@greghur (@ greghur)

ディスプレイごとにスクリーンショットが撮れるソフトを作ってみた

Python 初心者 Tkinter プログラミング初心者

最終更新日 2023年06月30日 投稿日 2022年12月31日

初めに

私は理系大学2年生です。大学に入って初めてプログラミングに触れたため経験歴はとても低いです。そんな中、目標を立てて作ってみたソフトを公開してみました。

作ったもの

タイトルにも書いてある通り、ディスプレイごとにスクリーンショットが取れるソフトを作りました。
デュアルディスプレイ、マルチディスプレイ環境にてスクショを撮ると、画像がつながった状態になっています。そこで、ディスプレイ一つ一つでスクショが撮れるソフトを作りました。

使用言語　python 3.7
確認が取れている動作環境　Windows10(64bit), windows11
　pcのスペックが低いと動作が重い可能性があります。
exe化に使用したもの　Nuitka

ソースコード

長いですが、お許しください。ご自由に使っていただいて結構です。

```
from tkinter import *
from tkinter import filedialog
import tkinter as tk

from pynput import keyboard                                 #LGPLv3
from PIL import ImageGrab, Image                            #HPND
import getpass
import os
import datetime
import time
from concurrent.futures import ThreadPoolExecutor
from screeninfo import get_monitors                         #MIT
import cv2                                                  #MIT
from pystray import Icon, Menu, MenuItem                    #LGPLv3
from win32 import win32gui                                  #PSF
import gc


import sys

f = open(os.devnull, 'w')
sys.stderr = f
sys.stdout = f
sys.stdin = f

#Global variable definition
user = getpass.getuser()
dir_path = 'C:\\Users\\{}\\Pictures'.format(user)
filename = ''

dis_count    = 0                                            #ディスプレイの枚数

temp_windowx_cd   = [0]        #仮のディスプレイの左上の座標 X
temp_windowy_cd   = [0]        #仮のディスプレイの左上の座標 Y

windowx_cd        = [0]        #最終的に使う数
windowy_cd        = [0]        #最終的に使う数

windowx_size = [0]        #ディスプレイの解像度(サイズ)
windowy_size = [0]        #ディスプレイの解像度(サイズ)

min_x_cd = 10000000  #初期設定  数は適当
min_y_cd = 10000000  #初期設定  数は適当

for d in get_monitors():
    dis_count = dis_count + 1

#Add display information to the array
```

```
Range = range(0, dis_count)

for data in Range:
    monitor = get_monitors()[data]
    temp_windowx_cd[data] = monitor.x
    temp_windowy_cd[data] = monitor.y
    windowx_size[data] = monitor.width
    windowy_size[data] = monitor.height

for i in Range:                             #座標比較
    if temp_windowx_cd[i] < min_x_cd:
        min_x_cd = temp_windowx_cd[i]

    if temp_windowy_cd[i] < min_y_cd:
        min_y_cd = temp_windowy_cd[i]

for j in Range:                             #切り抜きに使う数値へ変更
    windowx_cd[j] = -1 * min_x_cd + temp_windowx_cd[j]
    windowy_cd[j] = -1 * min_y_cd + temp_windowy_cd[j]


icon_data = """
R0lGODlhQAFAAfZ7AAAAACElliMnmCMomCcrmScrmiwwmy4wnS0ynDAznTE0njM0
njE2njY6oDc7oDs+ozs/pD1Aoz9BpEBDpEFEpUZKqUhMqUpMqUxQqk1QqlNVrVNW
rlVXrlVXr1lbr1ZYsFZZsFpdslxfsVxfs11hs2JmtWZouGZpuGhruGptuWxtumxv
um1uu3B0vXJ1vHd8wHp9wHp9wX6AwX+Bw3+Cw4CDwoGExIKExIOExYeJx4qMx4mM
yImMyYqMyJCTy5eZzpiazpud0J+g0aCh0qGj06Kj06Kk1KOl1amq1qqs1qqs16us
16qs2K6v2a+w2rO127i53rq63sTG48TG5MvM583P6M7O6M/P6NLT6tXV69TW69bX
69fX7NXY69jZ7NjZ7dna7dra7tvb7tzc79/g7+Dg7+Dg8OHh8erq9err9e/v9+/v
+PHx+PLy+fLz+fPz+fT0+fT0+vT1+vb2+vb2+/f3+/j4+/j4/Pv7/fv8/fz8/v//
/wAAAAAAAAAAAAAACH5BAEAAAAAIf8LWE1QIERhdGFYTVA8P3hwYWNrZXQgYmVn
aW49J++7vycgaWQ9J1c1TTBNcENlaGlIenJlU3pOVGN6a2M5ZCc/Pgo8eDp4bXBt
ZXRhIHhtbG5zOng9J2Fkb2JlOm5zOm1ldGEvJyB4OnhtcHRrPSdJbWFnZTo6RXhp
ZlRvb2wgMTIuNDAnPgo8cmRmOlJERiB4bWxuczpyZGY9J2h0dHA6Ly93d3cudzMu
b3JnLzE5OTkvMDIvMjItcmRmLXN5bnRheC1ucyMnPgoKIDxyZGY6RGVzY3JpcHRp
b24gcmRmOmFib3V0PScnCiAgeG1sbnM6dGlmZj0naHR0cDovL25zLmFkb2JlLmNv
bS90aWZmLzEuMC8nPgogIDx0aWZmOk9yaWVudGF0aW9uPjE8L3RpZmY6T3JpZW50
YXRpb24+CiA8L3JkZjpEZXNjcmlwdGlvbj4KPC9yZGY6UkRGPgo8L3g6eG1wbWV0
YT4KICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAg
```

```
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAK
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIAogICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAKICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
IAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAKICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgIAogICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgCiAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAKPD94cGFja2V0IGVuZD0ndyc/PgH/
/v38+/r5+Pf29fTz8vHw7+7t7Ovq6ejn5uXk4+Lh4N/e3dzb2tnY19bV1NPS0dDP
zs3My8rJyMfGxcTDwsHAv769vLu6ubi3trW0s7KxsK+urayrqqmop6alpKOioaCf
np2cm5qZmJeWlZSTkpGQj46NjIuKiYiHhoWEg4KBgH9+fXx7enl4d3Z1dHNycXBv
bm1sa2ppaGdmZWRjYmFgX15dXFtaWVhXVlVUU1JRUE9OTUxLSklIR0ZFRENCQUA/
Pj08Ozo5ODc2NTQzMjEwLy4tLCsqKSgnJiUkIyIhIB8eHRwbGhkYFxYVFBMSERAP
Dg0MCwoJCAcGBQQDAgEAACwAAAAAQAFAAQAH/oAAgoOEhYaHiImKi4yNjo+QkZKT
lJWWl5iZmpucnZ6foKGio6SlpqeoqaqrrK2ur7Cxspprtba3uGuDHry9vr+/BcLD
xMXELCrJKiwsKcrPyczJWNTV1tfXhHrb3N3cs4tl4uPk5eUpKOnq6ejr7igO8fLz
9PQk9/j5+vmCav7/AAP6U5WrYEFgCBEaW2jMmbIUyKA9azYNm0Vs3jJqBJfInEdz
```

```
Y96JHFmvZMl9KPexEchSYCqDMG0lnNmLoU1hESXqhHaxZzWNQLtxPPSx6LiRSNeZ
XCovpdN7LaP+exkTJk2aN23m3MnVp8+gYPUMLXTGaNF2SUkyXfrUqVSp/lSrHrya
MCtDrngrerUYNuhYQmaLpk26lm1blG+jxpWLi25du8byck2x92JfoH8HBf44GGlh
k4cRJ2a5mLFMx8AgR5YskVllvpczZha02WNntZ/thdY3mjQq07lQp1ZNjLVEiK8x
xvY2G0Btc7dF5q63m3fvgKWBCw9GfJhxncmvLWc++3m56O+m664O9TrA7Ka3++ru
/TvP8D/Hfytv/ih6peo1xV577g30G3C3yFcTfQXYdx9+WOi3X2b9+fdfOgEKOGCB
Ux2IYC0K8sJggw4qAyEWWUi4TXMVinOhOhnGMyCBBcLHWIgejFiiiSeqKBZ/Fb6I
YYwzksChgad8/niagjruqMKJEUrYnHMtovVfjA4UeaQaNsqFY5M7QqnilC2GJCSW
Wh7ZZVVfMujkkz2ONyVtLQoJD5EzbrlmTG3S96aYfs1JVJ1n4rkhh6souUaf3Rnn
0DOAyiYoIi2WgZaVg6GZJ6IEKckocawxA1E0LKAIIWaTUtqfmSikcEJ0KWh6aI2V
0GHrrbjmSgchQPTq66/AAjvBsMQWa6yxTiSr7LLMLpvCs9BGK+20HXzwgQbWZqvt
ttae4e234IYbLiriljtuCOimq+666ooQw7vwxivvvEPUa++9+OK7wb789utvvz8E
LPDABBNcq64I3zpIsAwzfOzDDzcrcbPT/lZsMbcYY2zuxt+Sy7G57IYc8rwkl5zv
ySf/q/K/Bbfc8sEJI7xwwzT3CvHNxE6sc7IW9xxtxkBn+/HGHg8drshIp1vy0vGi
7LS9K0e9r8tUCwxzzLnOXHPDOOO8s84+hx100EaXW3TZ3iadNNNsP/201FFXXfXV
WOMqyNY0d33z1xOH7fPYQKM97imCf6s20mwz7bbTcK8sN9V0123r3Xg7rHfEfFPs
t8UdVAv4toV3THjoh4uc+NKLo9y4yo+7HLnklFcu7OXHZq755tN+4PnnQod+xtlo
lz7y6SSnnvLq/rb+MiWS6xq77L7SXrvtzuKeO++g+w582cKzS3zx/sbbS4QQyCev
vMHMN283ANDPLn3O1Ctr/fXY9x769kZ3v+739IZfrxDkKx+/zoe+Sahvfe371fuK
FT/5ze9n9bNf4fA3NP2pi3/y8t+9OifADRBwYK+r2/Pat0D4NfCBEIzgB3z3u9EV
zoJKwyC8NGivDmiggx+0WvoOuCv2JdBmJZxAA3mGwmepsFvac6HgYIguGc6QhkPQ
gA0FmMOAhRBrI4ReEIU4xCJC5IgrTKIpfMfEEDjxXVAcwhSpWMUrxiyLstviEJ3g
xRSAkYUU/FgZzxgD/42PCETo4NTauMMDwrFycuyiF+8oxlKQkYl89J8QiBCEQAqy
ij9wY8IO/om3RJ5wkSrsAB6VWLYy7PGMfgRgAHFISAPysIc/BGIJ51jHI4qykaR4
JAwjGb5JCiEI/Loh8jCpyU3GEgiejF8tjzjKMZKOiTSggRP9WEk12rAD/+Igv7Dp
wVZK4pWTO2YyqbdMFTbTkc/0wLrUiS52pkudLkBlLwGpRikKs1/a3BcHiUmJN/jz
nwANKEABgIOCGvSgCEVoDxbK0IY61KGo6JxEJ0rRii7hohjNqEaXkASMooIJIA2p
SEGahJEy4QMcSCkHUJpSDWBLpSi1lgZSwYOa2vSmOL0pDXBAgxnQ4AZAvQEOgkpU
oOKgEgJNqkAJmtCmJvShUIVqRCtK/lWLbvSqF2WCR09h0q6KNAlJYGlKxZotlbbU
WjTNqVpxitCiupWoSFWqXN/AVKfaFQdRzStDp1rVvnYAq4Ddqim8SliQitWsiE0s
B9K61sYW9KdvjewN4jpXpd71rnrVK1/9StXABvajhe1qWBVLWrMytrFqxYENbCDZ
yFK2sgK9rF0zm9fNcpaingUsaEM70iRsoLSl/cBpUYtTGtggB61161H7Cdukytap
tI2qbW8r0dxidbe8DelogYtYmQ6XuDY1LmuTS9TlTqK5zn3uU6P70OlS96/W3Sh2
s2vYa3E3sd8FLw8gS96ivha9/lTvetnbUPdSN77y5Sp9RYqt/sPed6ao0G9Oh9rf
8v4XwAJWKIELfIr3WhXBgi3FgrXrYO5mK7/gpXCFg3ph9Gb4oBvmsCk8jFsQh5gU
IyZpibmLLRSjtgf8XfFkmQvgf77YoDHea4dpXF0bX3S+9NXAfRMr3AhL+KYqXrF5
JVFkgB65oEleqIFv6+QnK3jE252ySiF8iitjWcgsJnKXv4zXMI+Zs2VmApSzu+P7
+litOwg0nON83i4H+Mth7sGd/ZrnPfNWymo2rZUlHOgdDFqoLW4unRO96L6WeQmO
Lmya1YzWSbt5vHDONGw3beclMxm+Tg41YUc95Wv9ubGo1rKq51rXF3Pa1Uz+tKwJ
C+lI/q/01mvN9Yp3rVRBsDrJna6qsM+8YFpPmc2mcLNNlV1hZi/V2Yhu9YxfDWsb
gzaroS3pSdW8AbGmNQc5uGm8bxrobe+UqPeW7JYjAeBCPDvG0e5smc+NBJCiG90k
LWmxTcxSbJfCpjGQ97x5UOma9iAHBS1vljMOV0rYoYeQAKvIR07ykjPj5ChPucoL
4YWWt9wTF4i5zGdOc5rP4OY4v3kNbhCDnPscFTWQgdBjIHQZxCDoMqhBDaRZdAw4
/elQjzrUM5CKF7ygBFbP+gtS8IIVdL0FYAc7SZkgcrKTHaxmL3utJlHytrdd5XCH
uyBcTve6u/wSNc973n2Oc57f/oDvOEfF0YWOdBnwvOg3cAENmi71xk+d6qjQuglI
YHWuc93rYRd7STef9s5zPgmpcLvoRR730p987nZP/csroffWyxzwsP/5KYpO+9rb
XgaOz/3Tq551E4zA6iYAvvBfEPwjfN7zyNczKkY/etObHgCqj74lXO/62Ft/Bqi4
vfZpr/vcQ/4UVv9974kffOKT3+pHQP7xPw/65TPf7c4vPfSjn/rpU1/v14999re/
/e7nnvdZ93viB3zlZ37Gp34I2H6n8H7wF39yR3/1x3r3t3f5B3j7x3+3lwH+J3Xf
```

```
ZwoCqHXmZwLlV4AvkH7rl4DKt4AMaHIOuHIQaHf2N4E2/leBfHeBGFh7GriBUVd1
IxB8PhiC5nd+LxAEJ1iEZBd6K0hyLeiCL1h3EiiDM0eDNTh7N2h7OaiDTteBpaB1
AziAw1d8KHiCSJiEpLeEKDd/TXh3lACFMyiFgUeFVUh7V4iFGMCDLzCABSiCQVh8
RniCKWgKZDhyZniGaUh3T8iGF+CGOWeDcSgDc4iFAOiDI/iDQViCYeh5CgiIgQhW
g3h6haiGk4CIMhdNMfB3UogKP1V4VUiHT6eFpACCvgeCw0d86SdaCPiHpbCJnNiJ
LICGhXiIbEiKpkiDqOgCjSh0rLh7kXeHsvh7JDCJ6McEtRhSBVeEmZiLusiLLPCJ
/qAoCaIYczc3jMQIh8eYjE4HgMz4geP3g7OWgGMYiNrIjasXit+oiG9oCsfIeKzo
iqPwgXlYiZQ4jSZlhO9IhvEoj8AIhfZ4c4wYh+ZYh8soeeu4h5Zoi0VYkEmojb6Y
hgkpgwuJfeTYiObIj6IggD74eyMYguUnagiIkSuokQi5hvWoiKXYkKtojryXki9w
AlbHkyRoAgIZUgfohy7JgDD5iTGIiPZ4AzVgkzc4klWnk5SYkmDoVdZYlCOHBFqZ
UUlwlE34BXj3jTEQTWQZTT5VijVggacQTfn4kKngAi7wAixgdV7Xky+QeWE3duzn
jqmQBn7pl2gQmIIZmGlQ/ggicJiImZiKqZgJ0JiO+ZiQmQAI0AAIAAGWeZmYmZma
OQCc2Zme+Zmf6VKiKZr7Mpqm6XCPgAVXsJqs2ZquuZpVAAAZcAGsGAC2eZu4mZu5
iQAIcAC8+ZvAGZy86ZuN2QAJoAAK0JgH0ADImQDGeZzJmQAkEAIjMAIggC4gUJ3U
WZ3cWZ2D+Z2A+Z2BuQppIJ6DWZ6FSQiLuZ6KSQIiEJnwmQAH4JzzGQGaeZ/3CZr6
qZ+n2Z/9OQll8JoCKqCPuIG6eaAHKpwKuqDI2aANCp0N2gAH4KDbyZ0V2p0Yap6E
aZ5+uQoaKph/aZjsOaIeQALxGZnIeQC+iZ8sipn7/vminemfMiqak3AGA3qjrFmg
/oegPHqbC/qjwOmgDiqhDbqcEKoAJIChSrqkHxqe59mhqOCkH4oGIjqiI3qiWNqi
WgqjMDqjM0oJOIqj5tijPQqkZiqkaJqmyLmkbNqdUzqlfVmeb0ql6mmlV4ql8Wmf
WoqfXPqiXiqjYBqmAzqmZIqgZgqkapqoDdqmjDqnGtqXb4qeVWqni4mnebqnfNqn
n1kAAfCn/hmogvqahFqounmoP6qoisqobCoCjnqeaJCepyCnTQqlg0Cp7Gmp8Kmn
mJqZmvqZAUAAnvqfkxCqAjqqpIqbpsqgqKqmqsqmrQqitGoKz0qngmCr64mr/pCJ
ALq6q5fZq6AZrKcJqsTKmsZ6rLaZrAq6rMzarEo6rYOZCtNap9aamNj6mAfArZvp
rZ4JrqYpruN6BeVqrugqnOqapuzaru76qvD6rPI6r4dZr4+5rfiqr/vKrzQ6rP9K
rslorrs5sEFasEJ6sBk6rSGKCvFaqw6LmBDrmBLLrRQboxbrUv46rgF7rB77sSC7
qCLLne5asqdwstWasg+7sgqAr7z6sgMQszIrCWOQsRpbmxzrozfLmznroDvLswkL
q9LKsCgrtCsrnw9gtN2KtEqLmo1go04LsBsbtec6tQhQtTp7tQmrsCbrqNEqtCTg
AV+rAGErthCAtEmr/rQAmrZqC7Vs67ZvC7cKcLXeObcLO6V/qbVCKwImSrR+a5mA
W7aS4C2EW7OkOrUqqriLy7hzS62mELmuKp4N67DzubItu6uAuwGCGwln0LRpmwG4
i4UZwLZtC7oSCp9oygDO2QDooqTTmaTaibwj8KpOGp7OG7lxKqVSOp5dm7LNWa/3
CgES2wB7GruzCwllgLZOm7u6y7sB4JtTq5zGub7Ci6aPyaYhoLwjMJ08+5fQCqL4
G62ni7rMC72G6Z7WO6HRiacIkAARcMAHrL0RwL2aua3eSgAVK6Mc8Km0G6C3S5ut
qKNPdwHmi74ejAAGcKi/2wDCmwAMwACTiZwM/tCcJTy/3Im8x5ukMUydgwCY9su8
hCmnWjsbLScGYTAGXuDDYTDERFzEYXAESJzESrzESrwCTuzEKbACXLeTwZcCwffE
z3LCWrzFXGzCW0wAYBzGYEwMYizGZXnGaNxTOhcFbNzGbvzGb6yivinHdFzHciwA
eJzHerzHeMyZASCCKCCCJ/AqgWwCgyyCiIzIQoDEi7zISezIR7DIRJDELRcGlewF
YfAFmgwGllzEqTIJRhzKonzETFzKS8wCT5zKK2DIqqzKXszFsAzLZTzLtAzGaXzL
ZwzHuqzLdtzLdszHwKzHnCkAJ5DIxnzMxhzJj7zMkCzJR0AEQNwFY2DJ/l8ABtZ8
zdfcyZ8MCaPczURsyuCMxKjcylBMzk8cy+jMxbW8zmGMy+5MA7scz23sy/SsosF8
zwEwAMSMzPyczMrMyEsMyZM8yV3gBUCMyV8wBth8zV+AyV6wzY/gzd4czuBszhad
yumc0QzAzuz8zrgsz/Jcz/R8zyQtAP180iIIzs2szIAEzQbt0GJQzQs9Bl8gBkEM
0Y4g0d1M0aZ80T6t0enM0evs0bcM0vEs0r5c0veM0ifdyMq80ko80C4tzUEcxAsN
BjVd1TjdCDo9yjxdyj590UCNzkJdy0Sdxka9y0jdy/qs1HzM1P3syJAM0ErszNB8
0C130AeNzZo8/gZbzQhdLcpfzcRhbdFjHctlTctnjcZpzctrXcdt7dZ6DNf8XNf/
/NRJPNBHMAbSLM2c3XKejc3R/NeLENihPNhLXNitnMWHrc6JXcaLncuNHcePTceS
/daUfcylPNeYPdBe4NmdPQaczdmi3dCkrQimbcSo3cSqncrB19qu/drtHNtkOdu0
Xdv2fNuTndvGLNeWzdua/dIup9cth815fdyJkNxFvNxJ3NytDN1fLN3TTd3wbN3z
jN3Zrd14zN3+vMhMoEqYDdADDdx6rddXDZbofQjq/c3sfQTurcrwrcXyPd/Ubd/3
jd/6ncf8nch1LQT/bdlIrNl6DdzAfdV+/p3ghrDgQ9zgDv7g5xzhGz3hBEDf0WTh
```

```
bIzf+a3fG67IjOzhHn7ZTu3bEHjVYIDiKa7iLO7iLw7djSnjM07jNh4FOH4AGb7f
Ow6Uj/zfAPThUf3MREDgwx3mRH4IYTEpKk7K7K3kTvzK0O3kNF7fFj7lVb7PO37Z
mO3jW/7Mz0zTBl3QnE3VBd3nBn3i2tANdrANeZDoiS4oZ57kag7jMT7hbx7lcl7l
V47ly5zleC7ieV3gfi7cVD2Pg6AHdlDqpL4NdqDoecDoSN7gar4CkO7mUG7jlZ7h
lx7JvN3IKz3g0Tziwv3boM7nhsANiW7oerDoc9Lorv7oMC7r9E3p/jg+57f+3Uis
BLrOBCLu59pO05rccn0t6gCA6seO6oeO6Kue7K2e5sx+ws75ysY5mZMZ32VcAEI9
6bQe7ZbO3YNczHTN0wOd0ECs1w096Hnd2YVw6seeB+V+6OXO6gvu6Eq+xfGuxY2J
wpQp72JM7xxt73GO77au74cM5ADt1JGs2Q3d0AIf8ARu0AeP6sVOHg6v3hDv3qjs
nCf8nCTMAMbJ7pSZ87MsDPU+6x2P4VVezEZvyIKc9MVsyCfwLOnH23UtjSE+yZ89
3N7O5wmd1wlN5uO+IkYODlUQ9mIv9rBZBVgw9lWAu2q/9mzf9gbw9gZwAHA/93QP
90Vw93if/vd6XwQt3fd+//d+PwVSMPiELwWCX/iI//WKDwthWgUE2vaQ7/Z1P/l1
v/eWj/dGYAR3PwSA3/ktjfigD/pTMAWLX/qrIKiq+ZqRv/pqT/mub/eXH/t37/m0
H/q2P/ijT/qmv/ulQLisz/qv//qyL/u07/m3b/uCr/u8v/yg4Pu/H/nB7/rDH/vF
3/nHf/vKz/zavwlmP77PD/nRT/nTf/nVD/jXf/vbn/6a4Pzfz/bhP/njb/nl//fn
b/vqf/+WwP7t3/rvT/fxv/eAQCQ4SFg4OCSVqLjIyAjwCBkpOUlZaXmJmam5ydnp
+QmKeTVKWmpqmpGqusrKavAKGysr/ltUa3uLa2tkZNhLOHTUKCwcWmx8jJysvKx5
6uzcGh19MFs9m4uN7bstOOytyBwuPk5e/vmMTiq9rmrtDpsdb8u9/f1tjp+vv1+c
ns7O7t07efLo+bLnjZ/ChQz5+UMHcJ1AdwTjGeyFcFjDjRw7Hnv4LKK0idYqZrto
KCMxjyxbupwEEprIViSrmdSGkpDKRi97+uQY89TMVRgw1Lx2E1dOnTsX/XwKNV9Q
VEOLFj1KK+mtpYOaOo0KNqyyqaUuaLiwDm0ralhfCdpKpAivIXLf2hrClYhXcGL7
+vVEdhQAs+zOrmX77gBiA3ZrvRUkRC6vW0fwLt2b6K/mzeKy/njW4vmzFtChP2eZ
gjq1atSLWEs5AftEChOwUaAwYRtFbNkmJo35DTz4b87Eiy8sPTr0aNKlPa9+PqV1
9NexTdA+YR239erWKQn/bjy8eHLNy5v3nAi6+tS7d9su0X637+/Cx9u/f+y8/tDr
+0+JD9t2AMImCX304Ydggprst59/6w0IYWwFGlifghZeCAmD+jmoXoQRRkIheBiO
mKCG53EInYcQghhicCS+eJ+J5qH4nIoDQtJihTDuWJyM5dG4mo0AApCjiDweqZmP
zQGpmpDxFWkkklKGpWRpTLLn5G5Q6jhll09Vyd+V/2UZ25YueommT2A6JyaZZZo5
XJpy/ra05mltunkCnHHOyedGdYo5ppt6jtFnoQz9eaegehrK6D5ragEonnku2mil
5ZSX3H6QJkrmoJZ+Kg4Xoo5Kaqml+oBqqqquuioDrr4Ka6yx6kBrrbbeaiuouipj
aq+9sgossLIOOyyuxuK6a7LG+MrsqME+myqx0r56bLW0KovtJ802Cy20005rbbXZ
jrvJtsx2+2wC3xIb7rHkvnuJub6iG+y67LaLLLz6SiLvr/Syam+x+N66b8GP9Gvq
vwAHPOvAuRq8L8KnKqwqww07fC3E+kpMKsUVW0wtxhlr/C7HznqMKsghY9wDyfCa
LCrKKavMAMs9tOwyuTBzIbMP/jTXLLIOOesMc88/By300NnubDTNSCu9dNEyHx00
1NgyPbXTVVudLNYoUy0y111L/bXWYYutq9ceg40x2mlz4UUWcHvma9xe8Jy1yj1s
7fanccudRdy+ZqFFzHmD/HTfMIrBeOOOP844JCaUQPnklF+OeeZdbM555557vkLo
oo9OOukWWHDBBaevzjrqrH/hRexfzO5FGLPTfjvtim8Gee+9Zw588JR/TjzxpR9/
/Oqqt8786rDXnnv0tucexu6a+Y5948Jvf3nx3nOOfPiiN0/+69Gfj3711vuVffbc
c//99+KLX3796N+fuxfrs9++7+9vHz/vzS989Ssf/g74/oX99aV//vsf8EwQwOIN
EHkFJB8C8adAsTDwdw4EXgSNN8HSVbB5FzxfGPSXQbBsEHId9OAHQRdC042wdSWM
XuxSqMIVOq6Fmnth52Iowxk6r4b5w2FUdLhDHnbPh+AD4viEOEQi6s6IT0Gi9pRI
udEwsQtOfCIULSDF3FGxilYUAxaHt0UudnEFXzxdGKc4xp6U0YxnTKMau9jGCrzx
C+qL40vmeMYS2HGNbPziBd54Qj/2xAvAYeQYGGg5JQ5yjW0E4x4V+ZLfMNKRXuhf
ICeJxzbi74T3QyEmWTKGL3zHk3XsghaYSMg84i92spPeKVsyBjA0kpGsxOLmCvfC
/i+EMAUpEF0KTleBCjAvmSQsZey6ID1T3pIjN9uBNa9Jq2tq85oKWIA3vwnOcIIz
COQspznPeU4oqHOd7GwnFKKwzgbIswEOcMA851nPezaAB9vs5zZ5wIOWwWGgcHiD
QedABzpMEx908OcOsunQHYhzohNFp0Ut6s6MZvSe+dQnPfXJz4j2E6A6IOhA5zAH
g6qUDm+Yw0KLQQc4zAEOcpBDHeBABzeIVKQU7ek3LwrUcmp0qPH0qFFButN/WtOk
BW2pSp36Bpa+lBM5jalVa4pTnCbVoT71aVCDSlSiHnWs8gzpVrX5BpOyIaUtZStb
ozpVTMyUDjWta13dYNWz/vazqz39KlDDOtSOkpWjetWmDtJK0JkylaktjWslappT
u9rVqnEorDb5SlG/XhSwGhXsYOXpgBxY9qGKHehaF2tSpzpWEnSVrGuxWtnRShSz
```

```
4tQsRjnrzs961AFm1asO3IDa4DZVoauFBE1f69qByna2tB2nbdOJ23bqVp8OkG1J
hYtalr6huI+Iw1WRK9nlNjecz4VudIs63XmKdrTAxS5qt8vdOGQVvOGV7XidW15y
nped6b1nb8/6W/dml7sAiClO6WtX8d63AQnIr1D3q87+gta67Z0DGwRMUAIPNLII
rqmCx6uABjs4CBCOsISra1lalba0AuZuh+WA1wTb/ve+3hwxiUvsWd0u1w1z4PGF
33vQqBJ3tS+OqYxHS+Maj7jEUPhof3fM45YGF6VPhe9q59DhA9f1w/e1MZMlLE8K
97jHwQ3yW108WdcaecszprGXSwzmBoxWxTzmcXDXGuQhr9aqOHWDXeMA6NbGdKf/
vWaSF/BmCIPWnketJ6Pl7M8eWBOg/qwzjKe8WALHFNBarimgAS2HqhKaq0lO9H7x
OVZHz9Ohkp60P8fshvYu9rQmxaQeMIHTT3O6pm+Q73FnSml/BnuvpV5ydOEJBQAs
mqwMqCcD5Hyzau5A2gAd9kPrjFIMDzSDeug2Hr7dbT18e9yRIEUVqnCFc6v7/txW
GIUV1i0BCUQgAvGut73vHW8X6Hvf/O53vxWSg4ALfOAEzwEPDC4BClAg4QtXuMMf
nvAJTADd6V63xS+e7v2Je9wc7/i3IUEKK4h85CQnObrRje+Up9zfLGc5wAsOc4JL
/OE0r7nDL47znGvc4zz/+CPSfYWSC33k5063yo9e75Yrfd8vj7nTJ2DzqD8851Rf
9/563nOQu3voXKfCFZCO9KUvvelOLzgPpI52ClR97VfHesfL7XWuy/3rYFe52JVO
9rLLPO1RX3vV2+52coNc7oS3AgQgUHd8373ledd7wHkQAb7b3O9UB3zg8VDuwstd
AhNI/L0X73J+ON7s/pKfPOVxbnm3S0Lzm/f850H/b9GPnuClr/npUb++y/scEqzn
uutfD3umy372Ao987W9+e3ULJveXh8nWey/y39s7+PxWyA6IH/AdHH/qybc68wMP
k7hDP/rSzzf19a0QgGJ/Bwzffve9b73mO3/8Iy+/+c/feMezf/sKf7+6Fah6lPB8
vTcK9icB56dvVpYP2Jd9/Nd//lcFGZR1lRB09Ed+5YeALqCA+MCAAdd+xweBERhX
FWiBBpiBG2gOHZgDH1h7IehYJEh/5TcBEXCCw0d82ueALjiCFniBiTcBEDCDNbgP
HYiD/KeDU1UFJOcMVOB79peB+ed4LFh6RxhH/oIHCUw4cl6nhSE3dCaIgFBYdvtn
hBB4a9/XbXXgbR4HclSAhV53BW74hkE3gPaGeBKAeHdoh3lYh084fAenf2Lofus2
ChZXcRdXhrvzVLEGB25QZSv1CDVwAzQgiTcgAzdQA5eIiZmIicnEiRVwAZ4Iip8o
isl0AWxoiqeIiqjIEGnAiq3oiq/YiiAgi7NIi7VYizaAi7hIA5ZYA7qYi7+Ii1Gl
a4CGUnyWa3EAMW9wYbHGjM2ojIvIjJCoidNIjTXQideIjdeYituYiqsIi9/oirYo
juIIjDdwA8CIjrloUMM4jHDAjgZjUM0oj4z4BvJYjfeYidmoj9rIjf1I/gXeCI7g
OI4DOYvlSAPpmI4sxY66NlDvuC/zCJHziI8TuY8VWQH+6I8KEZAbSZAEWY4ImY4L
KZLDWDARaZKxNpH4aJH7iJH9qJEbKZAdOY7AeJAgCYwjiZPIuC8FtYwn2YwpeY8r
qY9V0JLb+JIwCYsyOY41aZMImZMjWTDx6JP2CJTUKJTZWJRGyQ9I+Y1KKY5M2ZTo
+JQiWZJTKZFVOY1XiY1Z2Y1byZWv6JW2iANh6ZRj6ZD6YpZniZb5qJadyJaq6JZv
GYtxSYtgSZe5aJd3CS95SZV7uYl9yYl/eYpHKZhpQJi0eJghmZi6VpaMiZKO+ZiQ
eZGSyYaUKZiX/lmQmXmTm/lpnemZoBmakEmapRmYlYmasqiaq8maOomXnhlrNACb
1iiao0mapvmWtwkCufmLuwloBcOT0GiWwAmbw0mckmmcXImcyomY7eiOORmV9eiZ
0gma1Dmb/1ibp3mb2omL3NmdOOmaefkG4umY5Dmb14mU2XmYOACW7NiQC1lrD+mb
8Rmc9Fmc53mc6XmYNDCX28mQcyCS/9mb8KmM8rmXkGkBylSe9gmTqPkBH2CY6WiO
kRiiN+BrvpZVCEUHJZpQOFUwy+GiLwqjWgABD0CjNWqjN3qjDgYEQoBSPeqjPZpQ
CdWjC4EHdWCkdVCkR4qkSmqkeMAESACl/lEqpVOKBElgpfI2b1mqpVu6paSyBVsA
GmEqN6OiBV4wGocYbuHWJWnKppYQo2+6HFiAo3NKpw4mBEDwo3kKpCiqZ/rQpEya
pEdapElKpYVKpUnApYmqqIKjBV+6BXZjpq6kBV8AGpvDppeKqSSCqW06CXD6pqJC
p6FqozqKp3pqqkG6EEv6p4E6qExqqK8apYoqq1r6SqPhqGVKOHEjqWbqBV2wqb/a
bRgCrGnaqZ76pqKKrDZGjKZqqkQaB3jwrNAqrdH6rNXqpLD6qhLwANo6g7O6paNh
N6DhBY5KruWaB8OaqReCrmoaCcZ6rMgaqjZ2B3HArM2qEN9GrdKK/q/UWq1Piq2F
+gATUKPeuqWbQzhlGjflqrBbkAfnuq7saiEPG6yQ4K4xCq/xOmLeJaT16qMLUa3W
aq3j9qx3cAd4kAT/WqgzKnEPQLBaarDjurAxK7HEqq4SCwl6ULEwerF1OmJ3EKQc
27H3Gq34Oq2Ahq8kGwcom7ICG7Atm6WuZKYxK7MzO7ERS7XdlrMvurNzamNAa6/7
ALLTKq0kS7Yk669KG6sL0K1OO29S67ZecLWHqCBxi7NZuxxbi6Nd67V5KrQi67dx
ULZke7JoG6sSx7ZZ6rZSmwVxK6xxa7d3i7ejOmJ7y7f8ULRhiweBS7KEK6WHu6WJ
u7CBw7g1/nu1jzsakSu5Dka5P6oQISu2I6u5nBurnou4oFuucTO6Vlu6pnuxh3d4
qZtfqxu0+zCtTHqkmnsHsgultFu7tluuuTu3jsu7vQsBDgC85SW8Q8oP+bqkrYq8
yosEzNu2zvu8V9u4u/u4W/u7Naq32Xuv0iqogvq9yiu+EUC+5Uu150u1pqsF6mu9
```

```
7Du52etS20u08WukyDu4nFu/90uu0Bu9+zu9yOoA8zajAKy6Aty6+Kqq3Yu8Z0u4
C8zAX9qw+au/Esu/FxsBDrC+NNq+q8sQ8LvBBxy4SUu/4hvCX6oHDmuzJfywJwyv
Wnq9z4XBRArDfzoHCFzDzHvDW+DA/gkivekLrxM8wUGsWUDwCHvrEqyauWX7CAn8
wTZ8w02MIMhLxmXbBGfcBEzwBGjMxm3MBC8Ax3Esx3NMx2rwBnaMx3esx3qMxyTC
x3v8x3n8Bi5AAy5QAzTwAjKQyC6gyC8QAzJAyJAsAzLwBJVsyZeMyZhsB5vMyZ3s
yZ2MIXhgB2VcxmzsBEzQxqncBE7QyHTsynMcyLH8xyiIIHlsy7GsBnNQA4a8y5Mc
x75MyIwMx4qcycVczJ+MzJ+MIaRMxqPMxkzgBE6gym3cyq9szbd8y7E8IrjMzWrg
AjFQAzHAyIzsy5NcyDQAzJRszOtcycnszpu8zMysuZvMxk+A/srTzMYuYM3WLM7d
fMfYTMv2Ycf+fMv7BpzDrM8vQM4yUMiJTMzsvM7v7M7xLM+ai8+qrATRvM8bDdCC
jM0Xwscd/cdwAM6GfM6TzNCFHMmRDMcQHdESjcwUXdFmfNFnLM2rHM3VvNGwLNJq
ANAWMss9rcc0EAMlzdDBTM6MXMiS/AIubcwwHdMXMtMWXdMZHc1OkNEJvdN17M+4
bCG2LNQD/c3hPM4nTciSiNIO7dTHDNWeLNNTfQc1jdNKYNVOsNWvTNA/rSBBTdBw
sMuEfMjkvNTorG/orNDqvNaX3NZuLdVwTbanLNc2Lc06fdcvoIxswAZhrcdArdl5
DAe8/lzUv6zI4vzLKo3Yid3Oi83Jbz3V0RzZTbAE0qzVlQ3HmI3ZBM3HQI3btkwH
32zIwgzJiTzMDm3ONIDaiq3a8NzYjn0HNx3ZV03bc2zbmb3bAS0e1W3LkhjY5XzY
kgzJSX3clpzcym0hzE2yr33GSZDT0R3H043dX43derxvRi3cvizcDl3I4Z3ayc3a
M43eaOwEs03b093Z1h0e8W3HvX2JRQ3J6Gzaivzd6HzaqD3edtDfFf3fNh0D7F3b
tv3eex3faUXUv73Ugo3IxH3O+v0EFX7hpEzPGd4EG87hBF7dnB3igG3WDd7g+jzJ
waziLL7ccG0HMN4EAl7Z7q3Z/gZuHAWOx36tb0YtiUrNyDK+1BOe2EBuIa8LrR8r
ttKK3lit0Rz+AjQu0rmt24Hc0yat5oGt0m1e2C7w4+ONIdbavfqKuXHw5VdN2XeN
5N2cxyDt03wN0Ofc5ud8idotndod5/x9IfA7qEObr1uOB5J90WCe1WJux5gt1Ht8
IQMN1rKsUDFwzr+t5sHM5oUd53dQB+9Msqt9IUgatiHL5c8azUuwBJVe15Ud3HEM
CWwAyNjcxyOSzQNt5gDw20Ud5YaO1Mqe6nQg0c5O3grCnHEgCEMgBNeO7dmu7UIA
Bt3u7d/u7WMQO+K+J/pCBueO7umu7mRwBmXg7ktRBJax/qKtVVNYpmbNeSHTPgRB
MATWvu3/zu3gLvBg0AW/0QW0ZDDrrvDo3u7vnhPAIAg4tWb0NVB1MOfM2e/7DvD/
PvACT0ubJE3wsvAjTwZlQAZ5EfEb1mEGdvG7ue/9vvHb3vHgPgYFr0k3tC8kT/Io
TwQbdlzgVVUWn+/MGfMbP/M1b/AGfzuEYu46P/JLcQRCEPUvJllw0PKsWfQAP/Mf
z/WplEBN7/QKvxRSL/VUb1cDbCHTnvUc3/E2X/Dj/u05H/ZinxNRf+1mb1dXv5lr
L/MdL+6M5PZxD/Zzn+5jb/d4X1d6n5h8r+1bX/PjHjuCL/KEr+5LoQRCwARCgPc4
/iX0aU/0jI/tjv/3Ni/570L5lZ8T2Y73KWr1Q7+boB/6fr85Sd8FpU8up1/4qY/5
mm/2WaX4dgn71771IP/3tj8uuI/uY5/5vE/1OdX50v75sI/0vSrub2/9069Jco/8
R1D3Un8EiF9Tvz+Wwc/tf69J5k/uqnTwCD/4p2/4ZY/4aA/9rx/8ugT4B0/us59K
4w47TD/5yA8IRIKDhIVER0dCQnKMjY6PcnB0cACVlpeYmZqbnJ2en5ZxoqOkpaWK
qKmqqmBdY65jXq5esV5eX2O5tl6gvb6/wMHCmmTFxsfIyYbLg4mIkHJ00NGSw9bX
wabao29vdHFDQ0FDq+WKvWBjX7e4te27X/Bg2PP09dbJ+PnMzELOjnOS3DSaM2mS
JDn2Etbbto1OtzfjhhAxVw6MrXTq1L2KxfHWGDDyFIocmTCfyWP7ljlb1AgOQGmM
XsIJOIekTWAMTTmUFI5IEIqrLKbL1YXWxVtGcYG8ybSpr5MnUxrqhwhAnpgu5zCy
Q9BNQadgw4odS7as2bNo06pdy7at27dw48qdS7eu3bt48+rdy7ev37+AAwseTLiw
4cOIEytezLixY2GBAAA7"""

def all():
    import keyboard                                          #MIT
    keyboard.release('left ctrl')
    keyboard.release('right ctrl')
    keyboard.release('1')
    keyboard.release('2')
    keyboard.release('3')
    keyboard.release('4')
    keyboard.release('5')
```

```
    keyboard.release('6')
    keyboard.release('7')
    keyboard.release('8')
    keyboard.release('9')
    keyboard.release('a')
    keyboard.release('b')
    keyboard.release('c')
    keyboard.release('d')
    keyboard.release('e')
    keyboard.release('f')
    keyboard.release('left shift')
    keyboard.release('right shift')

#タスクトレイにてアイコン表示
def ICON():
    def quit_app():
        icon.stop()
        closed()

    #アクティブ化
    def run_app():
        akuthibu = win32gui.FindWindow(None, 'DisShot')
        time.sleep(1)
        win32gui.SetForegroundWindow(akuthibu)

    image = Image.open('E.ico')
    menu = Menu(MenuItem('表示', run_app), MenuItem('終了', quit_app))
    icon = Icon(name='test', icon=image, title='DisShot', menu=menu)
    icon.run()

#create GUI
def make_display():
    global dir_path

    def get_photo_image4icon():
        return tk.PhotoImage(data=icon_data)  # PhotoImageオブジェクトの作成

    #print("moving data")
    root = tk.Tk()
    root.title('DisShot')
    root.geometry('250x150')
    root.attributes("-toolwindow",1)
    root.protocol("WM_DELETE_WINDOW",closed)
    root.resizable(0,0)

    photo = get_photo_image4icon()
    root.iconphoto(False, photo)

    #object definition
```

```
def dirdialog_clicked():
    global dir_path
    current_dir = os.path.abspath(os.path.dirname(__file__))
    dir_path = filedialog.askdirectory(initialdir=current_dir)
    entry_ws.set(dir_path)

def display_research():
    global dis_count, temp_windowx_cd, temp_windowy_cd,windowx_cd, windowy_cc
    #以下、再定義
    dis_count    = 0                                        #ディスプレイの枚数
    temp_windowx_cd   = [0]        #仮のディスプレイの左上の座標 X
    temp_windowy_cd   = [0]        #仮のディスプレイの左上の座標 Y
    windowx_cd        = [0]        #最終的に使う数
    windowy_cd        = [0]        #最終的に使う数
    windowx_size = [0]        #ディスプレイの解像度(サイズ)
    windowy_size = [0]        #ディスプレイの解像度(サイズ)
    min_x_cd = 10000000  #初期設定  数は適当
    min_y_cd = 10000000  #初期設定  数は適当
    #numbur of display
    for d in get_monitors():
        dis_count = dis_count + 1

    #Add display information to the array
    Range = range(0, dis_count)

    for data in Range:
        monitor = get_monitors()[data]
        temp_windowx_cd[data] = monitor.x
        temp_windowy_cd[data] = monitor.y
        windowx_size[data] = monitor.width
        windowy_size[data] = monitor.height

    for i in Range:                            #座標比較
        if temp_windowx_cd[i] < min_x_cd:
            min_x_cd = temp_windowx_cd[i]

        if temp_windowy_cd[i] < min_y_cd:
            min_y_cd = temp_windowy_cd[i]

    for j in Range:                            #切り抜きに使う数値へ変更
        windowx_cd[j] = -1 * min_x_cd + temp_windowx_cd[j]
        windowy_cd[j] = -1 * min_y_cd + temp_windowy_cd[j]

lebel_1 = tk.Label(root, text='画像保存先を指定')
entry_ws = tk.StringVar()
dir_entry = tk.Entry(root, textvariable=entry_ws, width=20)

dir_button = tk.Button(root, text="参照", command=dirdialog_clicked)
```

```
    research_button = tk.Button(root, text="ディスプレイ再カウント", command=displa

    lebel_1.pack()
    #↓　これによって初めからエントリーダイアログに入力されている形
    dir_entry.insert(tk.END, dir_path)
    dir_entry.pack()
    dir_button.pack()
    research_button.pack()
    root.mainloop()

#program termination constant
def closed():
    os.kill(os.getpid(), 9)

#save screenshot and name it
def save_image():
    global dir_path, filename
    screenshot = ImageGrab.grab(all_screens=True)
    #ファイルネーム指定
    now = datetime.datetime.now()
    filename = 'disShot' + now.strftime('%Y%m%d_%H%M%S') + '.png'

    screenshot.save(dir_path +'\\'+ filename, quaality = 100)

#take screenshot and save photo
def screen_shot_1():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 1:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[0] : windowy_size[0] + windowy_cd[0], windowx_cd[0]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    #gc.collect()
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_2():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 2:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[1] : windowy_size[1] + windowy_cd[1], windowx_cd[1]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_3():
```

```
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 3:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[2] : windowy_size[2] + windowy_cd[2], windowx_cd[2]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_4():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 4:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[3] : windowy_size[3] + windowy_cd[3], windowx_cd[3]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_5():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 5:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[4] : windowy_size[4] + windowy_cd[4], windowx_cd[4]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_6():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 6:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[5] : windowy_size[5] + windowy_cd[5], windowx_cd[5]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_7():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 7:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[6] : windowy_size[6] + windowy_cd[6], windowx_cd[6]
```

```
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_8():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 8:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[7] : windowy_size[7] + windowy_cd[7], windowx_cd[7]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_9():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 9:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[8] : windowy_size[8] + windowy_cd[8], windowx_cd[8]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_a():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 10:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[9] : windowy_size[9] + windowy_cd[9], windowx_cd[9]
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_b():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 11:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[10] : windowy_size[10] + windowy_cd[10], windowx_cd
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)
```

```
def screen_shot_c():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 12:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[11] : windowy_size[11] + windowy_cd[11], windowx_cd
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_d():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 13:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[12] : windowy_size[12] + windowy_cd[12], windowx_cd
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_e():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 14:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[13] : windowy_size[13] + windowy_cd[13], windowx_cd
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)

def screen_shot_f():
    global user, dir_path, dis_count, filename, windowx_cd, windowy_cd, windowx_s
    if dis_count >= 15:
        save_image()
        img = cv2.imread('{}\\{}'.format(dir_path, filename))
        img1 = img[windowy_cd[14] : windowy_size[14] + windowy_cd[14], windowx_cd
        cv2.imwrite('{}\\{}'.format(dir_path, filename), img1)
    all()
    gc.collect()
    time.sleep(0.05)
#ended

'''
def Print():
    global windowx_cd, windowy_cd, windowx_size, windowy_size, dis_count
    while True:
```

```
        print(windowx_cd)
        print(windowy_cd)
        print(windowx_size)
        print(windowy_size)
        print('count : ', dis_count)
        time.sleep(3)
'''


def keyboardShotcut():
    with keyboard.GlobalHotKeys({
            '<ctrl>+<shift>+1': screen_shot_1,
            '<ctrl>+<shift>+2': screen_shot_2,
            '<ctrl>+<shift>+3': screen_shot_3,
            '<ctrl>+<shift>+4': screen_shot_4,
            '<ctrl>+<shift>+5': screen_shot_5,
            '<ctrl>+<shift>+6': screen_shot_6,
            '<ctrl>+<shift>+7': screen_shot_7,
            '<ctrl>+<shift>+8': screen_shot_8,
            '<ctrl>+<shift>+9': screen_shot_9,
            '<ctrl>+<shift>+a': screen_shot_a,
            '<ctrl>+<shift>+b': screen_shot_b,
            '<ctrl>+<shift>+c': screen_shot_c,
            '<ctrl>+<shift>+d': screen_shot_d,
            '<ctrl>+<shift>+e': screen_shot_e,
            '<ctrl>+<shift>+f': screen_shot_f}) as h:
        h.join()


if __name__ == "__main__":

    with ThreadPoolExecutor(max_workers=3) as executor:
        executor.submit(keyboardShotcut)
        executor.submit(make_display)
        executor.submit(ICON)
        #executor.submit(Print)
```

使い方

ダウンロード先

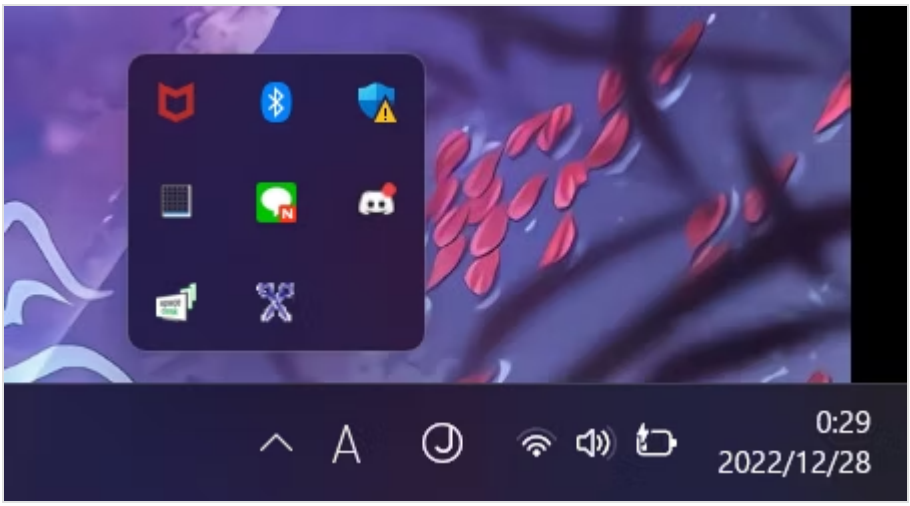
ダウンロードは以下からできます。
https://github.com/juannu1028/DisShot/tree/test

DisShot.zipをインストールしてください。

起動したら

以下のようなものが表示されます。

参照ボタンまたは入力欄に保存先を選択、入力してください。
デフォルトはc:\User\ユーザー名\Pictures
(場合によってはこのpathでは使えません)
ソフトを立ち上げた状態でディスプレイの配置を変えた、新しく接続した場合は**ディスプレイ再カウント**を押してください。



また、タスクトレイにアイコンが出ます(ハサミのマーク)。右クリックで**表示**と**終了**が出ます。
これでウィンドウをアクティブ化したり、タスクトレイからも終了したりできます。

コマンド(ショートカットキー)

スクショはすべてショートカットキーで行います。
ディスプレイ番号はwindoswの設定から各自確認してください。

ディスプレイの番号は16進数でカウントしてください
ディスプレイの最大認識数は15枚となっています。
キーは以下の通りです。

```
ctrl + shift + ディスプレイ番号
```

つまり、5枚目のディスプレイを撮りたいのであれば、

```
ctrl + shift + 5
```

また、14枚目のでぃすれぷいを撮りたいのであれば、

```
ctrl + shift + e
```

という形になっています。
大量のディスプレイを準備できないので私の環境では4枚まで動作していることを確認しています。

改善したい点

①一部ショートカットキーがかぶってしまっていたため、別のパターンを考えたい。
②起動が遅い。exe化にNuitkaを使ったのが原因？
③画像保存pathに日本語が混ざっているとエラーが起きているので直したい。
④画像保存pathを参照ボタンを押して選択中にキャンセルボタン、excapeボタンを押すとpathが空白になってしまう問題を直したい。
⑤classを使えるようにしたい。ソースコードで似たような処理をしているところがあるので、もっと見やすく少ないコードにできるようにしたい。

最後に

読んでくださり有難うございました。このソフトはwindowsでしか使えないので今度はlinuxでも使えるようにしたいと考えています。macOSは触ったことも見たこともないので知りませんが()。

また、今回簡単なソフトウェアとはいえ作ってみると、失敗ばっかり続いていたので改善点が多いですが、ここまで来れてわからないことがまだ多すぎると感じています。しかし、楽しさや満足感が気持ちいいのでしばらくソフトウェア勉強は続けられそうです。ハードウェアの知識も欲しいですね....

@greghur (@ greghur)

プログラミングに興味を持って大学から勉強中。

@aokikenichi (青木 健一)
2024年08月25日

2024年版機械学習・データ分析の必須10冊+ガチ90冊+Next5冊=105冊

本 機械学習 データ分析 データサイエンス

♡ 182

@TabataRin in 株式会社Nuco
2024年08月26日

アンチパターンで学ぶ「仕事の進め方」

ポエム 仕事のやり方

♡ 105

@ho_na in 株式会社PRUM
2024年08月25日

JavaScript｜importの書き方が多いのでまとめてみた

JavaScript プログラミング 初心者 フロントエンド エンジニア

♡ 104

@ochx
2024年08月26日

メモ帳でコーディングしていた話

HTML CSS JavaScript ポエム メモ帳

♡ 37

トレンド一覧を見る

コメント

この記事にコメントはありません。

いいね以上の気持ちはコメントで

ログイン 新規登録

記事投稿キャンペーン開催中

paiza×Qiita記事投稿キャンペーン「プログラミング問題をやってみて書いたコードを投稿しよう！」

2024/07/19~2024/08/29

詳細を見る

すべて見る

Qiita

How developers code is here.