Lecture 8: Corpora and Data Formats, Text File Encoding & Conversion

LING 1340/2340: Data Science for Linguists

Na-Rae Han

Objectives

- Your term project
  - Plan submitted, repo created!
  - Work on your DATA!
- Corpus data: standard and popular formats
  - Encoding, line break
  - Review of common data formats
  - Conversion operations... in command line!

Your term project

- Everyone's project repo is at our GitHub org.
- First progress report is due next Friday!
  - Focus on data: sourcing, curation and cleaning
- Managing your data
  - You will be manipulating and processing your data.
  - Should you include your data set in your GitHub repo?
    - Depends on your license!

Data standards & exchange formats

	What	Notes, reference
CSV	Comma-separated values	Compatible with Excel
TSV	Tab-separated values	
HTML	Web pages	Not meant as data format
XML	For markup and text encoding	A Gentle Introduction to XML by TEI
JSON	JavaScript Object Notation (Twitter, Jupyter Notebook)	Introducing JSON JSON example (vs. XML)

These are all TEXT files!

They are all TEXT files.

- Encoding: Latin-1 (=ISO-8859-1), ASCII, UTF-8, UTF-16, CP-1252 (Windows-1252), ANSI...
- Line endings:
  - **LF** (`'\n'`: OS X & Linux), **CRLF** (`'\r\n'`: Windows)
- But underneath it all, these files are all TEXT files with **special formatting syntax** and **special characters** designated for formatting purposes.
  - In command line, you can `cat` and `less` through the files. Also: `head`, `tail`
  - You can open them up in a **text editor** (Atom, Notepad++) and edit.
  - Some editors/applications are aware of the format-specific syntax and will highlight/render accordingly.
    - Unlike, say, PDF files, style attributes are NOT part of the files themselves. (e.g., markdown file)

File formats and conversion

- "Project Gutenberg Selections" corpus, from the NLTK Corpora page (https://www.nltk.org/nltk_data/).
  - You probably already have it on your system:

```
>>> nltk.corpus.gutenberg.words()
['[', 'Emma', 'by', 'Jane', 'Austen', '1816', ']', ...]
>>> nltk.corpus.gutenberg.root
FileSystemPathPointer('D:\\Lab\\nltk_data\\corpora\\gutenberg')
```

  - Download a fresh copy, examine the included text files ('austen-emma.txt', 'shakespeare-caesar.txt', ...).
  - What **encoding scheme** do the files have? Is every file UTF-8?
  - What about **line ending**? Do you see Windows style "CRLF" line ending?
  - The file command reports 'milton-paradise.txt' as a **'data' file**, not a plain text file. Is this correct?
  - Let's bring some **consistency** to this corpus! We want:
    - UTF-8 encoding
    - Unix-style LF line ending ("\n")

Corpus content, file sizes

```
narae@T480s MINGW64 ~
$ cd Desktop/gutenberg/

narae@T480s MINGW64 ~/Desktop/gutenberg
$ ls
README                  blake-poems.txt          chesterton-brown.txt     shakespeare-caesar.txt
austen-emma.txt         bryant-stories.txt       chesterton-thursday.txt  shakespeare-hamlet.txt
austen-persuasion.txt   burgess-busterbrown.txt  edgeworth-parents.txt    shakespeare-macbeth.txt
austen-sense.txt        carroll-alice.txt        melville-moby_dick.txt   whitman-leaves.txt
bible-kjv.txt           chesterton-ball.txt      milton-paradise.txt

narae@T480s MINGW64 ~/Desktop/gutenberg
$ ls -lh
total 12M
-rw-r--r-- 1 narae 197121 9.2K Feb 13 08:54 README
-rw-r--r-- 1 narae 197121 867K Feb 13 08:54 austen-emma.txt
-rw-r--r-- 1 narae 197121 456K Feb 13 08:54 austen-persuasion.txt
-rw-r--r-- 1 narae 197121 658K Feb 13 08:54 austen-sense.txt
-rw-r--r-- 1 narae 197121 4.2M Feb 13 08:54 bible-kjv.txt
-rw-r--r-- 1 narae 197121  38K Feb 13 08:54 blake-poems.txt
-rw-r--r-- 1 narae 197121 244K Feb 13 08:54 bryant-stories.txt
-rw-r--r-- 1 narae 197121  83K Feb 13 08:54 burgess-busterbrown.txt
-rw-r--r-- 1 narae 197121 142K Feb 13 08:54 carroll-alice.txt
-rw-r--r-- 1 narae 197121 447K Feb 13 08:54 chesterton-ball.txt
-rw-r--r-- 1 narae 197121 398K Feb 13 08:54 chesterton-brown.txt
-rw-r--r-- 1 narae 197121 314K Feb 13 08:54 chesterton-thursday.txt
-rw-r--r-- 1 narae 197121 914K Feb 13 08:54 edgeworth-parents.txt
-rw-r--r-- 1 narae 197121 1.2M Feb 13 08:54 melville-moby_dick.txt
-rw-r--r-- 1 narae 197121 458K Feb 13 08:54 milton-paradise.txt
-rw-r--r-- 1 narae 197121 110K Feb 13 08:54 shakespeare-caesar.txt
-rw-r--r-- 1 narae 197121 160K Feb 13 08:54 shakespeare-hamlet.txt
-rw-r--r-- 1 narae 197121  98K Feb 13 08:54 shakespeare-macbeth.txt
-rw-r--r-- 1 narae 197121 695K Feb 13 08:54 whitman-leaves.txt
```

`ls -lh`

File sizes in human-readable format

Encoding, line-ending

MINGW64:/c/Users/narae/Desktop/gutenberg

```
narae@T480s MINGW64 ~/Desktop/gutenberg
$ file *
README:                  ASCII text
austen-emma.txt:         ASCII text
austen-persuasion.txt:   ASCII text
austen-sense.txt:        ASCII text
bible-kjv.txt:           ASCII text
blake-poems.txt:         ASCII text
bryant-stories.txt:      ASCII text, with CRLF line terminators
burgess-busterbrown.txt: ASCII text, with CRLF line terminators
carroll-alice.txt:       ASCII text
chesterton-ball.txt:     ISO-8859 text
chesterton-brown.txt:    ASCII text
chesterton-thursday.txt: ASCII text
edgeworth-parents.txt:   ASCII text, with CRLF line terminators
melville-moby_dick.txt:  ASCII text, with CRLF line terminators
milton-paradise.txt:     data
shakespeare-caesar.txt:  ISO-8859 text
shakespeare-hamlet.txt:  ASCII text
shakespeare-macbeth.txt: ASCII text
whitman-leaves.txt:      ASCII text

narae@T480s MINGW64 ~/Desktop/gutenberg
$ file -i chesterton-ball.txt
chesterton-ball.txt: text/plain; charset=iso-8859-1
```

```
file
file -i
```

Mixed!
Every file should ideally have **UTF-8** encoding with the **Unix-style LF line ending**.

But why do they come up as ASCII?
Answer: **ASCII (in 8-bit) is in fact valid UTF-8**!
(But not all UTF-8 is ASCII, if they contain non-ASCII characters.)

Text file content: lines, words, characters

```
narae@T480s MINGW64 ~/Desktop/gutenberg
$ wc *.txt
   16823   158167   887071 austen-emma.txt
    8471    83308   466292 austen-persuasion.txt
   14796   118675   673022 austen-sense.txt
   99805   821133  4332554 bible-kjv.txt
    1441     6845    38153 blake-poems.txt
    5538    45988   249439 bryant-stories.txt
    1671    15870    84663 burgess-busterbrown.txt
    3331    26443   144395 carroll-alice.txt
    9548    81598   457450 chesterton-ball.txt
    7654    71626   406629 chesterton-brown.txt
    6793    57955   320525 chesterton-thursday.txt
   18297   166070   935158 edgeworth-parents.txt
   22924   212030  1242990 melville-moby_dick.txt
   10635    79659   468220 milton-paradise.txt
    3523    20459   112310 shakespeare-caesar.txt
    4922    29605   162881 shakespeare-hamlet.txt
    3286    17741   100351 shakespeare-macbeth.txt
   17435   122070   711215 whitman-leaves.txt
  256893  2135242 11793318 total

narae@T480s MINGW64 ~/Desktop/gutenberg
$ ls -lh bible-kjv.txt
-rw-r--r-- 1 narae 197121 4.2M Feb 13 08:54 bible-kjv.txt

narae@T480s MINGW64 ~/Desktop/gutenberg
$ wc bible-kjv.txt
  99805  821133 4332554 bible-kjv.txt
```

`wc` produces line count, word count, character count

Entire corpus contains about 2.13 million words!

The Bible file is 4.2MB in size. Because it's in ASCII (= UTF-8) format, each character is 8 bit = 1 byte. That means the text file should have about 4.2 million characters. `wc` output confirms it.

Encoding conversion

MINGW64:/c/Users/narae/Desktop/gutenberg

```
narae@T480s MINGW64 ~/Desktop/gutenberg
$ which iconv
/usr/bin/iconv

narae@T480s MINGW64 ~/Desktop/gutenberg
$ iconv -f ASCII -t UTF-16 bible-kjv.txt > bible-kjv.UTF16.txt

narae@T480s MINGW64 ~/Desktop/gutenberg
$ ls -lh bible*
-rw-r--r-- 1 narae 197121 8.3M Feb 15 11:22 bible-kjv.UTF16.txt
-rw-r--r-- 1 narae 197121 4.2M Feb 13 08:54 bible-kjv.txt

narae@T480s MINGW64 ~/Desktop/gutenberg
$ file bible*
bible-kjv.UTF16.txt: Big-endian UTF-16 Unicode text
bible-kjv.txt:       ASCII text

narae@T480s MINGW64 ~/Desktop/gutenberg
$ wc bible*
   99805   821133  8665110 bible-kjv.UTF16.txt
   99805   821133  4332554 bible-kjv.txt
  199610  1642266 12997664 total
```

`iconv` to create a new UTF-16 encoded version of the bible file.

UTF-16 means double the file size!

`wc` unfortunately isn't smart enough. It just goes by byte counts when outputting character count.

Peek into file content

```
MINGW64:/c/Users/narae/Desktop/gutenberg

narae@T480s MINGW64 ~/Desktop/gutenberg
$ head -5 austen-emma.txt
[Emma by Jane Austen 1816]

VOLUME I

CHAPTER I

narae@T480s MINGW64 ~/Desktop/gutenberg
$ tail -5 austen-emma.txt
of true friends who witnessed the ceremony, were fully answered
in the perfect happiness of the union.


FINIS

narae@T480s MINGW64 ~/Desktop/gutenberg
$ for x in *.txt
> do
> echo $x
> head -3 $x
> done
austen-emma.txt
[Emma by Jane Austen 1816]

VOLUME I
austen-persuasion.txt
[Persuasion by Jane Austen 1818]


austen-sense.txt
[Sense and Sensibility by Jane Austen 1811]

CHAPTER 1
bible-kjv.txt
[The King James Bible]
```

Use `tail`, `head`

Also: `less` (**space** to page down, **q** to quit)

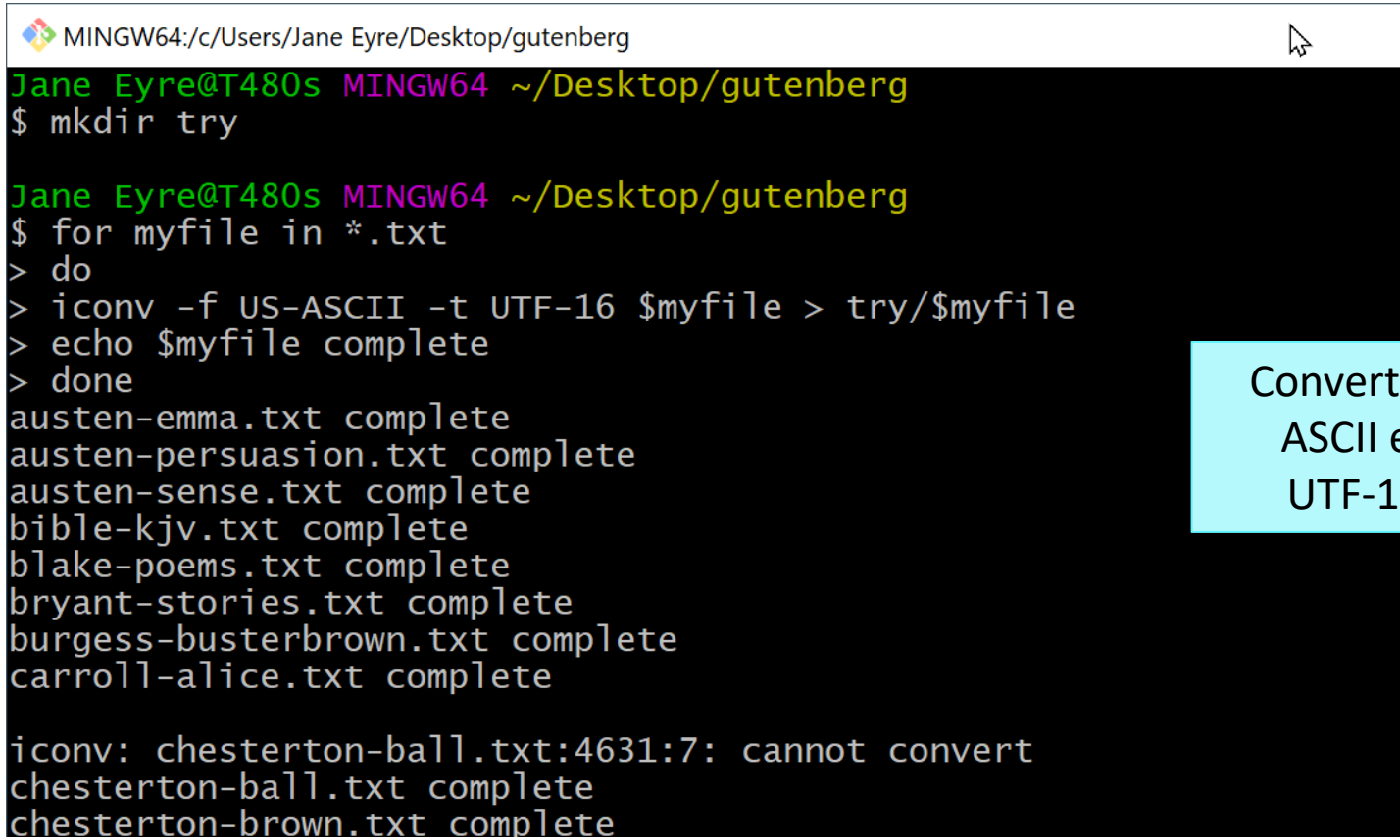
Using **for loop** to peek into first lines of every file

`x` is created as a variable, referenced later as `$x` (needs `$` prefix)

Batch processing through shell scripting

- Your command line is actually running a programming environment: bash shell.
- You can *program* in command line, even for loops!

Or: Z shell (zsh, on Macs)



Convert all files from ASCII encoding to UTF-16 encoding

Wrapping up

- No To-do out
  - Work on your project!
- Your project
  - Work on it! Focus on DATA.