Exercise 10.A: The subprocess Module

CS 1410

Background

In the slides for Module 7, you saw an example using **curl**, a command-line file transfer utility. In this exercise you will launch the **curl** command in a separate **shell subprocess** and receive the output of a transfer request into your Python script, and then saving it to a file. This exercise will prepare you for Project 7.

Data

The CIA website has graphics files of all the countries in the world. These files are available from the following web folder:

https://www.cia.gov/library/publications/resources/the-world-factbook/graphics/flags/large/

You can't just view that page in a browser, however—it is protected. To download a flag, you need the full URL including the name of the file the flag is stored in. The full URL for the USA flag is:

https://www.cia.gov/library/publications/resources/the-world-factbook/graphics/flags/large/us-lgflag.gif

The 2-letter code for the USA is "us", and the file name portion of the URL is "us-lgflag.gif". The codes for all 232 countries can be found in the file *flags.txt*, which you can find accompanying this exercise in Canvas. It is also used in Project 7. It looks like the following:

```
aa
ac
ae
af
ag
aj
al
…
wq
ws
wz
ym
za
zi
```

Requirements

Select 10 country codes at random from *flags.txt* and download each flag file in a file name ***xx.gif***, where **xx** is the 2-letter country code. The USA flag would be stored in the file *us.gif*, for example.

To download a flag, call **curl** in a *subprocess*, using the **subprocess** module, receiving its output back into your program. Then open a new file with the appropriate ***xx.gif*** name, for *binary output*, and save the downloaded content into that file.

Accumulate the length in bytes of each file and report the total number of bytes downloaded to standard output.

Submit a zip file that contains your code file, *flags.txt*, your screen output reporting the total number of bytes downloaded, and the 10 gif files downloaded.

Implementation Notes

You are calling **curl** as if from the command line, so you must use **shell=True** in your subprocess call. Since you are downloading raw bytes, you do *not* set **text=True** in your request. To find the number of bytes downloaded, just take the length of the content received with **len**.

You will notice that there will be unwanted output such as the following:

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed

  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  4377  100  4377    0     0  28607      0 --:--:-- --:--:-- --:--:-- 28607
```

This is because these headers are written to **stderr** (a different output channel) instead of **stdout**. By default, both **stdout** and **stderr** go to the console. To suppress **stderr** output, add one of the following arguments to your subprocess call:

```
stderr=open('/dev/null')      # Mac/Linux
stderr=open('NUL')            # Windows
```

These pseudo-files represent the *bit bucket*, which basically means "nowhere"—no output is actually generated.

The output we are concerned about, in addition to the 10 gif files, is something like:

```
Total bytes downloaded: 115970
```

You need to print that at the end of your program. Your number will vary depending on the flags randomly selected.

FAQs

Q. How do I randomly choose 10 country codes from *flags.txt*?
A. Read the codes into a list and call **random.sample**.

Q. Should I use **subprocess.run** or **subprocess.check_output**?
A. The **check_output** method is easier. In fact, **check_output** is equivalent to calling **run** as:

```
run(..., check=True, stdout=PIPE).stdout
```