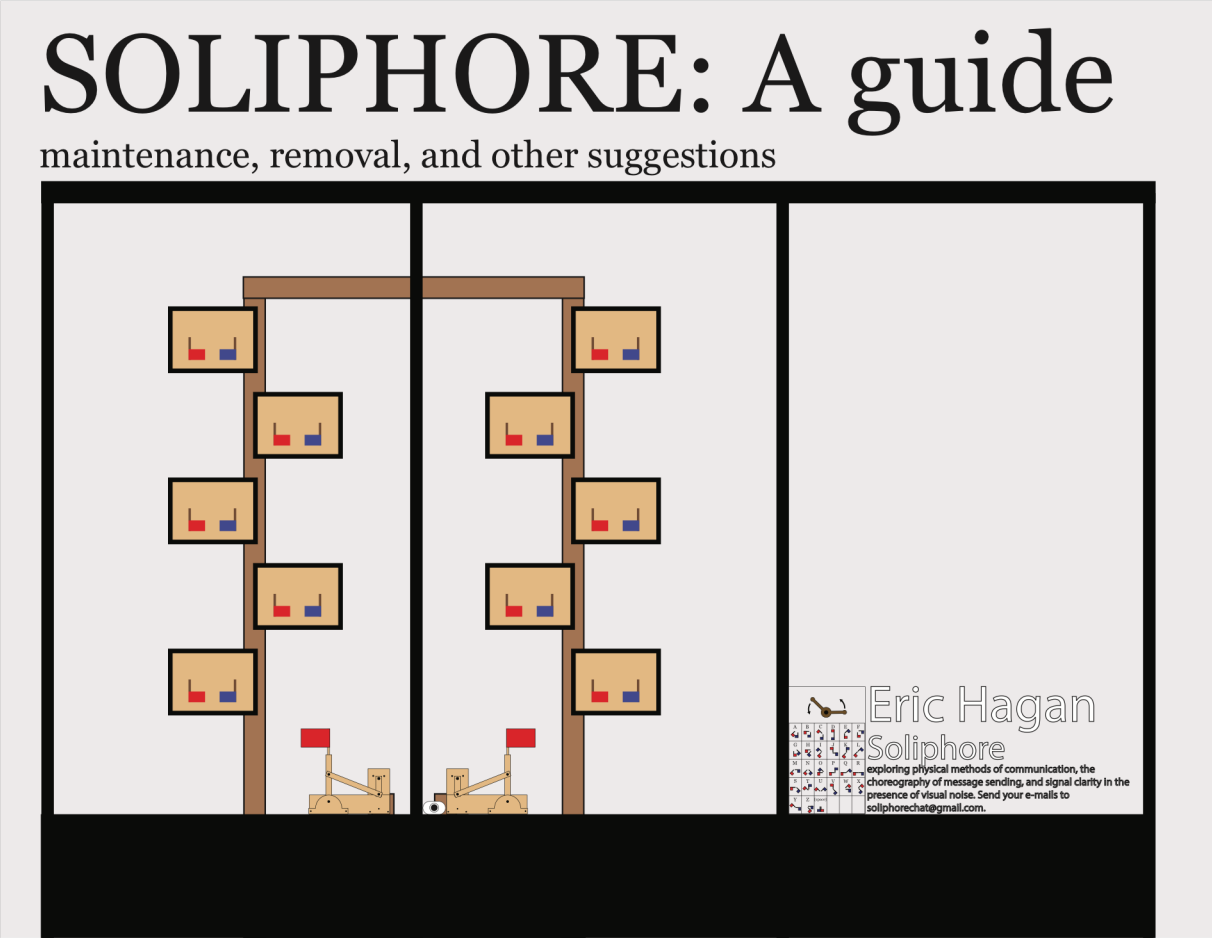
Text and Context

Working with words in python

By Eric Hagan

Introductions

Previous projects



Ideas?

What sort of text would you like to work with?

Law documents, Government documents, Books, Articles, Poetry, Movie reviews, etc

Make new documents? Alter old ones? Use documents to create social media bots? Display text on a small /LED screen?

Two separate repli.it

https://repl.it/@hagane/Word-tests-text

https://repl.it/@hagane/Word-tests-PDF

We will rewrite these examples together as well

General advice

1. Check your formats! Different formats require different input methods (TXT, PDF, docx, etc)

2. Text from scans (legal documents, or others which don't start as digital files for whatever reason) can be a bit odd and be misspelled or missing characters. VISUALLY CHECK YOUR TEXT

3. Build specific tools for your purpose, rather than trying to universalize them (at first)

Project Gutenberg – Downloading books

Alice Adventure's in Wonderland

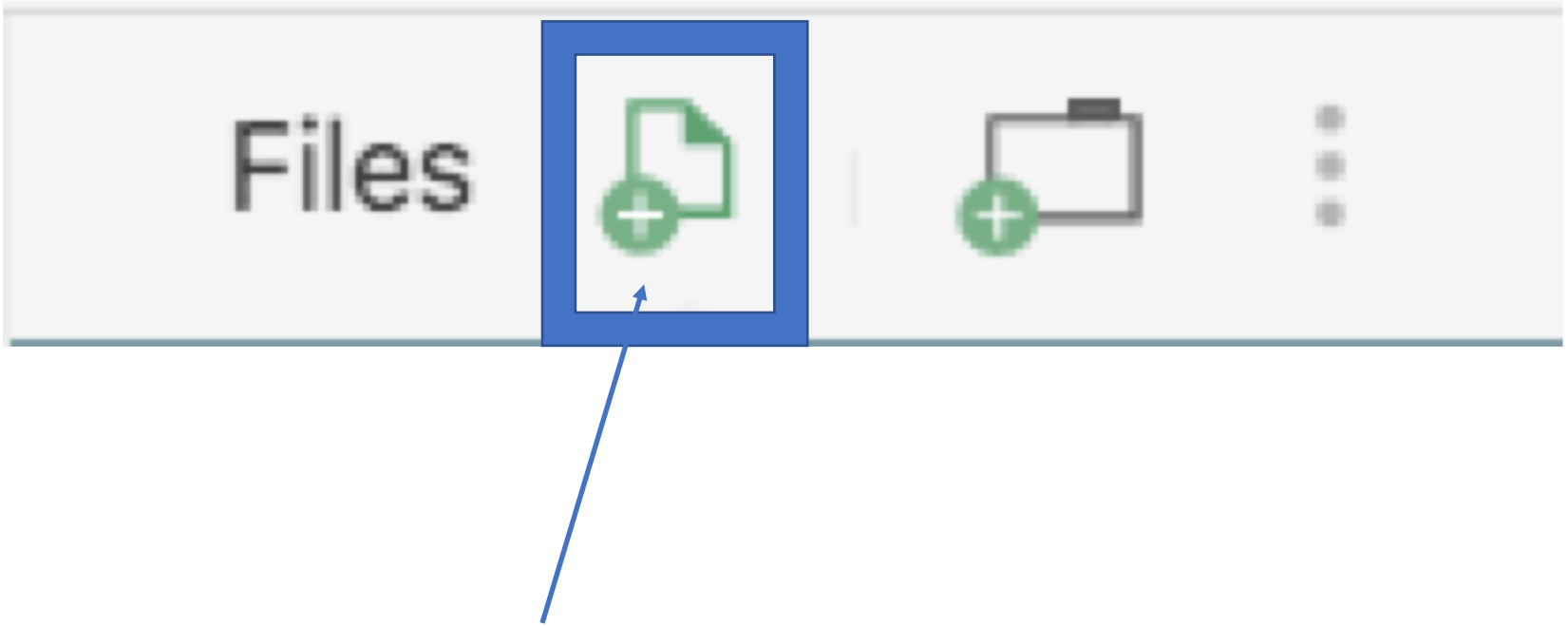
https://www.gutenberg.org/ebooks/11

We are going to download both the .txt file, and the .PDF file

These files should appear in the default downloads location (probably a downloads folder)

Rename these files 'alice.txt' and 'alice.pdf' for easier use later

Using Repl.it to load files



Upper left hand corner, add the file to the Repl.it you are working on

Import plain text files into python

```
With open (‘alice.txt’) as f:
        words = f.readlines()
        print words
        print(len(words))
```

Create a list with each line occupying a spot,

print out all of the lines to the console

print out the number of lines

Lists in Python

Lists are: Ordered (0, 1, 2, etc)

Changeable (we can alter the contents of the list

This doesn't tell us anything about the type of stuff we have stored in the list, but since we are working with text our list is made up of a number of different **Strings**

Strings are

String of Characters: Individual letters or symbols (including the digits of numbers but not the actual values) grouped together

Python does not have a “character” data type, just strings with a size of one

Single characters => String of Characters == String

Keyboard interrupt

When we run a python program, we expect that the program will do everything we ask, and then it will end. However, sometimes if our program takes a very long time we might want to end it early so that we can make changes to it before letting the whole thing run.

In this case, we can interrupt the python program using the keys. Keyboard Interrupt (if you are clicked in to the repl.it window) is

Ctrl + c

Try - except

Try – except will allow us to set up error collection…which is important here since this will allow us to stop our program even as it goes to read each line

```
try:
        #write python program here

except KeyboardInterrupt:
        #What will happen when we use the keyboard to cancel
        f.close # close our file
        print('keyboard close')     #tell us what stopped the program running
```

Displaying one line at a time

#this will give us time based elements

import time

#this will let us change things about replit…not necessary if you aren't #using replit

import replit

Printing out each line of text

```
for line in words:
    print(line)
    time.sleep(1)
    replit.clear()
```

Print the line, wait one second (can put in fractions) then clear the print window to keep the display console clear

Replacing words

Since we are loading all of our lines into a list, we can look through the list and change elements in the list

words = [w.replace(‘Alice’, ‘Your name’) for w in words]

Now the story is about you! But only in places where Alice is written with a capital A.

Note this also doesn’t change any pronouns

Replacing words

words = [w.replace(‘Alice’, ‘Your name’).replace(‘ALICE’, ‘YOUR NAME’).replace(‘hers’, ‘his’) for w in words]

#We can keep going adding elements here, we could create a dictionary (dict) of key value pairs to replace and run through each one, but you get the idea

Changing text formatting

Change the capitalization of the text

words.upper, words.lower,

Asking for user input and adding it to text

question = "Who is this story about? ” #this will be our user question

answer = input(question) #request information from the user

words = [w.replace(‘Alice’, answer).replace(‘ALICE’, answer.upper) for w in words]

NEW FILE: Working with PDFS

We are going to use a python library named PyPDF2

import PyPDF2

This will take a moment to add to your repli.it

Opening the PDF into an object

pdfFileObj = open('alice.pdf', 'rb’)

rb = Read binary, the way we are looking at the file

Reading the file

```
pdfReader = PyPDF2.PdfFileReader(pdfFileObj)
```

The pdfReader is an object in the library that will let us use the PDF, grabbing individual pages, text, counting the document length, etc

Page count

print(pdfReader.numPages)

Print out the page count for the PDF

Working with the text

```
page = pdfReader.getPage(50)
text = page.extractText()
```

This will get a single page from the PDF, and then allow us to collect all of the text from that page.

If we wanted all pages, we could use a for loop (this will print out each page, once a second)

```
for page_number in range(totalPages):
        page = pdfReader.getPage(page_number)
        page_content = page.extractText()
        print(page_content.encode('UTF_8’))
        time.sleep(1)
```

PDF text is formatted

This means there are plenty of hidden characters which determine how the text is drawn to the screen

/n is a new line character, and we will see these if we use

print(text.encode('UTF_8’))

UTF_8 is a standard way of encoding text (4 digit code, which includes all symbols) This will show us all of the excess formatting characters we don’t normally see

Stripping the formatting

In order to remove the format, we can use a Regular Expression!

#this will add the regular expression library to our python code

import re

Our regular expression

```
cleanText = re.sub("[^a-zA-Z0-9 ]+","
",text)
print(cleanText)
```

Sub means replace, the first grouping is what we wish to keep, (all lowercase and capital letters, and also numbers), what to replace any other symbols we find with (in this case, nothing) and which element/list we are applying this regular expression to

Then we can just print out our cleanText

print(cleanText)

This will produce our pdf text, minus any hidden or formatting characters…though it will also remove all punctuation!

Textwrap

Textwrap is another library, and will let us wrap any body of text to be a set number of characters long. Might be useful if we wanted to create a social media bot that produced single lines of Alice and posted them to the internet.

wrap() will also let us create start and end characters, incase we need these for some reason.

I used <>, {}, [], and !? to determine which flag messages got sent where

Questions? Project ideas? Next steps

Note, this PDF is now available at thiserichagan.com/textpython.pdf

So you could go very meta, change the name on the first page, and give this presentation yourself!