dashes and the like can affect word counts and the identification of sentences, for instance.)

The following is a list of the conversions that the current article cleaner (v2.0) performs:

Original Text	Clean Text
Mr.	Mr
Mrs.	Mrs
U.S.	US
U.S.A.	USA
U.S.S.R.	USSR
Rep.	Rep
Dem.	Dem

M.P.	MP
M.P.P.	MPP
Dr.	Dr
‘s	s
,	(replaced with a space)
:	(replaced with a space)
;	(replaced with a space)
(	(replaced with a space)
)	(replaced with a space)
[	(replaced with a space)
]	(replaced with a space)
!	(replaced with a space)
?	(replaced with a space)
“	(replaced with a space)

Some text will require much more pre-processing. This can be accomplished with any text browser that includes a find-and-replace mechanism.

Note that this is the only module in this release that actually changes the contents of the article which is processed. Thus, the ordering of this particular module matters. Putting (for example) the dictionary counter before or after the word counter will not impact the results for those modules; placing the article cleaner before or after the dictionary counter, however, will result in different outcomes.

2B. Word Counter v1.0

The word counter provides a simple count of the number of words in the text.

2C. Article Stemmer v1.0

This is the Porter Stemming Algorithm, developed by Martin Porter. Information is available at http://tartarus.org/~martin/PorterStemmer/.

2D. Dictionary Counter v2.0

This is the most critical module for most users: it counts the occurrences of words specified in a dictionary. The dictionary must meet certain specifications (see samples below). It must be formatted in XML, and it must be two-leveled. If the dictionary is not properly formatted, Lexicoder will not be able to work with it (and may not be able to tell you. . .)

When designing a dictionary, it is important to keep the following processing details in mind:

- The Dictionary Counter goes through each category in the dictionary sequentially (first level entries)

- It goes through each word or phrase in the category and checks to see if it exists in the article
- It counts (and adds up) each existence of the word or phrase in the text.
- As it counts, each existence of the word or phrase is removed from future consideration by the module (though all content will remain for other modules). This is to improve efficiency as well as to properly handle phrases. Thus, if a word or phrase occurs in two categories, the dictionary counter will only count it as belonging to the first category.

By way of example, image that you have dictionary that captures two categories: animals, and colors. The dictionary, in the correct xml format for Lexicoder, might look like this:

```
<?xml version= "1.0" encoding=UTF-8" standalone= "no"?>
<dictionary style= "Lexicoder" name= "Test Dictionary">
<cnode name= "Animals">
<pnode name= "fox" />
<pnode name= "cow" />
 <pnode name= "dog" />
</cnode>
<cnode name= "Colours">
<pnode name="black" />
<pnode name="brown" />
<pnode name= "red" />
</cnode>
</dictionary>
```

Now, if your text looked like this:

The quick brown fox jumps over the lazy dog.

Then the results of the analysis would look like this:

Article ID	Body	Animals	Colours
1	The quick brown fox jumps over the lazy dog.	2	1

There are two animal words (fox, dog) and one color (brown).

2E. First Mentions v1.0

This module uses the same kind of dictionary files as the Dictionary Counter, but in this case it captures the character at which a given word begins. The processor is useful in instances in which the research wants to know, for instance, whether the article mentions Democrats first or Republicans first. The processor returns the number of characters in the text up to and including the first character, for

the first instance in which in a word included in a given dictionary appears. Smaller numbers, then, indicate that the word appears earlier in the article.

2F. Sentence Proximity Analyzer v1.0

This module uses the same kind of dictionary files as the Dictionary Counter, but in this case it counts the occurrences of words specified in one dictionary that appear in the same sentence as words specified in another dictionary. The analyzer requires some careful pre-processing of the data because sentences are identified using periods, so all the extra periods (in "Mr." for instance) have to be removed before you can run this module reliably. (See the Article Cleaner, above.)

Image you are using a similar dictionary to the one used above, though this time just for colors:

```
<?xml version= "1.0" encoding=UTF-8" standalone= "no"?>
<dictionary style= "Lexicoder" name= "Color Dictionary">
<cnode name= "Colours">
<pnode name="black" />
<pnode name="brown" />
<pnode name= "red" />
</cnode>
</dictionary>
```

The Sentence Proximity Analyzer asks you to define a second dictionary — the dictionary the words with which animals and colors need to co-occur in order to be counted. Your second dictionary is as follows — it identifies animals:

```
<?xml version= "1.0" encoding=UTF-8" standalone= "no"?>
<dictionary style= "Lexicoder" name= "Animal Dictionary">
<cnode name= "Animals">
<pnode name= "fox" />
<pnode name= "cow" />
<pnode name= "dog" />
</cnode>
</dictionary>
```

Now, if your text looked like this:

The quick brown fox jumps over the lazy dog. At least, I though it was brown; it might have been red.

The result of searching for the first set of words (colors), conditional on their co-occurrence with the second set of words (animals) is:

Article ID	Body	Colors

1	The quick brown fox jumps over the lazy dog. At least, I though it was brown; it might have been red.	1

Note that the number of colors counted is just one, as it was earlier. The additional sentences include other color words ("brown" again, and "red"), but there is no animal in those sentences. Colors words are only identified when they co-occur with animal words.

3. Load Sink

When you click on Load Sink, a window will pop up and you will name the files into which you would like results to be stored. There are no particular rules here - data will be stored as plain tab-delimited text no matter what name you use here. It may help to use the .txt or .tab suffix here, depending on the software you will be using to open up the resulting dataset. Microsoft Excel, Apple Numbers, Filemaker and StatTransfer will be able to open the output without any difficulty.

4. Process

Click on the Process button. Sit back and wait. You will see a progress panel, and Lexicoder will let you know when the data are ready.

(The processing is of course the easiest step in a content analysis — getting and formatting the content, and designing your dictionary will take far more time. (See the Lexicoder website for citations and links to dictionaries, and work on building dictionaries.

Section 2. An Example

Let's walk through an analysis of the *ObamaExample.txt* file, using the *ObamaDict.lcd* dictionary. Both are available at lexicoder.com.

ObamaExample.txt includies President Obama's 2009 inauguration speech. Each entry is a separate paragraph from the speech; there are 36 entires in all. Note that this text has just two columns, ID and Body, and that they are separated by a tab. So long as a tab-delimited file is saved in this format, it can be loaded into Lexicoder.

To start, then, let's open the *ObamaExample.txt* dataset. With `Tab Delimited File` in the drop-down menu under `Select Source`, click on the `Load Source` button and select the file.

The dictionary, which can be opened and editing using any text processor, is as follows:

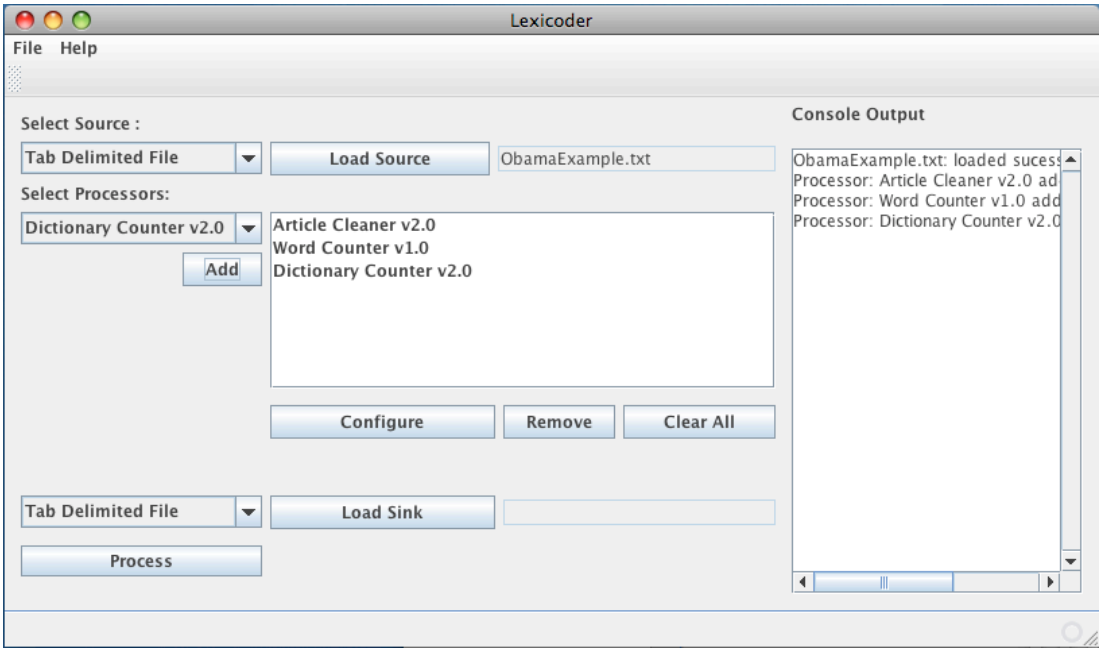
```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

```
<dictionary style="Lexicoder" name="ObamaDictionary">
<cnode name="POLITE">
<pnode name="HUMBLE"></pnode>
<pnode name="GRATEFUL"></pnode>
<pnode name="MINDFUL"></pnode>
</cnode>
<cnode name="BAD">
<pnode name="CRISIS"></pnode>
<pnode name="FEAR"></pnode>
<pnode name="CONFLICT"></pnode>
<pnode name="DISCHORD"></pnode>
<pnode name="DOUBT"></pnode>
</cnode>
<cnode name="GOOD">
<pnode name="HOPE"></pnode>
<pnode name="UNITY"></pnode>
<pnode name="FREEDOM"></pnode>
<pnode name="LIBERTY"></pnode>
</cnode>
<cnode name="CERTAIN1">
<pnode name="CERTAIN"></pnode>
</cnode>
<cnode name="CERTAIN2">
<pnode name=" CERTAIN"></pnode>
</cnode>
</dictionary>
```

Again, this dictionary is on the simple side. But it has five categories, with a varying number of words under each. (This dictionary is built just to test or demonstrate Lexicoder.)

To run the dictionary, let's first select the Article Cleaner. Select Article Cleaner from the Select Processors drop-down menu and click on the Add button. The Article Cleaner should now appear in the processors winder. Let's also get a count of the total number of words in the article. Select Word Counter from the Select Processors drop-down menu and click on the Add button.

Now select Dictionary Counter from the Select Processors drop-down menu, and click on the Add button. The dictionary counter should now show up the processors window. The Lexicoder window now looks as follows:



The source is listed after the Load Source button; the processors are listed in the processor window; and each step has been identified in the Console Output window.

The Article Cleaner does not need to be configured, but the Dictionary Counter does. So select Dictionary Counter v2.0 in the processor window (so it is highlighted) and click on the Configure button at the bottom left of the processor window. A window will pop up, and you can select the dictionary file you want to use. For this example, select *ObamaDict.lcd*.

Once you have selected the dictionary, a pop-up window will ask if you want the dictionary to be case-senstive, or not. If you select the case-sensitive option, a dictionary entry of 'OBAMA' will not match text that reads 'Obama.' For some searches, this is desirable. In most cases, however, not case-sensitive is most appropriate. In this case, select No.

Now you need to select the file into which results will be saved. With 'Tab Delimited File' selected, click on the Load Sink button. Name your file, and save it anywhere you like. Then click on Process. You will get a pop-up window when your analysis is completed.

The results can then be opened using any text editing software, or database software such as Excel. The file will include five columns: the original ID number in the first column, and then the five dictionary categories in the subsequent columns.

Results for this particular file are as follows (though the columns may not be saved in exactly this order:

ID	Word Count	POLITE	BAD	GOOD	CERTAIN1	CERTAIN2
1	3	0	0	0	0	0
2	51	3	0	0	0	0
3	82	0	0	0	0	0
4	115	0	1	0	0	0
5	45	0	2	0	0	0
6	42	0	0	0	0	0
7	21	0	2	2	0	0
8	34	0	0	0	0	0
9	80	0	0	0	0	0
10	103	0	0	1	0	0
11	20	0	0	0	0	0
12	23	0	0	0	0	0
13	19	0	0	0	0	0
14	49	0	0	0	0	0
15	100	0	1	0	0	0
16	134	0	0	0	0	0
17	58	0	0	0	0	0
18	141	0	0	0	0	0
19	113	0	1	2	0	0
20	132	0	0	0	0	0
21	79	0	0	0	0	0
22	136	0	0	0	0	0
23	122	0	0	0	0	0
24	99	0	1	0	0	0
25	80	0	0	0	0	0
26	109	1	0	1	0	0
27	94	0	0	0	0	0
28	148	0	0	0	0	0
29	9	0	0	0	0	0
30	20	0	0	0	1	0
31	63	0	0	1	0	0
32	89	0	1	0	0	0
33	42	0	0	1	0	0
34	103	0	0	2	0	0
35	5	0	0	0	0	0
36	8	0	0	0	0	0

There are 3 words from the POLITE category in the second paragraph of the speech. That paragraph is as follows: "I stand here today **humble**d by the task before us, **grateful** for the trust you have bestowed, **mindful** of the sacrifices borne by our ancestors. I thank President Bush for his service to our nation, as well as the generosity and cooperation he has shown throughout this transition."

Note that the dictonary counter captures the word "humble" even though it has a suffix in this line. This is an important feature of the dictionary counter. If you wish to avoid suffixes, you should leave a space after humble in the quotation marks; that is, you should use "humble " in the dictionary. The categories CERTAIN1 and CERTAIN1 are designed to show this as well, though using the beginning of the word. Note that CERTAIN1 includes the word "certain", which CERTAIN2 includes the word " certain" — with a space in front of it. Line 30 is "This is the source of our confidence: the knowledge that God calls on us to shape an un**certain** destiny." CERTAIN1 counts the word certain, even though it is preceded by "un"; CERTAIN2 does not count the word certain unless it is preceded by a space.

The process of identifying the many prefixes and suffixes or words, and including the appropriate forms while excluding the others, can of course be very complicated. It is important that you build your dictionary keeping in mind, then, the use (or not) of spaces, as well as the fact that the dictionary counts words in the order in which they appear in the dictionary (and then sets them aside).

Section 3. Final Comments

If you want to practice more, there are additional files available at lexicoder.com. In particular there is *USinaugurationsa1949-2009.txt*. This file includes, predictably, all inaugural addresses since 1949. Here, the cases are the addresses themselves — they are not divided by paragraph. The Lexicoder Topic Dictionary LTDv2.lcd is an early version of a topic dictionary, designed to use with Lexicoder. It is a much more complicated dictionary than the ones we have used thus far. It is by no means a finished product, but it is a useful file for practicing, and also a good template as you design your own dictionary.

Other dictionaries are available at lexicoder.com as well, including the Lexicoder Sentiment Dictionary, a dictionary designed to capture the sentiment in political texts.

Lexicoder was programmed by Mark Daku, and developed by Lori Young and Stuart Soroka, at McGill University. Comments and queries are very welcome, at lexicoder@me.com.