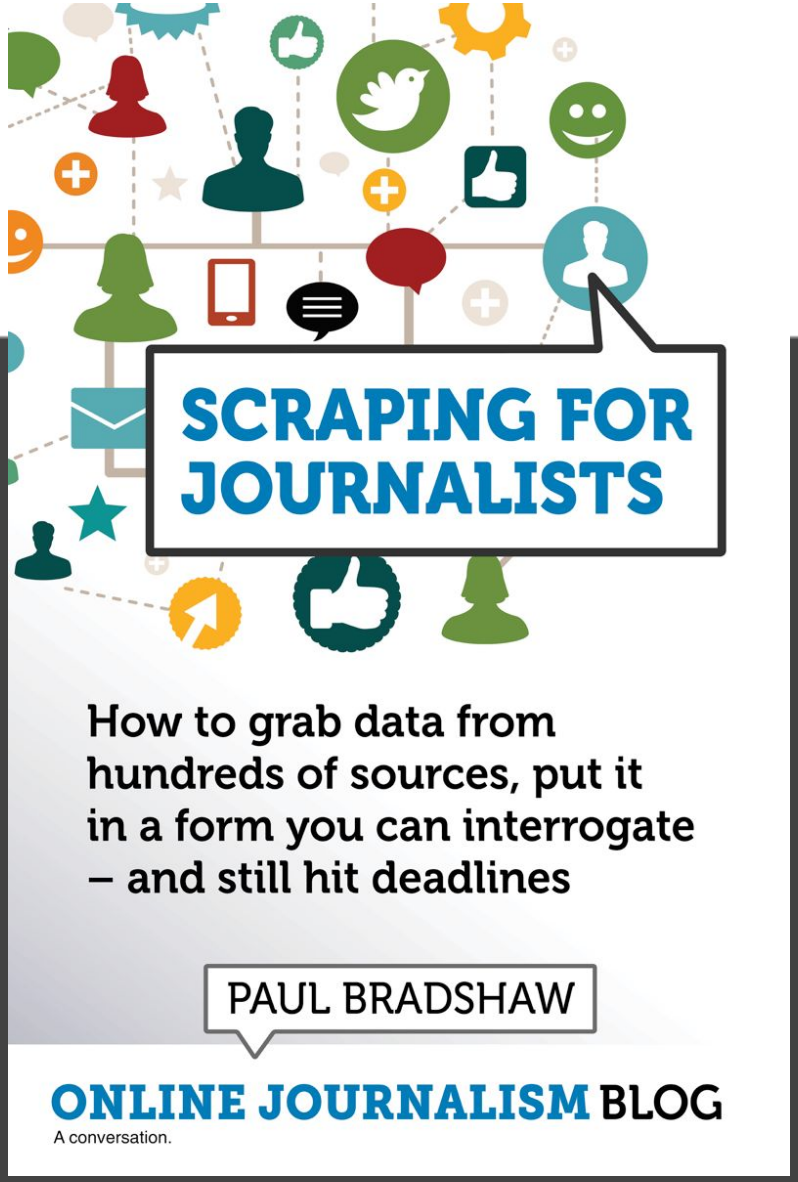
Lists redux: storing the data



Paul Bradshaw

Leanpub.com/scrapingforjournalists

What we'll cover

- Using **lists** to extract the data you want
- Using **pandas** data frames to store it

The story so far

- We’ve extracted 103 <th> tags
- And 100 telephone numbers
- How do we store them?

The data needs to line up

We need 100 of each, so need to exclude the first 3 <th> matches (headings). Some options:

- Change our selector to be more specific
- Only store the 4th item onwards
- Only store the last 100 items

Tip: slicing a list

Slicing a list involves specifying a start and end index like so:

```
first10 = mylist[0:10]
```

If you don’t specify a start or end point, it will default to the start or end of the list:

```
first10 = mylist[:10]
from10on = mylist[9:]
```

Don’t forget negative indices too:

```
last10 = mylist[-10:]
```

```
servicenames = servicenames[3:]
servicenames = servicenames[-100:]
```

```
#grab the contents of every <th> tag

servicenames = root.cssselect('th')

print(len(servicenames)

#limit to the last 100

servicenames = servicenames[-100:]

print(len(servicenames)

#grab the contents of each <div class="fcdetailsleft"> tag

tels = root.cssselect('div.fcdetailsleft')

#count how many matches are in that list

print(len(tels)
```

Introducing pandas!

We need to store the data

The pandas library has functions to create a data frame (table) and add to it

- The **`pandas.DataFrame()`** function creates a data frame with specified columns
- The **`.append()`** function adds extra rows to a data frame - those rows need to be stored in a dictionary

```
df = pandas.DataFrame(
columns=["service"] )



df = df.append(
  { "service" : servicename },
  ignore_index=True)
```

Introducing dictionaries!

The dictionary variable

- Uses **curly brackets**
- Contains a list of **pairs**, separated by a colon
- **`{"name" : "Paul", "age" : 21}`**
- The first part of the pair is the **key**
- The second part is the **value**
- ...So they're called **key-value pairs**
- The key is always a string; the value can be a string, number, True/False, or anything else
- Multiple dictionaries can be used to create rows in a table, e.g. row 2 might be:
  `{"name" : "Xian", "age" : 31}`

Creating a dictionary

```
#create a dictionary
#with 2 key-value pairs
mydictionary = {“name” : “Paul”,
“age” : 21}
```

Expanding a dictionary

- `#create an empty dictionary`
  `mydictionary = {}`
- `#create a key and store a value`
  `mydictionary['name'] = "Paul"`
  `mydictionary['age'] = 21`
- `#print the dictionary`
  `print(mydictionary)`

```
#Create a dataframe to store the data we are about to scrape
#It has two column called 'service' and 'details'
#We call this dataframe 'df'
df = pandas.DataFrame(columns=["service"])


#loop through a range of indices, generated using the range function
for i in servicenames:
  #extract the text
  servicename = i.text_content()
  #then add to the df
  df = df.append({
    "service" : servicename
  }, ignore_index=True)
  print(df)
```

Curly brackets = the dictionary

```
#Create a dataframe to store the data we are about to scrape

#It has two column called 'service' and 'details'

#We call this dataframe 'df'

df = pandas.DataFrame(columns=["service"])
```

Data frame created here (empty)

```
#loop through a range of indices, generated using the range function

for i in servicenames:

  #extract the text

  servicename = i.text_content()

  #then add to the df

  df = df.append({

    "service" : servicename

  }, ignore_index=True)

  print(df)
```

Extra row added to data frame here, each time the loop runs

We need to export the data

The pandas library has functions to import and export data to and from CSV

- The `.tocsv()` function creates a CSV with a specified name, using the data frame it's attached to

```
mydataframe.tocsv("mycsv.csv")
```

- The CSV file will be in the Files area in the left hand navigation in Colab

```
df.tocsv("scrapeddata.csv")
```

We need to loop through two lists at once

Our next problem is that we need to loop through both lists in order to store both pieces of data in a ‘row’.
To solve this problem we loop through a range of **numbers**, rather than the lists themselves, and use those as **indices** to access items from each list

- ```
  for i in range(0,100):
    list1[i].text_content()
    list2[i].text_content()
  ```
  ```

```
df = pandas.DataFrame(columns=["service","details"])


#Because we need to loop through two lists of the same length, we can
instead loop through a range of indices

for i in range(0,100):

  #extract the text from that index in each list

  servicename = servicenames[i].text_content()

  tel = tels[i].text_content()

  #then add to the df

  df = df.append({

    "service" : servicename,

    "details" : tel

  }, ignore_index=True)
```

The nth item in each list

Now we can store both items in one dictionary

```
df = pandas.DataFrame(columns=["service","details"])


#Because we need to loop through two lists of the same length, we can
instead loop through a range of indices

for i in range(0,100):

  #extract the text from that index in each list

  servicename = servicenames[i].text_content()

  tel = tels[i].text_content()

  #create an empty dictionary variable

  ourdata = {}

  #store the two pieces of data in that

  ourdata['servicename'] = servicename

  ourdata['tel'] = tel

  #then add to the df

  df = df.append(ourdata, ignore_index=True)
```

The nth item in each list

Now we can store both items in one dictionary

```
df = pandas.DataFrame(
columns=["service","details"] )

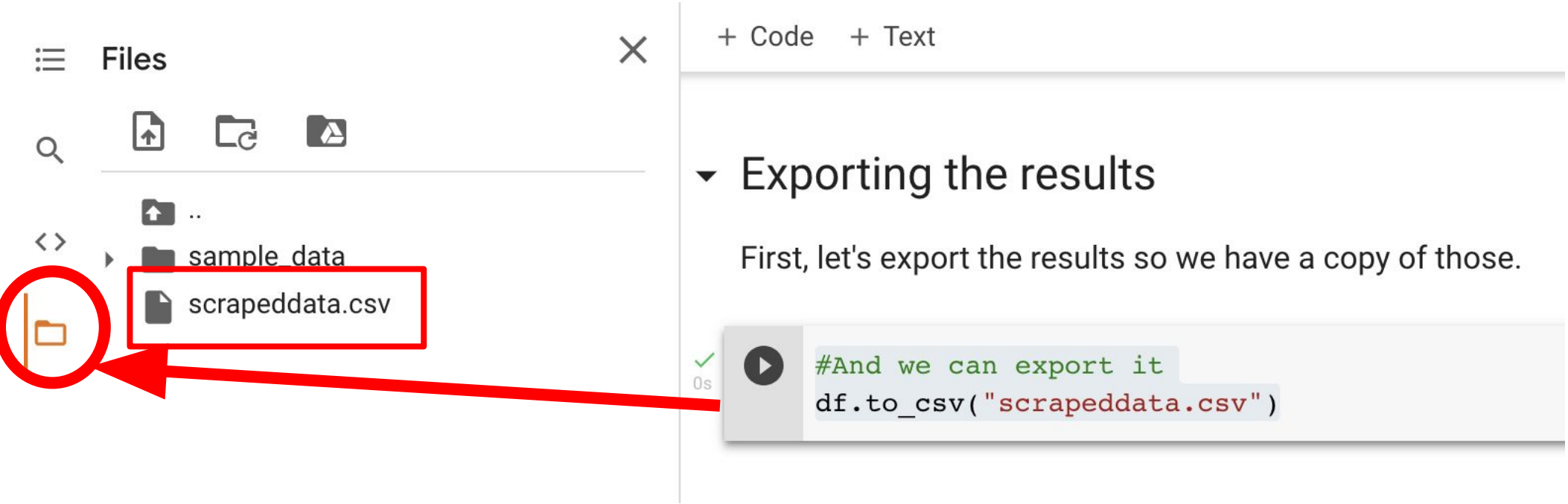


df = df.append(
  { "service" : servicename,
    "details" : tel },
  ignore_index=True)
```

Get it out!

```
#export it

df.to_csv("scrapeddata.csv")
```



Try it now:

- In your notebook scrape the page and extract the contents of:
  - <th> tags
  - tags
- Loop through a range and print, for the items at those indices, their `.text_content()`
- Store in a dictionary
- Append the dictionary to a pandas data frame
- Export it

Recap

- Use pandas to create a data frame to store data

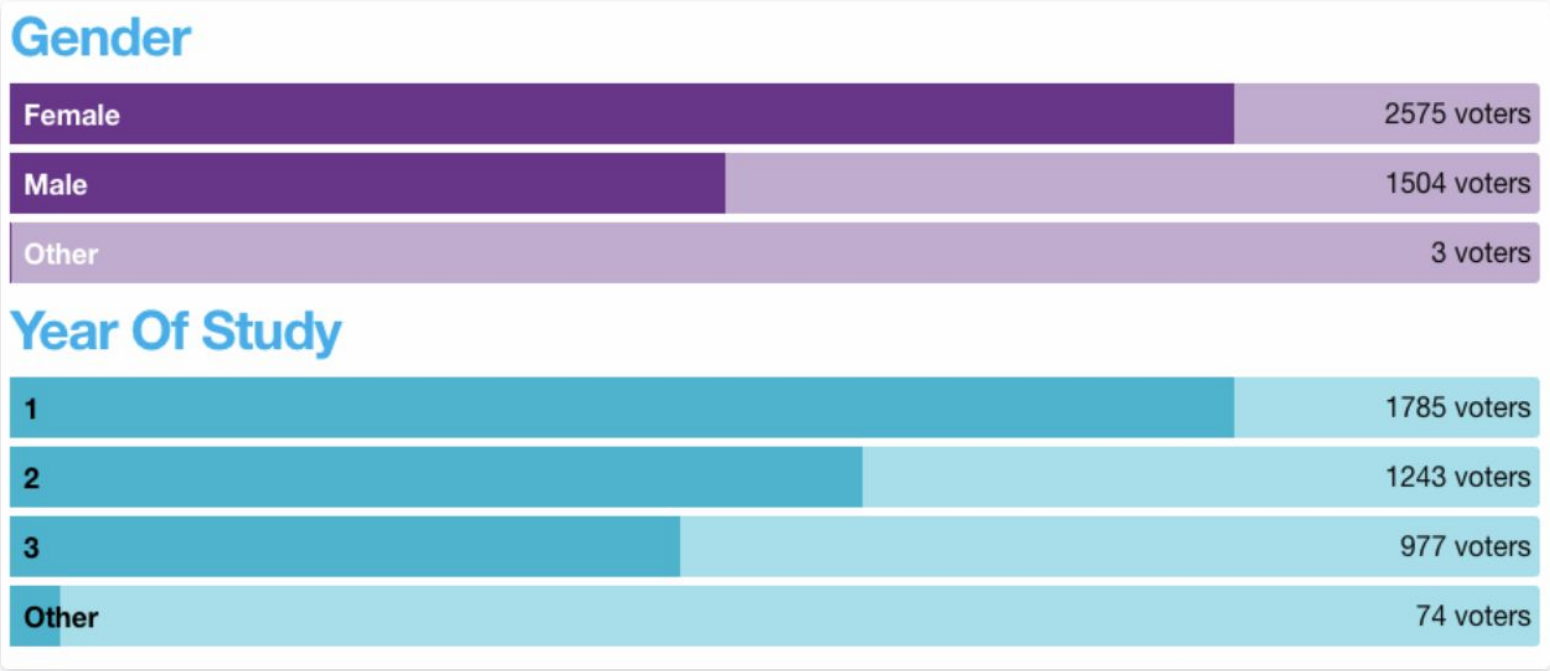
```
df = pandas.DataFrame(
columns=["service","details"] )
```

- Use a range of number as indices to extract data from multiple lists at the same time

```
for i in range(0,100):
  list1[i].text_content()
```

How to: find the data behind an interactive chart or map using the inspector

5 Replies



This interactive chart is generated from some data you can grab

Increasingly you might come across an interesting set of interactive charts from a public body, or an interactive map, and you want to grab the data behind it in order to ask further questions. In many cases you don't need to do any scraping — you just need to know where to look. In this post I explain how to work out where the data is being fetched from...

https://onlinejournalismblog.com/2017/05/10/how-to-find-data-behind-chart-map-using-inspector