Welcome to the Library Carpentry Etherpad!

This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try etherpad.wikimedia.org).

Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

Twitter #libcarpgmu

Checklist

- Logistics - restrooms, water fountain, emergency exits, emergency contact
- Review Code of Conduct with learners (https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html) We are all learners!
- Can you see the text (make it bigger?), can you hear us (speak louder?), can you slow down?, any other access questions?
- Schedule: Intro to Data, Shell (day 1), Git, OpenRefine (day 2) https://libcce.github.io/2018-12-13-GMU/ .(coffee breaks at 10:30 and 2:30, lunch at noon)
- Remind learners to use sticky notes to give feedback
- Get feedback at lunch and end of each day using sticky notes
- Collect attendee names
- Twitter hashtag is #libcarpgmu
- Lessons are online at https://librarycarpentry.org/lessons/
- Send out the post-workshop survey (https://www.surveymonkey.com/r/lcpostworkshopsurvey?workshop_id=2018-12-13-GMU)

Attending

Chris Erdmann / The Carpentries & CDL / chris@carpentries.org / @libcce
Sherry Lake / University of Virginia - Carpentries Instructor / shlake@virginia.edu / @shlakeuva
Peggy Griesinger (Helper) / George Mason / peggygriesinger@gmail.com / @peggygriesinger
Margaret Lam / George Mason University / mlam3@gmu.edu
Wendy Mann (Host) / George Mason University / wmann@gmu.edu / @gmudatasvcs
Debby Kermer (Helper) / George Mason / dkermer@gmu.edu / @gmudatasvcs
Jocelyn Lewis / George Mason / jlewis21@gmu.edu
Ashley Gates / George Mason University / lhowell4@gmu.edu
Liz Beckman/George Mason University/ebeckman@gmu.edu
Jen Stevens / George Mason University / stevenj@gmu.edu
Debra Hogan_George_Mason_University/dhogan1@gmu.edu
Carol Ido / American University / cido@american.edu
Bernadette Mirro / Marymount University/ bmirro@marymount.edu
Tricia Mackenzie / George Mason / tmacken3@gmu
Kevin Gunn / Catholic University of America / gunn@cua.edu

Vakil Smallen / George Washington University / smallen@gwu.edu
Debbie Bezanson / George Washington University / bezanson@gwu.edu
Kim Edwards / George Mason / kedwar13@gmu.edu
India Pasiuk/ American University/ pasiuk@american.edu
Chris Lewis, American University/clewis@american.edu
Pamela Dahlhauser / George Mason / pdahlhau@gmu.edu
Alayne Mundt/American University/mundt@american.edu
Catharine Cox / Archivist and Content Operations Supervisor, Alexander Street, a ProQuest Company / catcox70@gmail.com
Nicole Gordon / George Mason University / ngordon4@gmu.edu
Jen King, jenking@gwu.edu George Washington University
Beth Roszkowski / George Mason University / broszkow@gmu.edu
Mason Yang, Marymount Univeristy/ mason.yang@marymount.edu
Valerie Linsinbigler / George Mason University / vlinsinb@gmu.edu
Shane Hickey / American University / shickey@american.edu
Meghan Burke /Marymount University / mburke@marymount.edu
Faith Rusk / University of the District of Columbia / faith.rusk@udc.edu
Jackie Saavedra / University of the District of Columbia / jacqueline.saavedra@udc.edu
Erin Cheever / Reference and Outreach Librarian, LAC Group / erin.cheever@gmail.com
Kathleen Bell / George Mason University / kbell9@gmu.edu / @katkimbell'
Elena Landry-GMU/elandry@gmu.edu
Iris Lee Geworge Washington University
Kathy Butler / George Mason University / kbutle18@gmu.edu
Joy Suh/GMU/hsuh1@gmu.edu
Kim Hoffman / George Mason Univerity / khoffma@gmu.edu/ @kmh_nowinva
Holly Surbaugh / Georgetown University / holly.surbaugh@georgetown.edu
Jenise Overmier / Marymount University / jovermie@marymount.edu
Hannah Park / American University / hpark@american.edu
Jesse A Lambertson / jal360@georgetown.edu

Jargon Busting

Form groups
What technical terms/acronyms have you heard that you'd like to understand?
Write them down:

API (Application Program Interface) - RESTful or JSON, pull and push information, can work with them in R and Python Zenodo.org can be helpful for finding.; so is https://libraries.mit.edu/scholarly/publishing/apis-for-scholarly-resources/ ; hathitrust and orchid also have APIs

https://github.com/toddmotto/public-apis - list of some "fun" apis

https://www.programmableweb.com/apis/directory - extensive list; some may be outdated

https://www.data.gov/developers/apis - federal government Open Data API initiative

https://www.bradcoffield.com/APIs-for-librarians/ - Examples

XML

Python - programming language "easier" than C++

Java vs Javascript - essentially, Java is more of a programming language that can be used on its own to build software, where JavaScript is generally used within HTML or a web browser to provide a function. Java can also be used in a browser, but it can do a lot else.

SAML

Pandas - Used in Python for data analysis, allows you to get data structures and query/visualize data structures differently

shell - A way to give commands and access files on your computer through typing

semantic web

R and RStudio - software used to manipulate data, usually statistics, can work with API

Access

Git vs Github - Github is on the web, Git is a local command you can run on your desktop. Github uses Git, but has other functionality.

Regular Expressions

discutize

SQL (Do you pronounce this sequel or S-Q-L?) - actual database, Microsoft SQL, MY SQL - open database, query language used to get it out -- When created by IBM, it had an E, so it was sequel, so many people still say that, but as it has evolved, it is supposed to be s-q-l. For instance MySQL is officiallly My-S-Q-L. But, you'll hear both

Perl - original UNIX programming language, can have SQL commands

PHP - used to be scripting language used for websites

Database structure

pipes - a character (|) that indicates to transfer or save the output of one command to a file or another command

OpenRefine

difference between coding/programming and programmers/developers - Developers have a larger scope, programmers focus on writing the code, Coding and

- programming are interchangeable, but people may have different views when using these terms

Webhooks

JSON vs Javascript - JSON stands for JavaScript Object Notation and is more of a structured way to store metadata while Javascript is a language that you can use to interact with JSON data

blockchain

RESTful APIs

XSL

Flask - framework in Python

Git Git is a local command you can run on your desktop

Big Data Ecosystems

scripts - a file that has written code in it

programmer vs. developer (what's the difference?)

Splunk

ABAP

Numpy - Python package for doing statistics

Beautifulsoup - A package for Python that can do web scraping - take an HTML file (webpage) and parse (split) it into parts to extract the important information

ggplot2 - R Package for doing graphics

tidyr - R Package for merging datasets

dplyr - R Package for manipuating a dataset

AppliedPredictiveModeling

e107 - a Content Management System (CMS) like Wordpress and Drupal. Makes creating Webpages and Websites easier.

caret - R package for classifying and predicting

randomForest - R package for classifying and predicting

MASS - R package with lots of random useful functions

pls - R package to do a type of regression
glmne - Both glmn and glmnet are packages to do "General[ized] Linear Models" . That means regression (predicting a value from one or more other values)
Latent Direchlet Allocation
regedit - Graphical interface for Windows settings
developer
topic modelling - The computer looks for words that occur together a lot and aren't near others
OAI-PMH - Open Archives Initiative - Protocol for Metadata Harvesting
rejex (probably RegEx) which stands for Regular Expressions
CRAN libraries - Comprehensive R Archive Network web repository -- search for packages
PyPI - the Python Package Index web repository
Wizzywig - What you see is what you get (**WYSIWYG)**
CSV - Comma Separated Value record
GUI - Graphical User Interface

What is a "package"? It is a group of functions
What is a "function"? It does something, think back to algebra functions. You give a value and it changes it in a predictable way.

Get together as a class
Highlight some of the top terms you discussed in your group
Write down on the whiteboard and instructors, participants will try to explain together

Intro to Data Foundations

Learning to program is like learning another language!

Language used in the programming community
Geek?

So a little background...
Our training is a step-by-step process
All of what we will teach you is foundational
Each lesson builds on the previous lessons
People usually want to jump in right away
We start with a foundational approach (but we are always open to feedback)

Some tips as you learn...
- Borrow, borrow, borrow (Stack Overflow)
- Use what is used in your local context
- Can it help you in your professional development?
- Knowing a little bit about programming helps you talk to IT/researchers
- Can it free up your time? Something you do repeatedly?

Sometimes you really don't need to program...
For instance, what are some commonly used keyboard shortcuts?

Win | Mac

Save Ctrl + S | Command + S
Copy Ctrl + C | Command + C II
Cut Ctrl + X
Paste Ctrl + V |
Switch Applications Ctrl + tab (switch tabs) / Alt + tab (switch programs) | Command + tab (switch program) command + ~ (switch windows in the same application)
Find Text Ctrl + F
Print Ctrl + P
Undo Ctrl + Z
Redo Ctrl + Y
Take a Screenshot Fn + PrtSc | Shift + Command +4
Force Quit Ctrl + Alt + Delete | Command + Opt + Esc
Cancel Esc
Top of Page Cmd + down arrow
Bottom of Page Cmd + down arrow
Zoom In | Command + =/+
Zoom Out | Command + -/–

Select All CTRL A | Command + A
Anything else?

- Tabbing around through various open documents/tiles ALT + Tab
- Lock my screen Windows key + Flag (Cmd?) + L | Command + Ctrl +Q
- Screen shot (Mac Select screen portion = CTRL-SHIFT-4) (Windows = Snipping Tool, can be set to be delayed 1-5 secs so you can set up the screen)

Shift + arrow keys, Ctrl, End, and Home to select text) Mac - whole screen - Command-Shift-3. It captures a screenshot of your entire screen.

- https://edu.gcfglobal.org/en/techsavvy/keyboard-shortcuts/1/ & https://support.apple.com/en-us/HT201236

Often, the simplest approach is the best approach

Open formats vs proprietary...
Can you name some open/propietary formats? (Note: most carpentries tools are open except for Matlab)

Machine readable vs human readable
Some examples from our Jargon busting...
But we can try one format, Markdown...

Intro to Markdown`
HackMD is a great tool to explore Markdown:
https://hackmd.io/
Let's explore Markdown:
https://hackmd.io/Npzp9U_oROq9NRV0HWbB4A?both

How do you add a ____ in Markdown?
- Header #<space> | ##<space> | etc

- Link -- Link text in square brackets, followed by URL in parentheses
- Image <exclamation point><open square bracket><close square bracket><open parentheses>image location<close parentheses>
- Emoji https://gist.github.com/rxaviers/7360908
- Image
How do you ____ text in Markdown?
- Bold **text**
- Italicize *text*
- Blockquote > text
- >Text
Tables are hard, so use a generator:
  https://www.tablesgenerator.com/markdown_tables
  CSV to Markdown Table: https://donatstudios.com/CsvToMarkdownTable
More Markdown:
  https://github.com/OpenRefine/OpenRefine/wiki/Recipes

Tool for exporting Markdown to other formats (e.g., PDF, PPTX, HTML): https://pandoc.org/

When you start a project, how do you organize your files?
Folder structure - ..., ..., results, drafts, queries, scratch, temp, projects, data, code, originals, readme

File names - use dates as beginning of filename w/ ISO: 2018-12-13-water-from-CA.txt

File formats - CamelCase and...use underscore
dashes - kebab case,
Snake case (underline): Test_data, My_File
Best not to use spaces

Date formats (ISO 8601)
2018-10-18T16:58:00
More here:
  https://datadryad.org/pages/reusabilityBestPractices

Regular Expressions

Used in programming to match patterns (and replace)
"Finding a needle in a haystack"
Often used with text/code
You can plug it in and use it in text/code editors, scripts, OpenRefine! (Another that is often cited, brackets.io)
Also known as RegEx or regex

We will start off using:
  https://regexr.com/
  > Cheatsheet (let's first walk through this)

[A-Z] this is a range, will match characters between - and including A & Z - but capital letters only
[a-z] range will match characters between and including, a- z, lower case
\w matches any word character (alpa & underscore)
+ match one ore more of the proceding token (\w in this case)

\d matches a digit
\s matches white space
\s{2,} matches two or more spaces

\b\w{5}\b matches words with 5 characters

.* 0 or more times
.*? (not greedy - does not match across lines)

Examples of ways used:
In OpenRefine to change formatting in a column
on shell, looping to change file names
MarcEdit
to fix/standardize wonky date formats

Using grouping can then replace certain "groups"

Another helpful regular expressions cheatsheet:
http://www.cbs.dtu.dk/courses/27610/regular-expressions-cheat-sheet-v2.pdf

For our exercises, we will use The Carpentries Code of Conduct:
https://raw.githubusercontent.com/libcce/lc-lesson-materials/master/code-of-conduct.md (copy to regexr) Ctrl-A to select all, Ctrl-C to copy

Regular expression challenges using The Carpentries Code of Conduct:

- Find different spellings of organize
organi[zs]\w+
organi\w+

- Change the Markdown links to HTML links/tags
Expression: \(#{1,}) (.*) ; Replace using this in the Tools section: <h1>$2</h1>
- Change the bold Markdown headings (with HTML break tags) to HTML heading tags

- (\<br\>)() and replace with $2

- Change the Markdown headings to HTML headings

Start here as group:
- Switch the ISO dates to USA format
Expression: (\d{4})-(\d{2})-(\d{2})
Replace with: $2-$3-$1

RegEx Exercise - Possible Answers

- Find different spellings of organize

organi[zs]\w+
\organi.e
(organi).(e) and then to replace the s or z <<$1z$2>>, this way leaves the endings (-er) untouched

- Change the Markdown links to HTML links/tags?

[link text](https://www.google.com)

(\[)(.*?)(\]\s*\()(.*?)(\))
$2

- Change the bold Markdown headings (with HTML break tags) to HTML heading tags

(**)(.*?)(**)
$2

- Change the Markdown headings to HTML headings

(\#{2,4}\s*)(.*?)($)
<h3>$2</h3>

- Switch the ISO dates to USA format

(\d{4})(\-)(\d{2})(\-)(\d{2})
$3/$5/$1

* matches the astricks (*) - need to escape it with back slash "\"

More info/exercises:
https://librarycarpentry.github.io/lc-data-intro/04-regular-expressions/index.html

Shell Lessons for Libraries
Add notes, questions and comments here as we go through each lesson.

Data Files
You need to download some files to follow this lesson:
https://github.com/LibraryCarpentry/lc-shell/ralw/gh-pages/data/shell-lesson.zip

1. Download shell-lesson.zip and move the file to your Desktop.
2. Unzip/extract the file (ask your instructor if you need help with this step). You should end up with

a new folder called shell-lesson on your Desktop.

Intro Slides:
https://docs.google.com/presentation/d/1C_VgoSPp10AD-q6Dz8RBN_FWb7EX0Stdk5xh499_7ww/edit?usp=sharing

- What is the Shell?
- Windows Manual: http:/man.he.net/
- COMMAND --help
- ex: ls --help
- Navigating the Filesystem
    - Exercises
    - 1. Find out, using the manual page, how to list the files in a directory ordered by their filesize. Try it out in different directories. Can you combine it with the -l *argument* you learned before?
    - 2. Find out how you can order a list of files based on their last modification date. Try ordering files in different directories.

- Working with Files and Directories

- Shell Loops
    - Exercises
    - 1. Print the name, first line, and last line of each text file in the current directory.
- Counting and Mining with the Shell
    - Exercises
    - 1. Find out how many files and directories there are in the current directory. Try to see if you can pipe the output from ls into wc to find the answer, or something close to the answer.
    - 2. Check the manual for the wc command (either using man wc or wc --help) to see if you can find out what flag to use to print out the number of words (but not the number of lines and bytes). Try it with the .tsv files.
    - If you have time, you can also try to sort the results by piping it to sort.
    - 3. Search for all case sensitive instances of a word you choose in all four derived tsv files in **shell-lesson** directory. Print your results to the shell.
    - 4. Search for all case sensitive instances of a word you choose in the ‘America’ and ‘Africa’ tsv files in this directory. Print your results to the shell.
    - 5. Count all case sensitive instances of a word you choose in the ‘America’ and ‘Africa’ tsv files in this directory. Print your results to the shell.
    - 6. Count all case insensitive instances of that word in the ‘America’ and ‘Africa’ tsv files in this directory. Print your results to the shell.
    - 7. Search for all case insensitive instances of that word in the ‘America’ and ‘Africa’ tsv files in this directory. Print your results to a file results/new.tsv.
    - 8. Search for all case insensitive instances of that whole word in the ‘America’ and ‘Africa’ tsv files in this directory. Print your results to a file results/new2.tsv.
    - 9. Use regular expressions to find all ISSN numbers (four digits followed by hyphen followed by four digits) in 2014-01_JA.tsv and print the results to a file results/issns.tsv. Note that you might have to use the -E flag (or -P with some versions of grep, e.g. with Git Bash on Windows.).

- 10. In the earlier counting exercise in this episode, you tried counting the number of files and directories in the current directory.
    - Recall that the command ls -l | wc -l took us quite far, but the result was one too high because it included the “total” line in the line count.
    - With the knowledge of grep, can you figure out how to exclude the “total” li\ne from the ls -l output?

you can can create a bash script with .sh as the extension and run your shell command in a file, here is a test script:

```
#!/bin/bash
# My example bash script
echo "Hello World"
```

Library Carpentry Lesson: https://librarycarpentry.org/lc-shell/
Shell Lesson Reference: https://librarycarpentry.org/lc-shell/reference.html

GitHub and then Git

Learn Git via GitHub

Why would you want to learn this?
Some examples:

Democratic databases: science on GitHub Scientists are turning to a software–development site to share data and code.
https://www.nature.com/news/democratic-databases-science-on-github-1.20719

Making Code Citeable
https://guides.github.com/activities/citable-code/

Journal of Open Source Software
https://joss.theoj.org/

Academic Profile Website
https://themes.gohugo.io/theme/academic/

Our path to better science in less time using open data science tools
https://www.nature.com/articles/s41559-017-0160

FAIR Data Action Plan (Issues for Community Feedback)
https://github.com/FAIR-Data-EG/Action-Plan/issues

Code4Lib Community Statement in Support of Chris Bourg
https://github.com/code4lib/c4l18-keynote-statement

Conference Websites

https://libcce.github.io/TriangleJupyter/

Library Carpentry Lessons
https://github.com/LibraryCarpentry

Carpentries Workshop & Lesson Templates
https://github.com/carpentries/workshop-template
https://carpentries.github.io/lesson-example/setup.html

Go to GitHub
https://github.com/

Can search all public github repositories.

Sign up (create an account)
Once you've done this, copy and paste the link to your account here:
<< For example, mine is https://github.com/libcce >>
https://github.com/shlake (Sherry Lake)
https://github.com/ejcheever
https://github.com/peggygriesinger
https://github.com/mburke91
https://github.com/vakil67
https://github.com/deborahbezanson
https://github.com/jenking14
https://github.com/025point3e (Kim Edwards)
https://github.com/jyssy (Jesse Lambertson)
https://github.com/caroldid (Carol Ido)
https://github.com/olimlewis (Chris Lewis)
https://github.com/jrovermier
https://github.com/JacquelineSaavedra
https://github.com/bmirro
https://github.com/wendysmann
https://github.com/tmacken3
https://github.com/AlayneMundtSandler
https://github.com/ebeckman87
https://github.com/kbutle18
https://github.com/hlpark
https://github.com/libsh2
https://github.com/pasiukind
https://github.com/joysuh
https://github.com/pdahlhau

READMEs
What is a README? (Wikipedia link)
https://en.wikipedia.org/wiki/README
README Awesome List (see also other Awesome Lists on GitHub)
https://github.com/matiassingers/awesome-readme

LC Git Lesson README:
https://github.com/librarycarpentry/lc-git/

Readme file is the first thing one sees when they find your repository. Readme files should give enough information about your repository contents.

Can use the
HackMD tool to explore:
https://hackmd.io/

Can edit in HackMD and cut/paste into github md files.

- Create a new recipe file (e.g. lasagna.md)
> What is .md or Markdown again?
Markdown cheat sheet
https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet
> We are going to use https://hackmd.io
Share hackmd here (https://hackmd.io/7YsX_WmaSKSgCCF3QwSR9A?both)
> Copy your favorite recipe (mine is Lasagna the link above)
> Paste it to the hackmd
> Let's format create md with Hackmd
> Copy your Markdown formatted recipe
> Click Create new file button -
> Create a folder - if type this pasta/lasagna.md - the folder "pasta" is created
> Name file .md
> Paste formatted recipe
> Commit -> How to write a proper git commit message
https://gist.github.com/robertpainsi/b632364184e70900af4ab688decf6f53 (use imperative)
Note: You can also upload a .md file

<pause for questions>

Post-it exercise (forking, branching, merging w/ cats) No herding??
https://guides.github.com/introduction/flow/

(forking - creating a copy in your github account, branching - usesrs who forked a repo-make changes in their accoun this new branch, merging - now need to move the changes from the branches back to the original repository - the "ask" to put the changes on the original is called a pull request w/ cats) No herding??

Exercise
- Create an issue on another persons recipes repo
    Be concise/formatting/linking (show emoji / voting option)

- Fork a repository from someone in the room (GMU github repos to chose from are above)
  Click the "Fork" button at the top to put a copy of the repo in your github

  https://github.com/libcce/recipess

- Create a Pull Request, which forks (makes a copy in your repo) to suggest a text/format change, that can be merged by the owner

- Submit a pull request on brave soul's repository and recipe file
- Do the same for https://github.com/libcce (so I can demonstrate merging)
Note: another way to do this is to add people as collaborators in settings (can commit directly)
- Demonstrate diffs (differences between to files)
- Explain hashes or SHAs
Note: Reverting a pull request vs a commit (GitHub Desktop, GitKraken, command line)

GitHub PagesNavigate to myfirstrepo
Settings
Go to settings to turn on GitHub Pages
Lots of Jekyll themes to choose from

Example: https://github.com/libcce/TriangleJupyter
Let's fork this conference website example
Change the index, yml file

Let's move over to the command line

First set your local email (or replace with global)
git config --local user.email "email@example.com"
git config --local user.name "your name"
git config --local user.email
git config --local user.name

Have git track a folder
Call it myfirstgit - mkdir myfirst git
make this directory a git repository - git init
Check to see if it is tracking a folder - git status

Touch a file - touch file.txt
Check on the status - git status
Add the file - git add *
Check on the status - git status
Commit the file with a message - git commit -m "this is a new file"
If there are errors, go back up here in the etherpad and run the git config commands above
Check on the status - git status
View the log - git log

A visual of what you just did:
https://dev.to/sublimegeek/git-staging-area-explained-like-im-five-1anh

git show hash
Also git diff hash hash

Go back to Desktop

On github, go to recipes folder
Click on the "Clone/download" button
copy the URL from github - the copiedURL
put the github recipes repo on your desktop - git clone <copiedURL>

Go to your recipes directory - cd recipes
touch some files - touch a.txt
add - git add a.txt
commit - git commit -m "new file"
push back up to the github repo - git push

Fork https://github.com/libcce/2018-12-13-GMUl
Clone https://github.com/libcce/2018-12-13-GMU.git
cd 2018-12-13-GMU
git checkout -b <add-your-name> (e.g. add-chis-erdmann)
Note: This is called a branch
Open test.md in text editor and add your name using Markdown formatting
git add, commit
git push origin <add-your-name> (e.g. add-chis-erdmann)
Go to GitHub compare, pull request

Staying in sync
git checkout master
git remote add upstream https://github.com/libcce/2018-12-13-GMU
git fetch upstream
git rebase upstream/master (merge into your master branch)
git branch -d <add-your-name> (e.g. add-chis-erdmann)
Note: Delete your branch because it is no longer needed
git push origin --delete <add-your-name> (e.g. add-chis-erdmann)
Note: Also delete in your remote version

YouTube Video
https://www.youtube.com/watch?v=Gg4bLk8cGNo

OpenRefine for Librarians
Add notes, questions and comments here as we go through each lesson.

Downloading the data
There are a number of ways to get the data we will be using in OpenRefine.

1. Once you have started OpenRefine, use this link https://github.com/LibraryCarpentry/lc-open-refine/raw/gh-pages/data/doaj-article-sample.csv to import the data directly into OpenRefine using the **Web Addresses URLs** option.
2. You can download doaj-article-sample.csv, which is a csv file that will open in a new browser tab. Be sure to right click or control click in order to save the file (NOTE: In Safari, right click and select **download linked file**; in Chrome and Firefox, right click and select **save link as**). Make a note of the location (i.e the folder, your desktop) to which you save the file.

Intro Slides:
https://docs.google.com/presentation/d/1SmhcZ9nh1GP_AuOfDpxOjclaryIDW4pvRb4DO_9xMj8/edit?usp=sharing

OpenRefine Getting Help
If you encounter problems installing or running OpenRefine, a good source of support is the OpenRefine mailing list and forum
If you are installing OpenRefine on Windows, you may want to check the thread on Installing OpenRefine on Windows 7
There are also general and specialist tutorials about using OpenRefine available on the web, including:

- Getting started with OpenRefine by Thomas Padilla
  - http://thomaspadilla.org/dataprep/
- Cleaning Data with OpenRefine by Seth van Hooland, Ruben Verborgh and Max De Wilde
  - https://programminghistorian.org/en/lessons/cleaning-data-with-openrefine
- Blog posts on using OpenRefine from Owen Stephens
  - http://www.meanboyfriend.com/overdue_ideas/tag/openrefine/?orderby=date&order=ASC
- Identifying potential headings for Authority work using III Sierra, MS Excel and OpenRefine
  - https://epublications.marquette.edu/lib_fac/81/
- Free your metadata website
  - http://freeyourmetadata.org/
- Data Munging Tools in Preparation for RDF: Catmandu and LODRefine by Christina Harlow
  - http://journal.code4lib.org/articles/11013
- OpenRefine News (monthly round up of new blog posts, tutorials and other information)
  - http://openrefine.org/category/edge-case.html

Where to find the LC OpenRefine Lesson:
https://librarycarpentry.org/lc-open-refine/

How to get MARC (.mrk) files into OpenRefine (Owen Stephens)
http://www.meanboyfriend.com/overdue_ideas/2015/07/worked-example-fixing-marc-data-4/
How to get MARC (.mrk) files out of OpenRefine (Owen Stephens)
http://www.meanboyfriend.com/overdue_ideas/2015/07/worked-example-fixing-marc-data-5/
Importing XML data into and out of OpenRefine (Gretchen Gueguen) + lots of other stuff good for cleaning library XML data
https://github.com/dpla/Metadata-Analysis-Workshop/blob/master/OpenRefine.md

Open refine: Allocating more memory: https://github.com/OpenRefine/OpenRefine/wiki/FAQ%3A-Allocate-More-Memory