

Forum:[Hack Forums](#)

Thread: [Minecraft Account Checker](#)

AuntJason - February 13, 2016, 6:26 pm

I wrote a small Minecraft Account Checker in Python 3. You have to fix the Links since i can't post them. It can filter out migrated, demo and unloginable accounts.

Enter the Accounts in accounts.txt. If you have the password enter them <username>:<password>

Edit the Python file to filter out demo, migrated or unloginable accounts at will.

Run the file

PHP Code

```
<?php
import requests
import json
import logging
```

```
class MCAccountChecker(object):
```

```
    def __init__(self, usernames):
        self.userlist = {}
        for username in usernames:
            if ':' in username:
                self.userlist[username.split(':')[0]] = {'password' : username.split(':')[1], 'checked' : False}
            else:
                self.userlist[username] = {'checked' : False}
        logging.basicConfig(level=logging.INFO)
        self.logger = logging.getLogger(__name__)
```

```
    def filter_migrated(self, invert = False):
        self.fetch_users()
        remove = []
        for user in self.userlist:
            if self.userlist[user]['legacy'] is invert:
                remove.append(user)
        for key in remove:
            self.userlist.pop(key)
        return self.userlist
```

```
    def filter_demo(self, invert = False):
        self.fetch_users()
        remove = []
        for user in self.userlist:
            if self.userlist[user]['demo'] is invert:
```

```

        remove.append(user)
    for key in remove:
        self.userlist.pop(key)
    return self.userlist

def filter_login(self):
    self.fetch_users()
    remove = []
    for user in self.userlist:
        if 'password' not in self.userlist[user]:
            continue
        if self.userlist[user]['password'] is None:
            continue
        payload = {
            "agent": {
                "name": "Minecraft",
                "version": 1

            },
            "username": user,
            "password": self.userlist[user]['password'],
        }
        response = requests.post('https authserver mojang com/authenticate', data=json.dumps(
payload))
        if 'error' in response.json():
            remove.append(user)
    for key in remove:
        self.userlist.pop(key)
    return self.userlist

def fetch_users(self):
    # Save on API Calls and Network Usage
    account_check = []

    for account in self.userlist:
        if not self.userlist[account]['checked']:
            account_check.append(account)
    if len(account_check) == 0:
        return self.userlist

    chunks = [account_check[x:x+100] for x in range(0, len(account_check), 100)]
    if len(chunks) > 60:
        self.logger.error(
'Mojang API is limited to 600 Requests per 10 Minutes and you try to make {requests} Requests'.
format(requests = len(chunks)*10))
        chunks = chunks[:60]

    for chunk in chunks:
        payload = json.dumps(chunk)
        response = requests.post('https api mojang com/profiles/minecraft', data=payload)
        # None of the given Accounts exist
        rsp_json = response.json()
        if len(rsp_json) == 0:

```

```

        continue
    # Error in the API Response
    if 'error' in rsp_json:
        self.usernames = None
        raise ValueError('Minecraft API Failure : {error} : {errorMessage}'.format(error=rsp_json[
'error'], errorMessage=
        rsp_json['errorMessage']))

    for account in rsp_json:
        self.userlist[account['name']]['checked'] = True
        self.userlist[account['name']]['demo'] = 'demo' in account
        self.userlist[account['name']]['legacy'] = 'legacy' in account
        self.userlist[account['name']]['uuid'] = account['id']
    return self.userlist

accounts = []

with open('accounts.txt') as file:
    for line in file.readlines():
        accounts.append(line)

print(json.dumps(MCAccountChecker(accounts).filter_migrated()))

```

ExoticModzV1 - February 16, 2016, 5:53 pm

(Feb 13, 2016 07:26 PM)AuntJason Wrote: [\[CITE\]](#) I wrote a small Minecraft Account Checker in Python 3. You have to fix the Links since i can't post them. It can filter out migrated, demo and unloginable accounts.

Enter the Accounts in accounts.txt. If you have the password enter them <username>:<password>

Edit the Python file to filter out demo, migrated or unloginable accounts at will.

Run the file

PHP Code

```

<?php
import requests
import json
import logging

```

```

class MCAccountChecker(object):

```

```

    def __init__(self, usernames):
        self.userlist = {}
        for username in usernames:
            if ':' in username:
                self.userlist[username.split(':')[0]] = {'password' : username.split(':')[1], 'checked' : False}
            else:
                self.userlist[username] = {'checked' : False}

```

```
logging.basicConfig(level=logging.INFO)
self.logger = logging.getLogger(__name__)
```

```
def filter_migrated(self, invert = False):
    self.fetch_users()
    remove = []
    for user in self.userlist:
        if self.userlist[user]['legacy'] is invert:
            remove.append(user)
    for key in remove:
        self.userlist.pop(key)
    return self.userlist
```

```
def filter_demo(self, invert = False):
    self.fetch_users()
    remove = []
    for user in self.userlist:
        if self.userlist[user]['demo'] is invert:
            remove.append(user)
    for key in remove:
        self.userlist.pop(key)
    return self.userlist
```

```
def filter_login(self):
    self.fetch_users()
    remove = []
    for user in self.userlist:
        if 'password' not in self.userlist[user]:
            continue
        if self.userlist[user]['password'] is None:
            continue
        payload = {
            "agent": {
                "name": "Minecraft",
                "version": 1

            },
            "username": user,
            "password": self.userlist[user]['password'],
        }
        response = requests.post('https authserver mojang com/authenticate', data=json.dumps(
payload))
        if 'error' in response.json():
            remove.append(user)
    for key in remove:
        self.userlist.pop(key)
    return self.userlist

def fetch_users(self):
    # Save on API Calls and Network Usage
    account_check = []

    for account in self.userlist:
```

```

        if not self.userlist[account]['checked']:
            account_check.append(account)
    if len(account_check) == 0:
        return self.userlist

    chunks = [account_check[x:x+100] for x in range(0, len(account_check), 100)]
    if len(chunks) > 60:
        self.logger.error(
'Mojang API is limited to 600 Requests per 10 Minutes and you try to make {requests} Requests'.
format(requests = len(chunks)*10))
        chunks = chunks[:60]

    for chunk in chunks:
        payload = json.dumps(chunk)
        response = requests.post('https api mojang com/profiles/minecraft', data=payload)
        # None of the given Accounts exist
        rsp_json = response.json()
        if len(rsp_json) == 0:
            continue
        # Error in the API Response
        if 'error' in rsp_json:
            self.usernames = None
            raise ValueError('Minecraft API Failure : {error} : {errorMessage}'.format(error=rsp_json[
'error'], errorMessage=
            rsp_json['errorMessage']))

        for account in rsp_json:
            self.userlist[account['name']]['checked'] = True
            self.userlist[account['name']]['demo'] = 'demo' in account
            self.userlist[account['name']]['legacy'] = 'legacy' in account
            self.userlist[account['name']]['uuid'] = account['id']
    return self.userlist

```

```
accounts = []
```

```

with open('accounts.txt') as file:
    for line in file.readlines():
        accounts.append(line)

```

```
print(json.dumps(MCAccountChecker(accounts).filter_migrated()))
```

i thought this might be usefull to you to give you an idea on how to shorten code
and this a premium account checker, so you know your tool doesnt have
you can easily shorten this down via an api, here i made a code you can look at this, its more
simpler and will shorten your code alot i created this along time ago, but its still working perfectly,
you should probably and some erroe handling such as IOError

PHP Code

```

<?php
from bs4 import BeautifulSoup
import urllib2
import webbrowser
print("coded by ImHereForLeak")

```

```
wfile = open('user.txt', 'r')
```

```

wfile2 = open('premium.txt', 'w')
wfile3 = open('nonpremium.txt', 'w')
print("CODED BY @leakforums . net @ImHereForLeak")
print("you must create a txt file called 'pass'\nwhere the usernames should be stored")
raw_input("press enter to start!")
counter = 0
for line in wfile:
    url = " http : // api . predator . wtf /minecraft / ? arguments="+str(line)

    counter = counter +1
    source_code = urllib2.urlopen(url)
    soup = BeautifulSoup(source_code)

    print("[ " + str(counter) + "]" + str(line) + str(soup))
    if str(soup) == "This User Is Premium!":
        wfile2.write(line)
    else:
        wfile3.write(line)
wfile2.close()
wfile3.close()
raw_input()
'''
leave credits if you use this please !
'''
but nice code bro

```

AuntJason - February 17, 2016, 5:23 am

Libraries have the downside that they're less easy to oversee and that's not good for building Trust.

emadaldin1 - February 19, 2016, 11:09 pm

Nice job thanks for the share :D

bet4r00t - March 3, 2016, 9:15 am

Haha i lost my mc account and I'll try this stuff out to try getting my account name xD