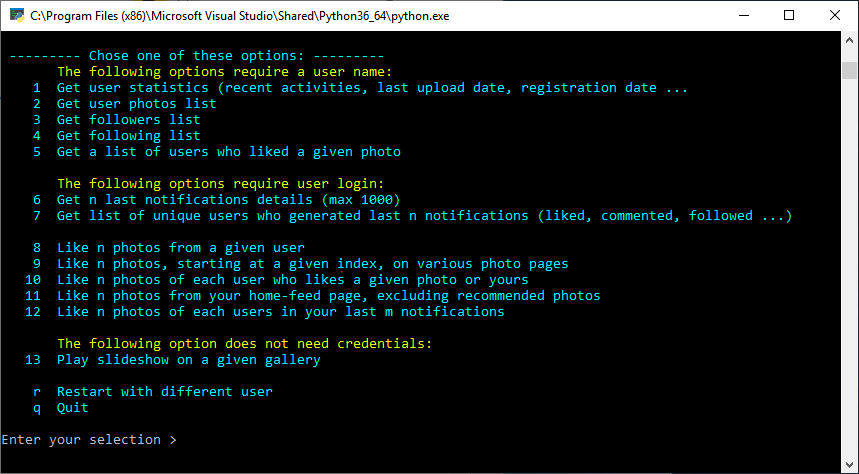
Menu



The first 7 options are for data collection

The results data are saved on disk in both CSV and HTML formats. User will be offered to view the html file on the web browser when the task finishes.

The next 5 options, from 8 to 12, are the automated processes, or bots, that will perform some actions.

The program can also run in un-attended mode, where it takes in command-line arguments and perform a single task.

A complete syntax of the command-line options is presented at the end of this document.

Following are the detailed description of each option and sample outputs.

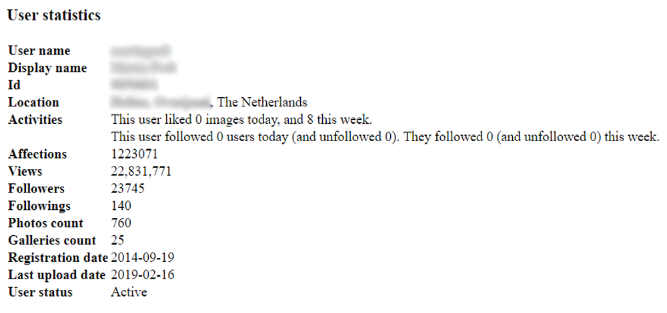
Please note that all the names and photo titles in this document are not real. They are either auto-generated, or blurred out.

Option 1: Get user statistics

Automatic processing:

* Open user home page [https://500px.com/[user\_name](https://500px.com/%5buser_name)]
* Run the document JavaScript to render the page content
* Extract interested data (using built-in lxml library)
* Use regular expression to extract the JSON part in the document body, and obtain more data from it.

Sample output:

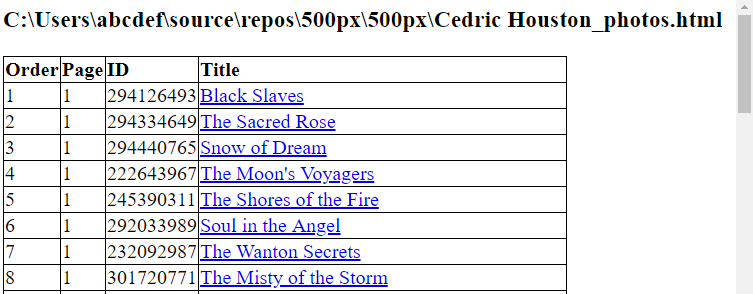


Option 2: Get User Photos list

Automatic processing:

* Open user home page. Scroll down until all photos are loaded
* Run the document JavaScript to render the page content
* Parse and extract photos details
* To construct the href of a photo, we need the page number where the photo was assigned to, the user id and user name

Sample output:



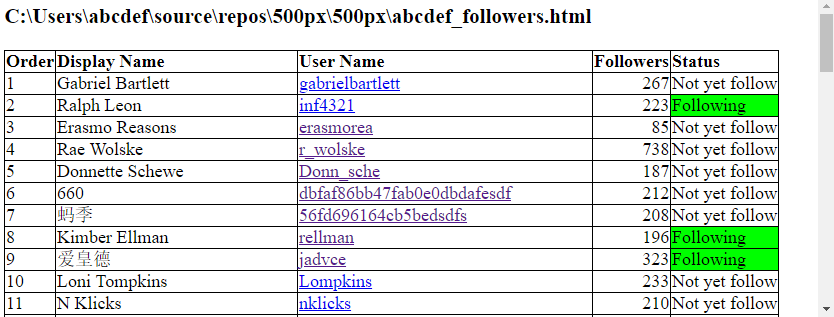
Option 3, and 4: Get Followers and Friends (Followings) list

Automatic processing:

* Open user home page, locate the text “Followers” and click on it to open a model windows containing the list of users
* Keep scrolling down until all items are loaded
* Run the document JavaScript to render the page content
* Extract data and write it to a list. An item contains Display Name, User Name and Number of Followers
* Write the list to csv and html file. When done, offer the user option to view the html file in the web browser

Note: The option Get Followers list does not require you to log in, but if you do, you will obtain the followed/not yet followed status of your followers

Sample output:



Option 5: Get a list of users who likes a given photo of yours

Automatic processing:

* Ask user for the link to the photo. Open it.
* Run the document JavaScript to render the page content
* Extract photographer name and the title of the photo
* Locate the Like count number and click on it to open the modal window containing all users who had liked the photo
* Keep scrolling down until all users have been loaded
* Extract info ( Display Name, User Name (with link) and Number of Followers
* Save list in to csv, html files. Offer to show the html file in the web browser.

Sample output:

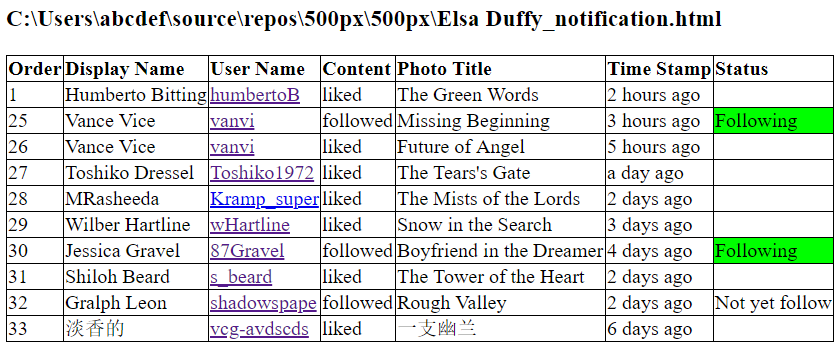
The output is the same as previous figure (in step 3, 4)

Option 6: Get n last notifications details

Automatic processing:

* Ask the user for desired number of notifications (n) and to enter the password
* Open the login page, submit the credentials. Then open the user’s notification page
* Scroll down (m) times for (n) notifications to show up on screen. (m) is calculated using NOTIFICATION\_PER\_LOAD=20
* Extract info, write to csv, html files. Offer to show the html file in the web browser.

Sample output:



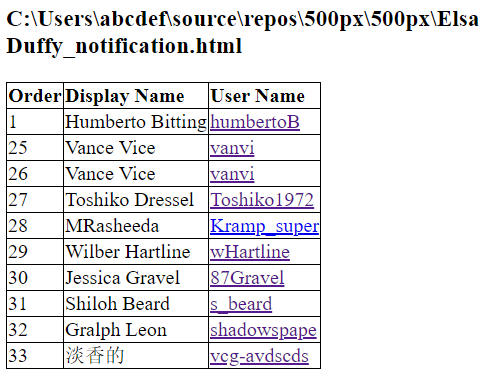
Option 7: Get list of unique users who generated last n notifications

Automatic processing:

Same process as in the previous option (Option 6: Get n last notifications). But the output list just contains the pairs of display name, user name.

Duplicated names are removed as well

Sample output:



Option 8: Like n photos of a given user

Automatic processing:

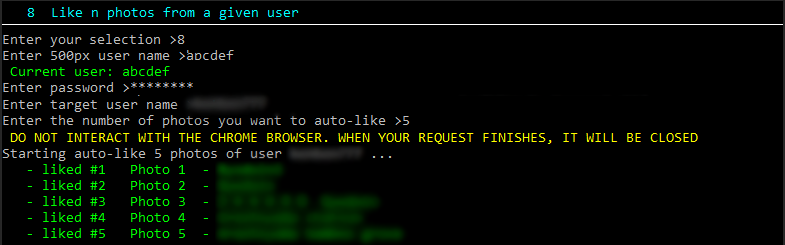
* Ask user for some inputs, as in figure below.
* Login, then open the given user’s home page
* Locate the first photo, click on like-icon if it is not yet liked, or move to the next photo if it is
* Continue until the asking number of photos is reached, scrolling down the page if needed

Note about the counting of the desired number (n):

When you process a photo, if it is already liked, you will count it as if it was done by the process. For example, if the request is three photos, and you found that in the first three photos in the list, two of them were already liked, then you just need to like one.

If you want to change this designed behavior, there is a boolean named include\_already\_liked\_photos\_in\_count. Set it to False.

Sample result:

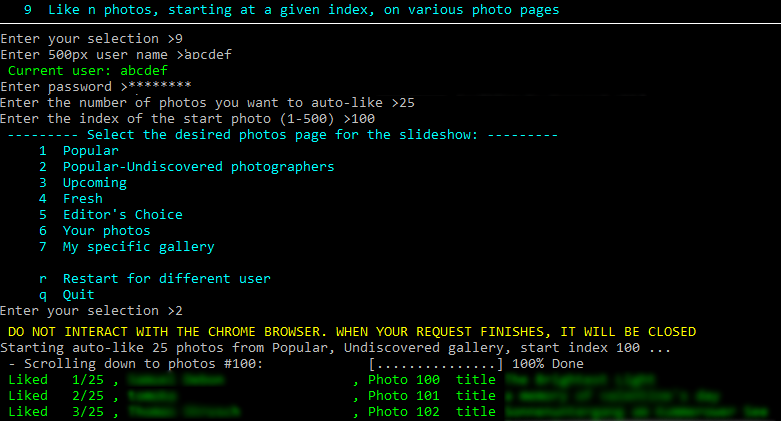


Option 9: Like n photos on various photo pages, starting from a given index

Automatic processing:

The process is similar to the previous option ( option 8: Like n photos of a given user). But it starts with the given photo page instead of the user home page.

Sample result: Like 5 photos, starting at photo #100, on the page Popular of Undiscovered photographers

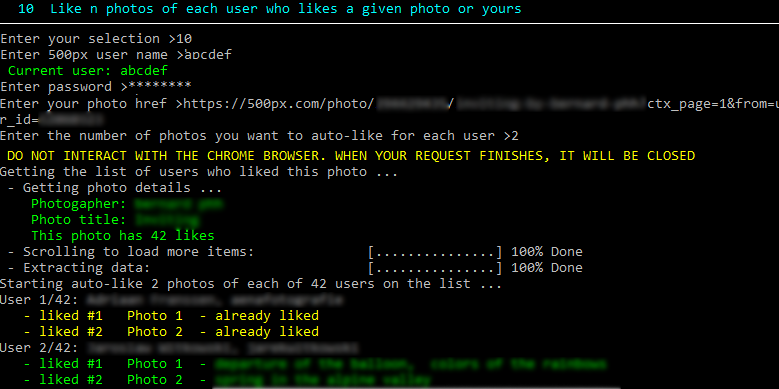


Option 10: Like n photos of each user who liked a given photo

Automatic processing:

* First step, do as in option 5 : get a list of users who likes a given photo
* Second step, for each user in the list, do as in option 8: like n photos of a given user

Sample result: Auto-like 1 photo of each user who liked a given photo, skipping already-liked photos



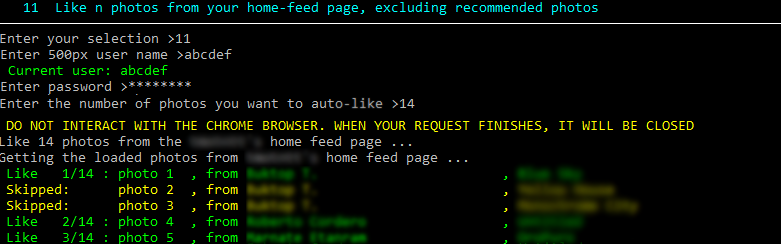
Option 11: Like n photos from the user’s home feed page, excluding the recommended photos

Automatic processing:

* Open the user home page, locate the 500px logo and click on it to open the user’s home feed page
* Get the list of all <img> elements on the page
* Remove from the list unwanted images such as thumbnails, avatar, recommended photos
* For each image elements on the list, like it if it is not yet liked, then go to the next image

Note: there are users who may upload 5, 10 photos at a time. We don’t want to like all of them. So the program will only process the first photo and will skip all consecutive photos of the same users.

Sample result: Auto-like (15) photos from the user’s home feed page, excluding the recommended photos and skipping consecutive photos of the same users

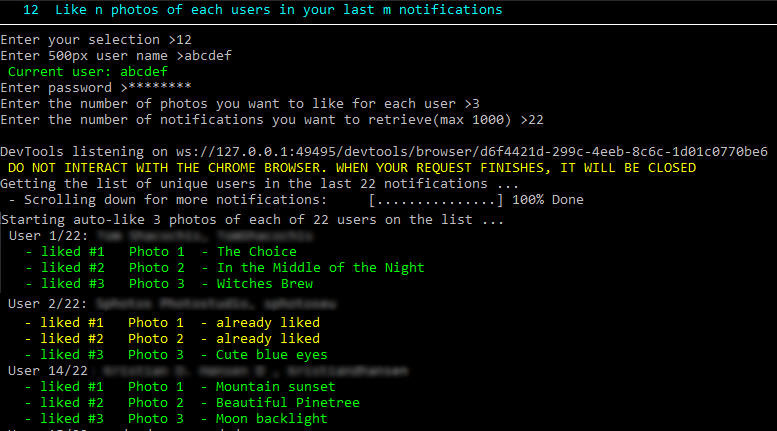


Option 12: Like n photos of each user in the last m notifications

Automatic processing: In general, there are two steps:

* First step, do as in option 7: get list of unique user who generated last **m** notifications
* Second step, for each user in the list, do as option 8: auto-like n photos of a given user

Sample result: Auto-like 2 photos of each user in the last 10 notifications



Option 13: Playing slide show on a selected gallery

Automatic processing:

* The user is asked to select or enter a desired gallery, and the time interval between photos
* Open the given gallery page in full-screen browser, with scroll bar and info bar hidden.
* Click on the first photo of the gallery to open the photo page
* Locate the Expand icon at the top right corner, click on it to expand the photo
* After the interval time, locate the Next icon at the middle-right side of the photo, click on it to advance to the next photo

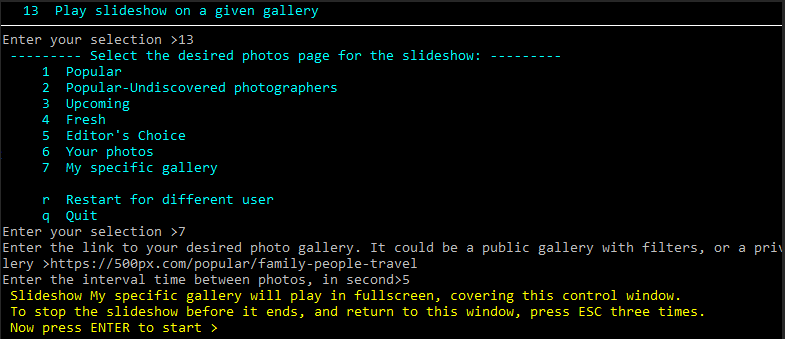
Note:

* In single-monitor system, the full-screen browser will cover the control terminal (console). To stop the slideshow before the last

photo is played, click on the browser area, then press “ESC” keys three times

* The user is asked to select one in 5 public galleries, as shown in figure below, or enter a specific photo gallery link
* Specific gallery may be a personal gallery of any user, or a category-filtered gallery. For example, a link of the popular gallery, with three categories family, people, travel selected, is: <https://500px.com/popular/family-people-travel>
* If you want to show the NWFW ( not-suitable-for-work) contents, the on-screen instruction will give you a chance to login. (The logged in user should have this “Show NWFW” turned on in the settings)

Sample result: Play a slideshow from a public or private photo gallery.



Command-line arguments

You can run un-attendedly a specific task by creating on the desktop a shortcut to the script, and feed it with arguments.

(Create a shortcut to 500px-APIless.px on the desktop, right-click the shortcut and choose Properties, then append the arguments to the Target box.

Following are available arguments switches.

There are two forms of switches, shorthand switches that start with a hyphen “-” or the full switches which start with double hyphen “--”.

The values in the parentheses are default settings

-c --choice ( 0 )

-u --user\_name (“”)

-d --password (“”)

-p --photo\_href (“”)

-g --gallery\_href (“”)

-l --number\_of\_photos\_to\_be\_liked ( 1 )

-i --index\_of\_start\_photo ( 1 )

-n --number\_of\_notifications ( 200 )

-a --target\_user\_name ( “”)

-t --time\_interval ( 4 )

Examples:

1 Get user statistics (recent activities, last upload date, registration date ...:

Full: 500px-APIless.py --choice 1 --user\_name JohnDoe

Short: 500px-APIless.py -c 1 -u JohnDoe

Mixed: 500px-APIless.py --choice 1 -u JohnDoe

2 Get user photos list:

Full: 500px-APIless.py --choice 2 --user\_name JohnDoe

Short: 500px-APIless.py -c 2 -u JohnDoe

3 Get followers list:

Full: 500px-APIless.py --choice 3 --user\_name JohnDoe --password \*\*\*\*\*\*

Short: 500px-APIless.py -c 3 -u JohnDoe -p \*\*\*\*\*\*

4 Get following list:

Full: 500px-APIless.py --choice 4 --user\_name JohnDoe

Short: 500px-APIless.py -c 4 -u JohnDoe

5 Get a list of users who liked a given photo:

Full: 500px-APIless.py --choice 5 --user\_name JohnDoe –photo\_href https://500px.com/....

Short: 500px-APIless.py -c 5 -u JohnDoe -p “https://500px.com/....”

6 Get n last notifications details (max 1000):

Full: 500px-APIless.py --choice 6 --user\_name JohnDoe --password \*\*\*\*\*\* --number\_of\_ notifications 500

Short: 500px-APIless.py -c 6 -u JohnDoe -d \*\*\*\*\*\* -n 500

7 Get list of unique users who generated last n notifications (liked, commented, followed..):

Full: 500px-APIless.py --choice 7 --user\_name JohnDoe --password \*\*\*\*\*\* --number\_of\_ notifications 500

Short: 500px-APIless.py -c 7 -u JohnDoe -d \*\*\*\*\*\* - n 500

8 Like n photos from a given user:

Full: 500px-APIless.py --choice 8 --user\_name JohnDoe --password \*\*\*\*\*\* --number\_of\_photos\_to\_be\_liked 50 –target\_user\_name JaneDoe

Short: 500px-APIless.py -c 8 -u JohnDoe -d \*\*\*\*\*\* -n 500 -a JaneDoe

9 Like n photos, starting at a given index, on various photo pages:

Full: 500px-APIless.py --choice 9 --user\_name JohnDoe --password \*\*\*\*\*\* --number\_of\_photos\_to\_be\_liked 50 --index\_of\_start\_photo 1

–gallary\_href “https://500px.com/popular/travel”

Short: 500px-APIless.py -c 9 -u JohnDoe -d \*\*\*\*\*\* -n 50 -I 1 -g https://500px.com/popular/travel

10 Like n photos of each user who likes a given photo or yours:

Full: 500px-APIless.py --choice 10 --user\_name JohnDoe --password \*\*\*\*\*\* --number\_of\_photos\_to\_be\_liked 50 --photo\_href [a photo link]

Short: 500px-APIless.py -c 10 -u JohnDoe -d \*\*\*\*\*\* -n 50 -p [a photo link]

11 Like n photos from your home-feed page, excluding recommended photos:

Full: 500px-APIless.py --choice 11 --user\_name JohnDoe --password \*\*\*\*\*\* --number\_of\_photos\_to\_be\_liked 50

Short: 500px-APIless.py -c 11 -u JohnDoe -d \*\*\*\*\*\* - l 50

12 Like n photos of each users in your last m notifications:

Full: 500px-APIless.py --choice 12 --user\_name JohnDoe --password \*\*\*\*\*\* --number\_of\_photos\_to\_be\_liked 50 -number\_of\_notifications 200

Short: 500px-APIless.py -c 12 -u JohnDoe -d \*\*\*\* -l 50 -n 200

13 Play a slideshow:

Full: 500px-APIless.py --choice 13 --user\_name JohnDoe --time\_interval 5 –gallery\_href https://500px.com/Popular

Short: 500px-APIless.py -c 13 -u JohnDoe -t 5 -g https://500px.com/Popular