Little Languages

Lecture 25:

Applications Hands-on: xargs and Scraping

VM day today!

Pull 590-material from upstream.

Upcoming Dates

- **thegrep** – Monday April 1st at 11:59pm
  - 2nd part will go up this weekend

- **2nd Midterm** - Monday April 8th

Let's Script a UNC Catalog Scraper

- Goal: Produce a list of all course codes and titles at UNC
  - http://catalog.unc.edu/courses/

- Spend a minute to jot down a plan of how you would do this by hand with a neighbor assuming you can only click, copy, and paste.
  - What are your overarching steps?

- Check-in on PollEv.com/compunc once you have a plan.

Hands-on: Downloading the Listing Page

- Open the 0-download-depts.sh bash script
- Read the instructions on the TODO and open up the manual page for wget (**:!man wget** in vim). You need to find the flag that specifies the correct "Output File"
- Your goal: download the courses web page
  1. Use wget to download the courses page http://catalog.unc.edu/courses/
  2. Store the downloaded page in ./data/depts.html
- To run your script the command is
  - (from CLI) bash 0-download-depts.sh
  - (from vim) :! bash 0-download-depts.sh
- Check in when you have the departments HTML downloaded to data.

Follow-along: Converting HTML to Markdown

- As shown Wednesday, we'll simplify our scraping problem by converting HTML to Markdown
  - This is a hack that works well for this site, but 99% of scraping is a hack
- In 1-convert-to-markdown.sh:
  - pandoc -o ./data/depts.md ./data/depts.html
- This produces a depts.md markdown file in the data directory.

Hands-on: Use **egrep** To Filter to Department Links

- Open 2-filter-dept.sh

- Department Links will look like:

- **[COMPARATIVE LITERATURE (CMPL)](/courses/cmpl/)**
- **[COMPUTER SCIENCE (COMP)](/courses/comp/)**
- **[CONTEMPORARY EUROPEAN STUDIES (EURO)](/courses/euro/)**

- Your task: find an egrep pattern that matches only these lines.
  - Warning: these links are all on the page twice. The duplicate has a *slightly* different format. You should match only one or the other.

- Check-in when you've got a pattern for matching only these lines.

Hands-on: Use sed to extract only the last group of letters in the URL

- **[COMPARATIVE LITERATURE (CMPL)](/courses/cmpl/)**
- **[COMPUTER SCIENCE (COMP)](/courses/comp/)**
- **[CONTEMPORARY EUROPEAN STUDIES (EURO)](/courses/euro/)**

- Experiment at the command-line outside of vim. Here's a starting point example that matches what's inside the square brackets:
  - `sed -E 's/.*\[(.*)\].*/\1/g' data/dept-lines.md`
- Warning: if you match against the literals /, (,), [,] they must be escaped.
- Your output should be just the lowercase codes.

Follow-along: Let's use the following pattern.

- Open 3-extract-code.sh and use:

sed -E 's/.*courses\/(.*)\/.*/\1/g' ./data/dept-lines.md >./data/dept-codes

Now, that we have all the department codes…

- We want to download *all* of the pages for each code.

- Why did we want the department code?
  1. So we can generate URLs (but we already had the URLs)
  2. So that we can save the URLs to files named <code>.html

- Knowing **wget**'s purpose is to download a URL and save it to a file, given the list of department codes we want to run a **wget** command for each one.

```
...
poli
port
pros
psyc
...
```

```
...
wget -O ./data/depts/poli.html http://catalog.unc.edu/courses/poli/
wget -O ./data/depts/port.html http://catalog.unc.edu/courses/port/
wget -O ./data/depts/pros.html http://catalog.unc.edu/courses/pros/
wget -O ./data/depts/psyc.html http://catalog.unc.edu/courses/psyc/
...
```

Introducing the **xargs** Utility Program

- The xargs utility - "build and execute command lines from from stdin"
- We'll look at one specific use of xargs today: given a line of standard input, run a command per line, and substitute the line in the command.

```
xargs -I % command %
```

- **`-I %`** specifies the "replace token" that will be substituted with each line of standard input each time the command runs.
- **command** is the start of the command being run and **%** specifies where to substitute in each line of input
- Try out an example:

```
cat ./data/dept-codes | xargs –I % echo 1:% 2:%
```

Downloading all Department Pages

```
cat ./data/dept-codes \
| xargs -I % \
        wget -O ./data/depts/%.html \
        http://catalog.unc.edu/courses/%/
```

- After this, we'll go ahead and run the 5th step of the process which is converting all the html files to markdown. This takes a couple minutes.

Hands-on: Filter Lines With Course Titles

- Lines with courses look like this:

```
**COMP 50. First-Year Seminar: Everyday Computing. 3 Credits.**
**COMP 60. First-Year Seminar: Robotics with LEGO®. 3 Credits.**
**COMP 65. First-Year Seminar: Folding, from Paper to Proteins. 3
**COMP 80. First-Year Seminar: Enabling Technology--Computers Helping
**COMP 85. First-Year Seminar: The Business of Games. 3 Credits.**
**COMP 89. First -Year Seminar: Special Topics. 3 Credits.**
**COMP 101. Fluency in Information Technology. 3 Credits.**
**COMP 110. Introduction to Programming. 3 Credits.**
```

- Add your pattern to 6-extract-courses.sh and test it out.

- Check-in when you have titles of all courses printing. If you want to see page-by-page, try piping into less:
  - `bash 6-extract-courses.sh | less`