VST LIST　Maintenance（VST 管理アプリ）ソースコード　Python

```
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
from tkinter import filedialog
from PIL import Image, ImageTk

import os
import sys
import shutil
import ctypes #管理者権限用
import webbrowser
import tkinter.font as tkFont
import pandas as pd
from datetime import datetime

#管理者権限用チェック
if os.name == "nt":  # Windows の場合
    documents_path = os.path.expanduser("~\\Documents")
    sleep_folder = os.path.join(documents_path, "VstSleep")
else:  # macOS や Linux の場合
    documents_path = os.path.expanduser("~/Documents")
    sleep_folder = os.path.join(documents_path, "VstSleep")

def is_admin():
    try:
        return os.getuid() == 0
    except AttributeError:
        return ctypes.windll.shell32.IsUserAnAdmin() != 0

def get_files_info(root_folder):
    folders = [root_folder]
    file_list = []
    for folder in folders:
        for dirpath, dirnames, filenames in os.walk(folder):
            for file in filenames:
                file_path = os.path.join(dirpath, file)
                file_size = os.path.getsize(file_path) / 1024 / 1024
                # 小数点以下を含まないファイルサイズを数値として取得
                file_size_num = round(file_size, 2)
                access_time = datetime.fromtimestamp(os.path.getatime(file_path))
                access_time = access_time.strftime("%Y/%m/%d %H:%M")
                folder_name = os.path.dirname(file_path)
                file_list.append([str(file), file_size_num, access_time, folder_name, file_path])
    return file_list
```

```
#アプリケーションクラス
class Application(tk.Frame):
    def __init__(self, master=None):
        super().__init__(master)
        self.master = master
        self.master.title("VST list maintenance Ver.1.0")
        self.master.geometry("1420x860")

        #管理者権限判断
        if not is_admin():
            messagebox.showinfo(
                "Information(ご注意)",
                "アプリが管理者権限で起動していません\n 行の右クリックでのファイルの移動などが出来ません\n 管理者権限で起動するとフル機能使えます\n\nThe app is not running with administrator privileges\nYou can't move files by right-clicking on a line\nFull functions can be used when started with administrator privileges."
            )
            #sys.exit(0) #終わらせる場合

        #ウィジット
        self.create_widgets()

    def create_widgets(self):

        # トップエリア(ツリービュー)の作成
        self.tree = ttk.Treeview(self.master, columns=["Name", "Size (MB)", "Access Time", "Folder"])
        # ...

        # 操作方法の説明エリア
        # ボトムエリアの作成
        self.bottom_frame = tk.Frame(self.master, height=60)
        self.bottom_frame.pack(fill=tk.X, side=tk.BOTTOM)

        # 空のラベルを作成してボトムフレームに配置
        self.bottom_label = tk.Label(self.bottom_frame, text="", height=2)
        self.bottom_label.pack(fill=tk.BOTH, expand=True)


        # ラベルと画像のフレームの作成
        self.info_frame = tk.Frame(self.bottom_frame)
        self.info_frame.pack(side=tk.LEFT, padx=60, pady=10)

        # 画像の読み込みと表示
        try:
```

```
            img = Image.open("soncho_logo.png")
            img = img.resize((200,60))  # サイズを縮小する
            img = ImageTk.PhotoImage(img)
            self.logo_label = tk.Label(self.bottom_frame, image=img, height=60, width=200,
cursor="hand2")

            self.logo_label.image = img
            self.logo_label.pack(side=tk.LEFT, padx=20, pady=0)
            self.logo_label.bind("<Button-1>", self.open_website)
        except Exception as e:
            print(f"Failed to load image: {str(e)}")

        # ラベルの作成
        self.info_label = tk.Label(
            self.info_frame,
            text="VST list maintenance Ver.1.0 Development: Soncho Music Studio
URL:https://soncho.jimdofree.com/ \nRight click on row: Move to Folder（ファイル移動） & Move
to Sleep（Sleep フォルダーへ退避） & Delete（削除）\nSleep folder is
C:\\Users\\user\\Documents\\VstSleep",
            anchor=tk.NW,
            #bg="#ffffcc",
            height=3,
            padx=20,
            pady=0,
            justify=tk.LEFT
        )
        # ラベルのフォントを変更
        font = tkFont.Font(self.info_label, self.info_label.cget("font"))
        font.configure(size=13, weight="bold")
        self.info_label.configure(font=font)
        self.info_label.pack(side=tk.LEFT, padx=60, pady=10)

        # パックしないように設定
        self.info_frame.pack_propagate(True)


        self.tree.column("#0", width=0, stretch=0)
        #self.tree.heading("#0", text="No.", anchor=tk.W)
        self.tree.column("Name", width=400, anchor=tk.W, stretch=True)
        self.tree.column("Size (MB)", width=90, anchor=tk.E, stretch=0)
        self.tree.column("Access Time", width=150, anchor=tk.W, stretch=False)
        self.tree.column("Folder", width=750, anchor=tk.W, stretch=True)
```

```
        #self.tree = ttk.Treeview(self.master, columns=["Name", "Size (MB)", "Access Time",
"Folder"])
        #self.tree.heading("#0", text="No.")
        self.tree.heading("Name", text="File Name", command=lambda:
self.treeview_sort_column(self.tree, "Name", False))
        self.tree.heading("Size (MB)", text="Size (MB)", command=lambda:
self.treeview_sort_column(self.tree, "Size (MB)", False))
        self.tree.heading("Access Time", text="Access Time", command=lambda:
self.treeview_sort_column(self.tree, "Access Time", False))
        self.tree.heading("Folder", text="Folder", command=lambda:
self.treeview_sort_column(self.tree, "Folder", False))
        self.tree.pack(side=tk.LEFT, fill="y", expand=0)



        style = ttk.Style()
        style.theme_use("default")
        style.configure("Treeview.Heading", background="#e6ffe6")

        data = []
        folders = [
            r"C:\Program Files\Steinberg\VstPlugins",
            r"C:\Program Files\VstPlugins",
            r"C:\Program Files\Common Files\VST3",
            r"C:\Program Files (x86)\VSTPlugIns",
            r"C:\Program Files (x86)\Common Files\VST3",
            r"C:\Program Files (x86)\Steinberg\VstPlugins",
            sleep_folder  # 追加する
        ]

        for folder in folders:
            if os.path.isdir(folder):
                data.extend(get_files_info(folder))

        data.sort(key=lambda x: x[1], reverse=True)
        for i, row in enumerate(data):
            item_tags = []
            if "VstPluginsSleep" in row[3]:
                item_tags.append("sleep")
            if os.path.dirname(row[4]) == sleep_folder:
                item_tags.append("red")
            item = self.tree.insert("", "end", text=i, values=row, tags=item_tags)

        def treeview_sort_column(self, tree, col, reverse):
```

```
            l = [(tree.set(k, col), k) for k in tree.get_children("")]
            l.sort(reverse=reverse)
            for index, (val, k) in enumerate(l):
                tree.move(k, "", index)
            tree.heading(col, command=lambda: self.treeview_sort_column(tree, col, not reverse))

        # スクロールバーの作成
        self.tree_scrollbar = ttk.Scrollbar(self.master, orient=tk.VERTICAL,
command=self.tree.yview)
        self.tree.configure(yscrollcommand=self.tree_scrollbar.set)

        # Treeview ウィジェットとスクロールバーをパックする
        self.tree.pack(side=tk.LEFT, fill="both", expand=1)
        self.tree_scrollbar.pack(side=tk.RIGHT, fill="y")




        # 右クリックメニュー
        self.popup_menu = tk.Menu(self.master, tearoff=0)
        self.popup_menu.add_command(label="Move to Folder", command=self.move_to_folder)
        self.popup_menu.add_command(label="Move to Sleep", command=self.move_row)
        self.popup_menu.add_command(label="Delete", command=self.delete_row)

        self.tree.bind("<Button-3>", self.show_popup_menu)

    def treeview_sort_column(self, tree, col, reverse):
        data_type = str  # デフォルトは文字列として扱う
        if col == "Size (MB)":
            data_type = float  # ファイルサイズの列の場合は数値として扱う

        for index, iid in enumerate(tree.get_children()):
            tree.move(iid, '', index)
        l = [(data_type(tree.set(k, col)), k) for k in tree.get_children("")]
        l.sort(reverse=reverse)
        for index, (val, k) in enumerate(l):
            tree.move(k, '', index)
        tree.heading(col, command=lambda: self.treeview_sort_column(tree, col, not reverse))


    def show_popup_menu(self, event):
        if not is_admin():
            return
        item = self.tree.identify_row(event.y)
        if item:
```

```
        self.tree.selection_set(item)
        self.popup_menu.post(event.x_root, event.y_root)


def move_to_folder(self):
    selected_item = self.tree.selection()[0]
    file_path = self.tree.item(selected_item)["values"][4]
    dest_folder = filedialog.askdirectory(title="Select destination folder")
    if dest_folder:
        try:
            shutil.move(file_path, os.path.join(dest_folder, os.path.basename(file_path)))
        except Exception as e:
            messagebox.showerror("Error", f"Error occurred while moving file: {str(e)}")
            return

        # 元のファイルが残っている場合は削除する
        if os.path.exists(file_path):
            os.remove(file_path)

        self.tree.delete(selected_item)
        self.update_list()

def move_row(self):
    selected_item = self.tree.selection()[0]
    file_path = self.tree.item(selected_item)["values"][4]
    confirm = messagebox.askokcancel("OK", f"{file_path}\n do you want to move the file")
    if confirm:
        dest_folder = sleep_folder
        os.makedirs(dest_folder, exist_ok=True)
        try:
            shutil.move(file_path, os.path.join(dest_folder, os.path.basename(file_path)))
        except Exception as e:
            messagebox.showerror("Error", f"Error occurred while moving file: {str(e)}")
            return

        # 元のファイルが残っている場合は削除する
        if os.path.exists(file_path):
            os.remove(file_path)

        self.tree.delete(selected_item)
        self.update_list()
```

```
    def delete_row(self):
        selected_item = self.tree.selection()[0]
        file_path = self.tree.item(selected_item)["values"][4]
        confirm = messagebox.askokcancel("OK", f"{file_path}\n do you want to delete the file")
        if confirm:
            os.remove(file_path)
            self.tree.delete(selected_item)
            self.update_list()

    def update_list(self):
        self.tree.delete(*self.tree.get_children())
        data = []
        folders = [
            r"C:\Program Files\Steinberg\VstPlugins",
            r"C:\Program Files\VstPlugins",
            r"C:\Program Files\Common Files\VST3",
            r"C:\Program Files (x86)\VSTPlugIns",
            r"C:\Program Files (x86)\Common Files\VST3",
            r"C:\Program Files (x86)\Steinberg\VstPlugins",
            sleep_folder  # 追加する
        ]

        for folder in folders:
            if os.path.isdir(folder):
                data.extend(get_files_info(folder))

        data.sort(key=lambda x: x[1], reverse=True)
        for i, row in enumerate(data):
            item_tags = []
            if "VstPluginsSleep" in row[3]:
                item_tags.append("sleep")
            if os.path.dirname(row[4]) == sleep_folder:
                item_tags.append("red")
            item = self.tree.insert("", "end", text=i, values=row, tags=item_tags)

    #ホームページに飛ぶ
    def open_website(self, event):
        url = "https://soncho.jimdofree.com/"
        webbrowser.open(url)

if __name__ == "__main__":
    root = tk.Tk()
    app = Application(master=root)
    app.mainloop()  # 括弧を追加
```