Федеральное государственное бюджетное образовательное учреждение высшего образования «Университет «Дубна»

Кафедра ИСАУ

Дисциплина “Технологии программирования”

**Текст программы**

утилита для навигации по системе windows

**BindedCorners2024**

**А.В.00001-01 01**

|  |  |
| --- | --- |
| **Рабочая группа** | **Должность** |
| Батаев И. С. | Архитектор |
| Программист |
| Технический писатель |

**Преподаватель**

|  |
| --- |
| Мельникова О. И. |

Дубна 2024

# Аннотация

Данный документ содержит описание текста вспомогательной компьютерной навигационной программы для *Windows*.

Исходным языком разрабатываемой программы является *Python*, средой разработки – *Visual* *Studio* *Code*.

Оформление программного документа “Текст программы” выполнено в соответствии с требованиями ЕСПД (ГОСТ 19.105-78(СТ СЭВ 2088-80)).

Оглавление

[Текст программы BindedCorners2024 3](#_Toc160700688)

[Файл **json\_pack.py**: 3](#_Toc160700689)

[Файл **get\_wallpaper.py**: 3](#_Toc160700690)

[Файл **navigator.py**: 3](#_Toc160700691)

[Файл **main.py**: 5](#_Toc160700692)

# Текст программы BindedCorners2024

## Файл **json\_pack.py**:

from json import dump, load

def save\_configure(data, fname = "data"):

    ''' Сохранение конфигурации в JSON файл. '''

    with open(f"{fname}.json", "w", encoding="utf-8") as file:

        dump(data, file)

        return True

def get\_configure(fname = 'data'):

    ''' Получение конфигурации из JSON файла. '''

    try:

        with open(f"{fname}.json", "r", encoding="utf-8") as file:

            return load(file)

    except:

        return False

## Файл **get\_wallpaper.py**:

import ctypes

SPI\_GETDESKWALLPAPER = 0x0073 # Код для получения пути для обоев.

MAX\_PATH = 260 # Максимальное количество пути для обоев.

wallpaper\_path = ctypes.create\_unicode\_buffer(MAX\_PATH) # Создание буфера для пути для обоев.

# Вызов функции SystemParametersInfoW для получения пути обоев

ctypes.windll.user32.SystemParametersInfoW(SPI\_GETDESKWALLPAPER, MAX\_PATH, wallpaper\_path, 0)

wallpaper\_path = str(wallpaper\_path.value) # Преобразование буфера в строку.

## Файл **navigator.py**:

import pyautogui as ptg

from time import sleep

# Отключение экстренного выхода при наведении курсора мыши в левый верхний угол

ptg.FAILSAFE = False

running = False # Переменная, отображающая состояние модуля

Prompts = [] # Список комбинаций клавиш в формате, который принимает функция hotkey

# Отслеживание курсора мыши в разных углах

def is\_bottom\_left():

    return ptg.position()[0] == 0 and ptg.position()[1] == (ptg.size()[1] -1)

def is\_bottom\_right():

    return ptg.position()[0] == (ptg.size()[0]-1) and (ptg.position()[1] == ptg.size()[1]-1)

def is\_top\_left():

    return ptg.position()[0] == 0 and ptg.position()[1] == 0

def is\_top\_right():

    return ptg.position()[0] == (ptg.size()[0]-1) and ptg.position()[1] == 0

def text\_to\_prompt(text: str):

    ''' Преобразование текста в список комбинаций в необходимом формате. '''

    promt = text.split('+')

    promt.extend([''] \* (4 - len(promt)))

    return promt

def navigate(promt):

    ''' Иммитация клавиш с обработкой ошибок'''

    try:

        ptg.hotkey(promt)

        return True

    except Exception as e:

        print(e)

        return False

def switch\_pause(value: bool = not running):

    ''' Переключение состояния модуля. '''

    global running

    running = value

def stop():

    ''' Остановка модуля. '''

    switch\_pause(False)

def get\_index():

    ''' Получение индекса угла. '''

    if is\_top\_left(): return 0

    if is\_top\_right(): return 1

    if is\_bottom\_left(): return 2

    if is\_bottom\_right(): return 3

    return -1

def run():

    ''' Запуск модуля. '''

    switch\_pause(True)

    # Прошлый индекс, нужен для избежания повторного срабатывания

    prev\_index = -1

    while running:

        ''' Главный цикл, который отслуживает мышь и осуществляет иммитацию нажатия клавиш. '''

        index = get\_index()

        if prev\_index != index:

            prev\_index = index

            if (index > -1 and index < 4):

                navigate(Prompts[index])

        sleep(0.01)

## Файл **main.py**:

from tkinter import StringVar, Canvas,Tk

from tkinter.ttk import Style, Frame, Combobox, Button

from PIL import ImageTk, Image

from threading import Thread

from pystray import Icon, Menu, MenuItem

from os import \_exit

# Подгрузка своих модулей

import get\_wallpaper as gw

import json\_pack as jp

import navigator as nav

# Поток для работы модуля навигации

nav\_thread = None

# DATA

# Создаем изображение для значка

icon\_image = Image.open("./icon.ico")

# Иннициализация окна

w\_p = (10,10)

size = (600,430)

root = Tk()

root.geometry(f'{size[0]}x{size[1]}')  # Установите размер окна

root.resizable(False, False)

root.title('BindedCorners')

root.iconbitmap('./icon.ico')

# Установка стилей окна

Style().configure(".",  font=("Segoe UI", 12), foreground="#262626", background="#bdb4bf", relief='solid border')

Style().configure("TButton",  font=("Segoe UI", 12), foreground="#262626", padding=-3)

Style().configure("TCombobox",  font=("Segoe UI", 12), foreground="#262626", padding=3)

# Получение сохранненой ранее конфигурации программы

corners\_prompts = jp.get\_configure()

# Объявление вспомогательных переменных

corner\_comboxs = [None,None,None,None]

bind\_labels = ('win+tab', 'win+ctrl+right','win+ctrl+left','win+a','win+r')

corners\_name = ('nw','ne','sw','se')

labels = [None,None,None,None]

corner\_vars = [StringVar() for \_ in range(4)]

if corners\_prompts:

    '''Конвертирование сохраненных комбинаций в текстовый формат. '''

    corners\_prompts = corners\_prompts['prompts']

    for i in range(4):

        corner\_vars[i].set('+'.join(filter(lambda x: x != '', corners\_prompts[i])))

# Инициализация интерфейса

window\_corners = Frame(root)

window\_corners.pack(fill='both', expand=True)

image\_size = [0,0]

image\_size[0] = (size[0] - 4\*(w\_p[0]))

image\_size[1] = image\_size[0]\*9//16

image = ImageTk.PhotoImage(Image.open(gw.wallpaper\_path).resize(image\_size))

canv = Canvas(window\_corners, width=image\_size[0]+w\_p[0], height=image\_size[1]+w\_p[1], bg='black')

canv.place(x=size[0]//2, y=size[1]//2, anchor='center')

canv.create\_image(w\_p[0]\*0.7,w\_p[1]\*0.7, image=image, anchor = "nw")

for i in range(len(corner\_vars)):

    corner\_comboxs[i] = Combobox(window\_corners,textvariable=corner\_vars[i])

    corner\_comboxs[i]['values'] = bind\_labels

    corner\_comboxs[i].place(x=i%2\*(size[0]) + w\_p[0]\*(1-i%2\*2), y=w\_p[1]\*(1-(i//2)\*2) + (size[1])\*(i//2), anchor = corners\_name[i])

# Functions

def on\_close():

    ''' Скрытие окна. '''

    root.withdraw()

def nav\_start():

    ''' Запуск модуля навигации. '''

    nav\_thread = Thread(target=nav.run)

    nav.Prompts = corners\_prompts

    nav\_thread.start()

def func\_button():

    ''' Обработка нажатия кнопки "RUN" и "STOP".'''

    global corners\_prompts

    corners\_prompts = []

    for var\_c in corner\_vars:

        corners\_prompts.append(nav.text\_to\_prompt(var\_c.get()))

    jp.save\_configure({'prompts':corners\_prompts})

    if nav.running:

        main\_button['text'] = 'RUN'

        nav.stop()

        return

    nav\_start()

    main\_button['text'] = 'STOP'

# Создание главной кнопки

main\_button = Button(window\_corners, text="RUN", command=func\_button)

main\_button.place(x=size[0]//2, y=size[1] - w\_p[1], anchor='s')

# Переназначение функции кнопки "крестик"

root.protocol("WM\_DELETE\_WINDOW", on\_close)

# Создание иконки на панели задач Windows

icon = Icon("BindedCorners", icon\_image, "BindedCorners",  menu=Menu(MenuItem("Open", root.deiconify), MenuItem("Close", root.destroy)))

icon\_thread = Thread(target=icon.run)

icon\_thread.start()

# Запуск окна

if \_\_name\_\_ == '\_\_main\_\_':

    root.mainloop()

    icon.stop()

    nav.stop()

    \_exit(0)