|  |
| --- |
| **HIBP\_email\_checker\_V2** |

|  |  |  |
| --- | --- | --- |
| **Modules** | | |
|  |  | |  |  |  |  | | --- | --- | --- | --- | | [pandas](pandas.html) | [requests](requests.html) | [time](time.html) |  | |

|  |  |  |
| --- | --- | --- |
| **Functions** | | |
|  |  | **main**()  Specifies the content to be passed into iterate\_over\_csv, calls Iterate\_over\_csv with file paths and start row, and prints out  the emails and row numbers of the rows in the CSV that made a  bad request to HIBP’s API (Response status\_code = 400 or 403).  **iterate\_over\_csv**(file\_path\_with\_emails, content=None, append=False, start\_row=0, stop\_row=0, lookup\_file\_path=None)  Reads a CSV file\_path and a start\_row for which row to start  processing data on. The CSV file should be the All CAR Members  list from Pardot.  :param file\_path\_with\_emails: The file path that contains the      emails to be checked. :param content: Either a list of headers if new CSV is      wanted as a result, otherwise, provide a file path      that will be used to append the results to. If a file path     is provided, 'content' is then to be used in conjuction with  'append'. If 'content' is a file path and 'append' is True,  the results of the program will be appended to the file  provided by the file path. If append is False, a new CSV  will be created and all of that file's headers will be  used as the fields to be found. If the latter is the     case, be sure that the headers of the content file have the    same exact headers as wherever the data associated with  the email is coming from.     Default is None, i.e. create a new CSV with all headers and    data associated with the emails within  'file\_path\_with\_emails'. :param append: If 'append' is True, the program will append the    results to the file provided by content. If 'append' is  False, the program will create a new CSV with the file  name specified by the global variable 'save\_to\_file\_path',  but with the same headers as the file specified by content. :param start\_row: The row to start processing data on in      email\_file\_path. :param stop\_row: The row, inclusive, in the CSV file to stop  processing data on. :param lookup\_file\_path: If the data associated with the emails     exist in another file, pass the file path here. Should be  used if for example, 'file\_path\_with\_emails' only has a  specific list of emails to check, but the data associated  with the emails exists within a different file.  **process\_email**(index, email, email\_df, lookup\_df)  Calls check\_if\_pwned. If True is returned, add email to  pwned\_emails and add all data specified in content to DataFrame.   :param index: The current row we are at in 'email\_df'. :param email: The email we are processing. :param email\_df: The DataFrame that contains the emails to be      processed. :param lookup\_df: The DataFrame, if not None, that contains the     data associated with the email if 'email\_df' does not have  the data.  **check\_if\_pwned**(email)  Method to check if the email has been pwned  (Response status\_codde = 200), if email has not been pwned  (Response status\_code = 404), or something went  wrong. Either an error (Response status\_code = 400 or 403),  or too many requests have been made  (Response status\_code = 429), and we must wait until we can  make another request to HIBP’s API at:   https://haveibeenpwned.com/API/v3.  :param email: The email to be checked if pwned.  **pwned\_emails\_to\_csv**()  Converts pwned\_emails\_df DataFrame that contains the pwned emails that  Have been found to a CSV file, then prints the current row we are at  within the CSV file that contains the emails to be processed. |

|  |  |  |
| --- | --- | --- |
| **Data** | | |
|  |  | **API\_KEY** = 'PRIVATE\_API\_KEY’ **HIBP\_URI** = 'https://haveibeenpwned.com/api/v3/breachedaccount/' **bad\_excel\_rows** = [] **headers** = {'hibp-api-key': API\_KEY, 'user-agent': 'CAR HIBP Email Checker'} **pwned\_emails** = [] **pwned\_emails\_df** = Empty DataFrame Columns: [] Index: [] **pwned\_emails\_df\_row\_log** = 0 **save\_to\_file\_path** = '/Users/username/Desktop/File Name.csv' **user\_agent** = 'CAR HIBP Email Checker' |

Flow / Order of Operations:

main() calls iterate\_over\_csv() with CSV file that contains emails to be checked if they are pwned. iterate\_over\_csv() makes pandas DataFrames from the file paths provided by main() and iterates through the rows of the CSV file that contains the emails to be checked to obtain the email and passes the email, index, the DataFrame made out of the CSV file that contains the emails to be checked, and another DataFrame made out the parameter ‘content’ to process\_email(). Process\_email() in turn calls check\_if\_pwned() passing in the email, which returns True if the email is pwned, and False if it is not. If the email is pwned, process\_email() then adds the email and the data associated with it to a DataFrame created by iterate\_over\_csv() but is visible globally in the program. This continues until there are no more emails to process or if the parameter ‘stop\_row’ was specified when main() called iterate\_over\_csv().