from re import sub

import PySide2

from PySide2.QtCore import \*

from PySide2.QtGui import \*

from PySide2.QtWidgets import \*

import pyautogui

import webbrowser

from datetime import datetime

import sys

from email.mime import message

from typing\_extensions import Self

import paho.mqtt.client as mqtt

import time

import socket

from pyowm.owm import OWM

from pyowm.utils.config import get\_default\_config

import threading

import psutil

from ui\_client2 import Ui\_MainWindow

Главный класс

class MainWindow(QMainWindow):

Инициализация конструктора

    def \_\_init\_\_(self):

        QMainWindow.\_\_init\_\_(self)

Объект класса ui main window

        self.ui = Ui\_MainWindow()

        self.ui.setupUi(self)

Обращение к элементу для закрытия приложения

        self.ui.pushButton\_close.clicked.connect(self.close)

        self.ui.pushButton\_ghost\_w.clicked.connect(lambda: self.showMinimized())

        self.ui.lineEdit\_for\_writetext2.setPlaceholderText('Ввести текст для mqtt/text/chat')

        self.ui.textEdit\_for\_view2.setPlaceholderText( 'Здесь отображаются сообщения ' + '\n'

      ' \_\_    \_\_     \_\_\_\_\_\_     \_\_\_\_\_\_   \_     \n'

      '/\ -./  \   /\  \_\_ \   /\\_\_  \_\ /\\_\_  \_\  \n'

      '\ \ \-./\ \  \ \ \/\\_\  \/\_/\ \/ \/\_/\ \/  \n'

      ' \ \\_\ \ \\_\  \ \\_\_\_\\_\    \ \\_\    \ \\_\  \n'

      '  \/\_/  \/\_/   \/\_\_\_/\_/     \/\_/     \/\_/  \n'

        + '\n')

Обращение к кнопкам для перехода по страницам

        self.ui.pushButton\_page1.clicked.connect(lambda: self.ui.stackedWidget.setCurrentWidget(self.ui.page))

        self.ui.pushButton\_page2.clicked.connect(lambda: self.ui.stackedWidget.setCurrentWidget(self.ui.page\_2))

        Обращение к брокеру mosquitto.org

self.client = mqtt.Client()

        self.client.connect("test.mosquitto.org")

        self.start\_take()

        self.ui.btn\_for\_sendmessage2.clicked.connect(self.start\_send)

        self.ui.textEdit\_for\_view2.setReadOnly(True)

        self.setWindowFlags(Qt.FramelessWindowHint)

        self.setAttribute(Qt.WA\_TranslucentBackground)

        self.ui.\_old\_pos = None

        self.center()

        self.hdd = psutil.disk\_usage('/')

Список подписок

        list\_for\_publish = [

            'mqtt/text/chat',

            'device/memorystatus/harddrive/c',

            'device/work/cpu',

            'device/work/ram',

            'music/track1/start',

            'music/track1/stop',

            'android/get\_ascii',

            'android/get\_ip',

            'android/tts'

        ]

        self.ui.comboBox\_for\_select\_topic2.addItems(list\_for\_publish)

Функции, отключающие панель навигации

    def center(self):

        qr = self.frameGeometry()

        cp = QDesktopWidget().availableGeometry().center()

        qr.moveCenter(cp)

        self.move(qr.topLeft())

    def mousePressEvent(self, event):

        self.oldPos = event.globalPos()

    def mouseMoveEvent(self, event):

        try:

            delta = QPoint(event.globalPos() - self.oldPos)

            self.move(self.x() + delta.x(), self.y() + delta.y())

            self.oldPos = event.globalPos()

        except AttributeError:

            pass

Создание нового процесса, который отправляет сообщения

def start\_send(self):

        threading.Thread(target=self.public(self.client),daemon=True).start()

    def public(self,client):

        now = datetime.now()

        cur\_time = now.strftime("%H:%M:%S")

        text = self.ui.lineEdit\_for\_writetext2.text()

        subtop = self.ui.comboBox\_for\_select\_topic2.currentText()

Отображение информации на клиенте об отправляемой реакции

        if subtop == 'device/memorystatus/harddrive/c':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<'+ str(self.hdd.free / (2\*\*30)) + '> ' + '\n')

        if subtop == 'device/work/cpu':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<'+ str(psutil.cpu\_percent(interval=1)) + '> ' + '%' + '\n')

        if subtop == 'mqtt/text/chat':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<mqtt/text/chat>  '+ text + '\n')

        if subtop == 'device/work/ram':

            values = psutil.virtual\_memory().percent

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<'+str(values) + '> '+ '%' + '\n')

        if subtop == 'music/track1/start':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<команда старт для Track1> ' + '\n')

        if subtop == 'music/track1/stop':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<команда стоп для Track1> ' + '\n')

        if subtop == 'android/get\_ascii':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<запрос на ascii> ' + '\n')

        if subtop == 'android/get\_ip':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<запрос на ip> ' + '\n')

Методы, отвечающие за принятие сообщения с другого клиента

        if subtop == 'android/tts':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + '<запрос на речь> ' + '\n')

        self.client.publish(subtop,text)

        self.ui.lineEdit\_for\_writetext2.clear()

Новый процесс, который принимает сообщения

    def start\_take(self):

        threading.Thread(target=self.take\_message(self.client),daemon=True).start()

    def take\_picture(self):

        msg = QMessageBox()

        msg.setWindowTitle('Кролик')

        msg.setIconPixmap('krolik.jpg')

        msg.exec()

    def weather(self):

        config\_dict = get\_default\_config()

        config\_dict['language'] = 'ru'

        owm =  OWM('472d111f15c9d5266faa15de9a0bc03c', config\_dict)

        place = 'Киров'

        mgr = owm.weather\_manager()

        observation = mgr.weather\_at\_place(place)

        weather = observation.weather

        temp = weather.temperature('celsius')['temp']

        status = weather.detailed\_status

        return temp,status

Если пришло сообщение, вызываем метод на клиенте

    def on\_message(self,client,userdata,message):

        now = datetime.now()

        cur\_time = now.strftime("%H:%M:%S")

        if message.topic == 'mqtt/example1':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + message.topic + ' ' + message.payload.decode('utf-8') + '\n')

        if message.topic == 'device/ip':

            h\_name = socket.gethostname()

            IP\_addres = socket.gethostbyname(h\_name)

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + message.topic + ' ' + IP\_addres + '\n')

        if message.topic == 'mqtt/picture':

            self.take\_picture()

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + message.topic + '\n')

Обработка принятого сообщения

        if message.topic == 'mqtt/get\_weather/temp':

            temp,status = self.weather()

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] '+ message.topic + ' ' + str(temp) + '°' + '\n')

        if message.topic == 'mqtt/get\_weather/status':

            temp,status = self.weather()

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + message.topic + ' ' + str(status) + '\n')

        if message.topic == 'mqtt/browser/client\_2/open':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + message.topic + ' ' + '<Браузер>' + '\n')

            webbrowser.open\_new\_tab('https://www.vyatsu.ru')

        if message.topic == 'mqtt/pc/client\_2/get\_screen':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + message.topic + ' ' + '<Скриншот>' + '\n')

            image = pyautogui.screenshot()

            image.save('screen.png')

        if message.topic == 'android/get\_ip/return':

            self.ui.textEdit\_for\_view2.insertPlainText('['+ cur\_time + '] ' + message.topic + ' ' + message.payload.decode('utf-8') + '\n')

Метод подписки на сообщения

    def take\_message(self,client):

        # client.connect("localhost")

        self.client.subscribe('mqtt/example1')

        self.client.subscribe('device/ip')

        self.client.subscribe('mqtt/picture')

        self.client.subscribe('android/get\_ip/return')

        self.client.subscribe('mqtt/get\_weather/temp')

        self.client.subscribe('mqtt/get\_weather/status')

        self.client.subscribe('mqtt/browser/client\_2/open')

        self.client.subscribe('mqtt/pc/client\_2/get\_screen')

        self.client.loop\_start()

        client.on\_message = self.on\_message

Конструкция, которая даёт возможность запуска файла

if \_\_name\_\_ == "\_\_main\_\_":

    app = QApplication(sys.argv)

Создание объекта MainWindow и вызов метода Show

window = MainWindow()

    window.show()

    sys.exit(app.exec\_())