Working with Bibliographic Data I

Dr Christopher Ohge
15.03.2023

Plan for today

1. What is ‘software carpentry’, and why should you care about *library carpentry*?
2. Principles of Working with (Unstructured) Text Data
   a. Command Line Interface
   b. Regular Expressions
   c. Grep
3. Git for version control
4. Working with data
   a. Data Visualisation Exercise: Rawgraphs
   b. OpenRefine

Download sample dataset from the Universal Short Title Catalog (USTC) project.

Software Carpentry

Software carpentry teaches data fundamentals using an ***applied approach, avoiding the theoretical and general in favor of the practical and specific****, by showing learners how to solve specific problems with specific tools and providing hands-on practice.*

Library carpentry ***focuses on building software and data skills within library and information-related communities****. The goal is to empower people in these roles to use software and data in their own work and to become advocates for and train others in efficient, effective, and reproducible data and software practices.*

Data and data structure as source study

Tim Hitchcock: ‘We can now engage with that new resource in new ways, but to do so we need scholars who can work with old sources in their new digital guide, and who can recognise that digital makes them different’.

Digital format of records can be understood as a new kind of historical source study.

Born-digital versus digitised data:

- Born-digital created from (and for) digital applications (rarely intended for humanistic study, therefore messy)
- Digitised data is converted from printed sources (good for humanistic study but often unclear what methods were used in the conversion)

Plain text

‘Plain text’ refers to platform-independent text data that can be parsed and processed by the computer.

File extension	Format	Plain text?
.doc, .docx	Word processing	no
.xsl, .xslx	spreadsheets	no
.txt	Any text	yes
.xml	extensible (structured) markup	yes
.csv, .tsv	Comma-separated or tab-separated values for spreadsheets	yes

(Jonathan Blaney, Sarah Milligan, Marty Steer, and Jane Winters, *Doing Digital History*, Manchester University Press, 2021.)

Command Line Interface (CLI)

CLI primarily allows you to navigate through your computer and to run scripts for a variety of purposes.

Most people interact with personal computers is called a **graphical user interface** (GUI), but CLI allows you to use commands to do specific tasks (rather than through menu-based items).

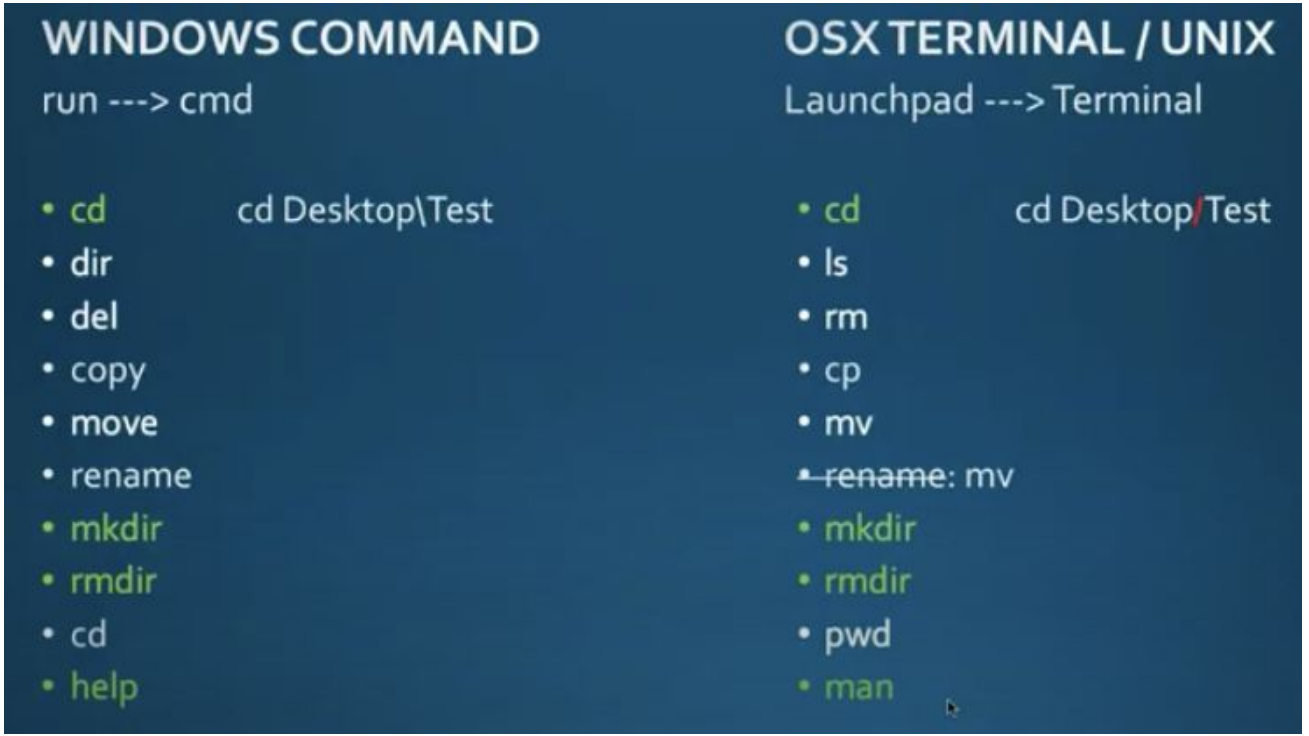
The program for performing CLI work is usually a **unix shell** or **terminal**. A unix shell is both a **command-line interface** (CLI) to read commands, and a scripting language that allows repetitive tasks to be done automatically and fast.

The most popular Unix shell is Bash (which is standard on most Mac- and Linux-based systems).

Bash shell prompt

When you first open the shell, you will see a **prompt**, indicating that the shell is waiting for input. Usually you will see the prompt begin with a dollar sign ($).

You then add the first command.



Version control with Git

Git is a version control software.

Most people use the cloud-based tool GitHub to host data, collaborate with others, and manage and share their work.

Git allows users to access repository data from the cloud (via a **pull**), make changes, **add** and **commit** those changes, and **push** the changed data back to the cloud so that other collaborators can see the changes.

Typical git commands on the CLI

> git **clone** <url of repo> [location on your computer]

```
  git clone https://github.com/SASDigitalHumanitiesTraining/lrbs-bibsoc-series.git
lrbs-bibsoc-2023
```

> git **pull**

> git **add [filename]**

```
   git add printing_dataset_lrbs.csv
```

> git **commit** -m “summary of your changes”

> git **push**

Git project management

- Work from the **root**
  - Add and commit your own work, and push it when it is ready
- Create a **branch**
  - Create a copy of the main repo that is specific to you. Add and commit your own work to the branch. When finished, or ready for testing, merge your branch with the root.
- **Fork** a repository
  - Create a full copy of a repo (and, if applicable, all its branches) but without writing (pushing) privileges. Many people do this to use or modify software packages. In some cases you can fork to make changes to a project and then feed back to the repo owner.

Learn more about Git

Software Carpentry Git Tutorial

Regular expressions

- Sequence of characters that define a search pattern
- Powerful way of searching and manipulating data

(?<=\.) {2,}(?=[A-Z])

I watch three climb before it's my turn. It's a tough one. The guy before me tries twice. He falls twice. After the last one, he comes down. He's finished for the day. It's my turn. My buddy says "good luck!" to me. I noticed a bit of a problem. There's an outcrop on this one. It's about halfway up the wall. It's not a

https://en.wikipedia.org/wiki/Regular_expression

Regular expressions

[ChBl]eating matches *Cheating* and *Bleating*

Regular expressions

[0123456789] matches any number

Regular expressions

[0-9] matches any number

Regular expressions

[0-9l] matches any number or *l*

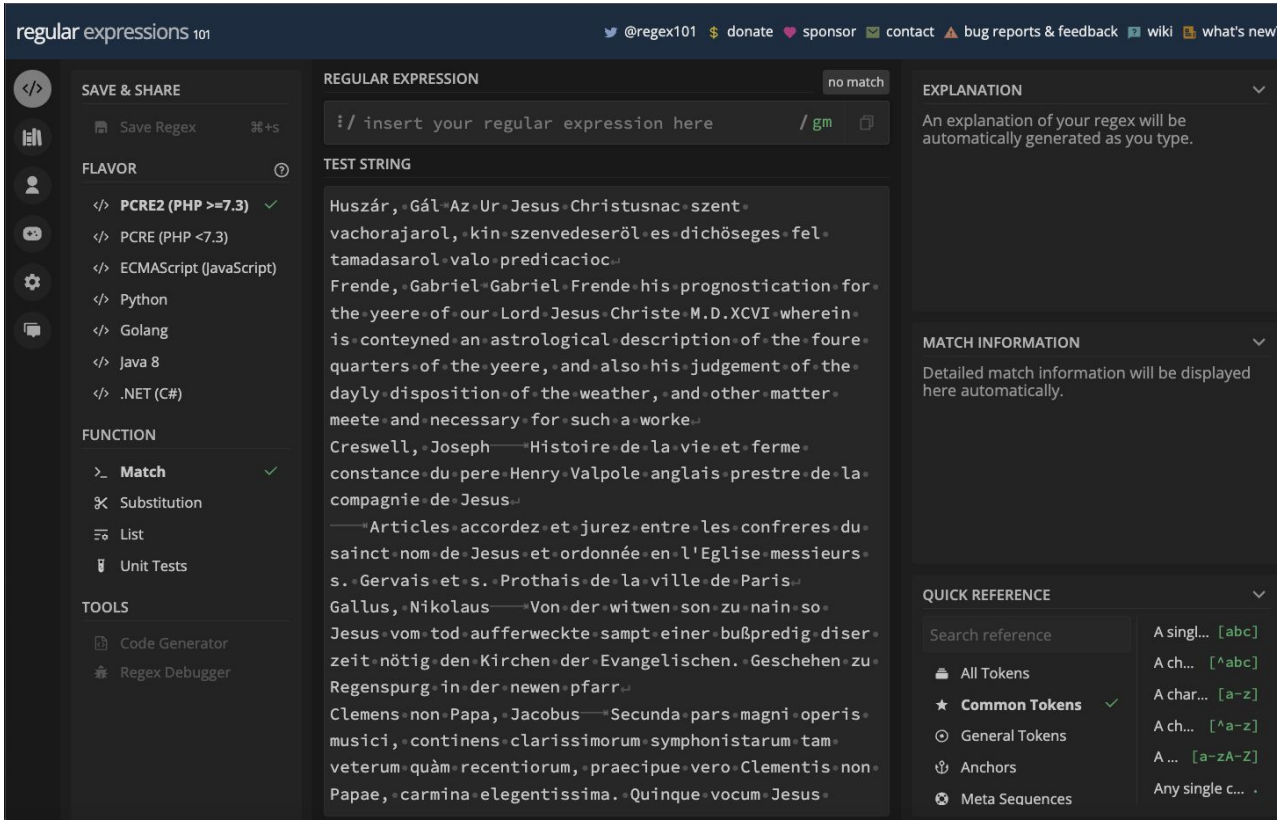
So [0-9l]+ matches *1717* or *l7l7*

Regular expressions

What does [a-z]+ match?

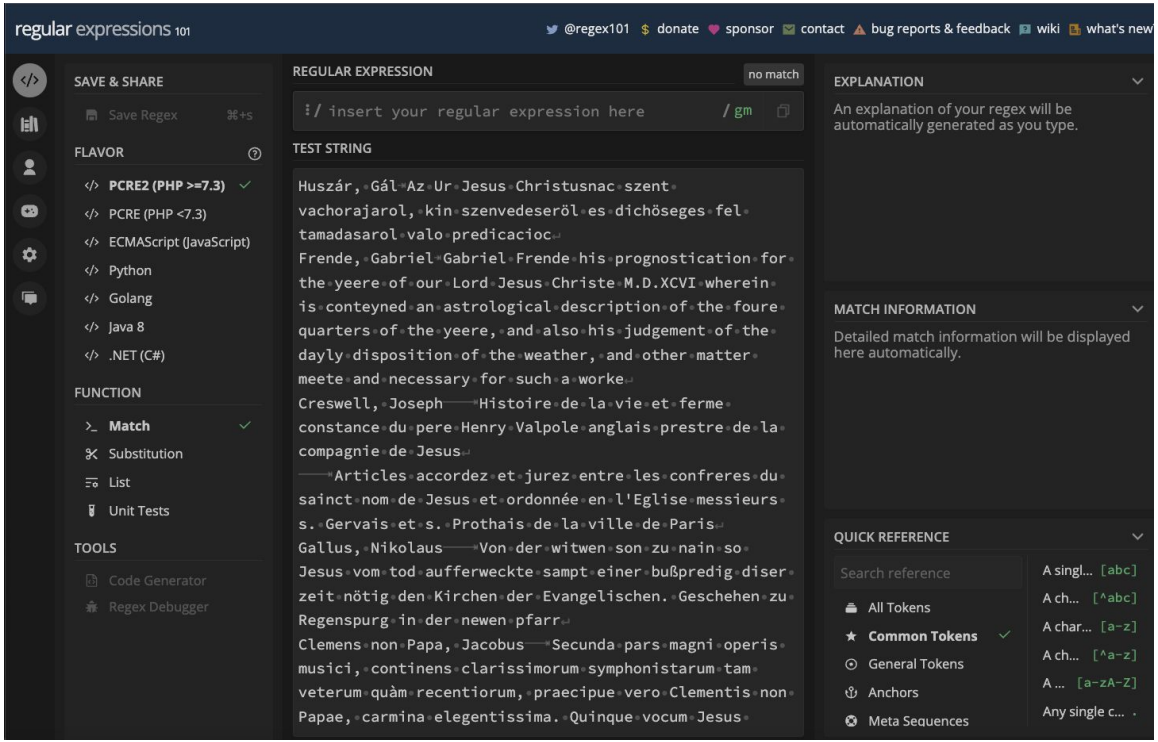
Demo

https://regex101.com



Examples

https://regex101.com

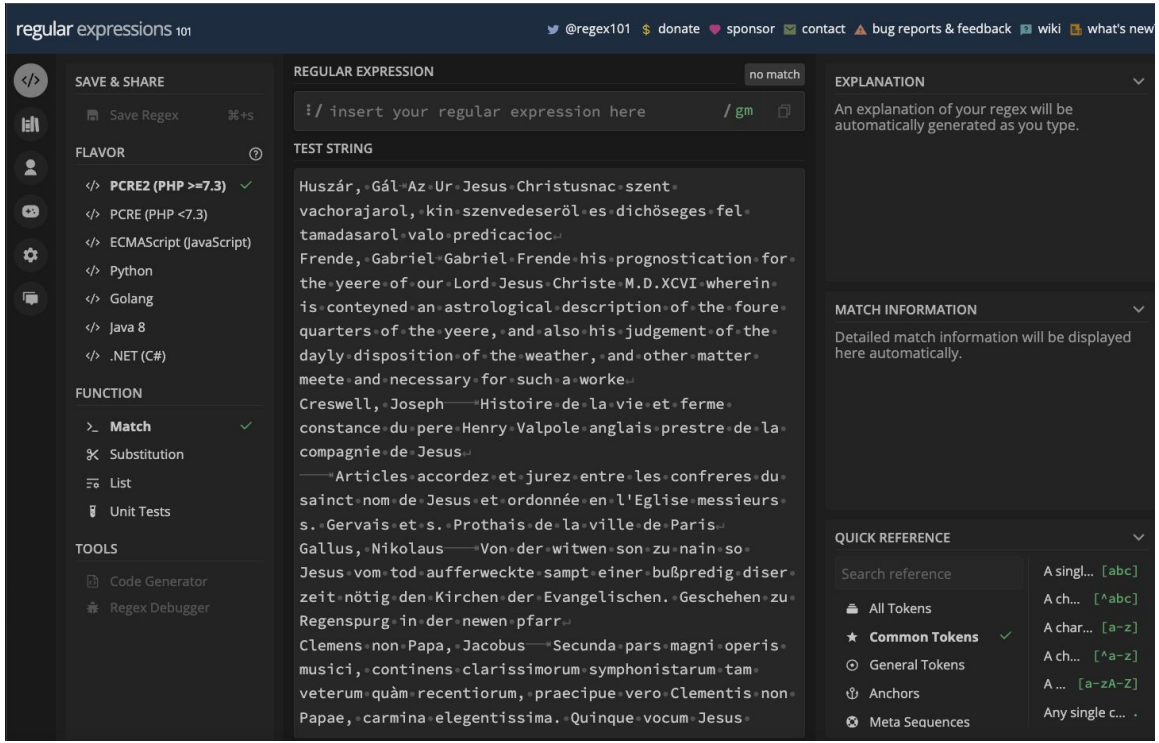


Regular expressions to:

- Match all sentences
- Match all words
- Match all words at end of sentence
- Match all words start with ‘s’
- Match all words ending with ‘s’
- Match all Proper Nouns

Solutions

https://regex101.com



Regular expressions to:

- Match all sentences
- Match all words
- Match all words at end of a line
- Match all words start with ‘s’
- Match all words ending with ‘s’
- Match all Proper Nouns

.*?\.

\b.*?\b

\w*\.

\bs.*?\b

[A-Za-z]+s

[A-Z][a-z]*?\b

Notes

- ar – search for string you want to find
- a.l - wildcards or character classes, the dot
- a.l\. – to find a fullstop (escape with backslash)
- .{5} - quantifiers – number of repetitions, not overlapping
- l{2}
- .* or .+ - all or one or more
- m[aeiou] – any one character, say vowels, or digits [0-9]
- m[0-9]+
- [A-Z][aeiuo]
- \w – any alphanumeric (or \W non alpha)
- \d – any digit (\D non digit)
- \s – any whitespace (space, tab, newline) (or \S non whitespace)
- [a-z]+?s\W
- t.*s - all words begin with T and end with S, then introduce greedy ?
- \b[Tt]\S+?s\b– word break, start with T, end with S, word break

grep

- **grep** stands for global regular expressions. This is a powerful command to search through multiple files. It literally processes a text line by line and prints the results that match the regular expression.
- Syntax:

  ```
  $ grep [OPTIONS] PATTERN [FILE...]
  ```
- On the command line, use **cd** to navigate to the USTC data you just downloaded.
- Try this command:

  ```
  $ grep --color “Holy Roman Empire” printing_dataset_lrbs.csv
  ```
- To count the results, use a count option:

  ```
  $ grep -i --count “Holy Roman Empire” printing_dataset_lrbs.csv

  $ grep -i --count “Jesus” printing_dataset_lrbs.csv
  ```
- Pipe the results into a separate file:

  ```
  grep -i --color "Jesus" printing_dataset_lrbs.csv > jesus-results.csv
  ```

grep and sed

- The next major use of regex on the command line is **sed** (literally "stream editor"), a simple search and replace function that can be used for multiple files. With **sed** we can regularise the word over multiple files using this syntax:

```
$ sed -e "s/STRING_TO_REPLACE/STRING_TO_REPLACE_IT/g" file.txt
```

- Suppose you want to replace “Köln” with “Cologne”

```
$ sed -e “s/Köln/Cologne/g” printing_dataset_lrbs.csv
```

RawGraphs

RawGraphs is free, open source, web-based data visualization software that can create and export more than 30 types of data visualisations.

Go to the SAS PORT page to access the RawGraphs exercise.

OpenRefine

OpenRefine is a free, open source tool for working with data: cleaning it; transforming it from one format into another; and extending it with web services and external data.

Typical uses:

1. Remove duplicate records
2. Separate multiple values contained in the same field
3. Analyse the distribution of values throughout a data set
4. Group together different representations of the same categories

Learn more about OpenRefine

[Programming Historian tutorial]

Programming Historian

ABOUT CONTRIBUTE LESSONS EVENTS SUPPORT US BLOG EN ES FR PT

Cleaning Data with OpenRefine

Seth van Hooland, Ruben Verborgh, and Max De Wilde

This tutorial focuses on how scholars can diagnose and act upon the accuracy of data.

Peer-reviewed CC-BY 4.0 Support PH