Programmer’s Documentation, Final Submission

Kelemen Szimonisz

CIS 443

10/08/2018

Introduction

The purpose of this document is to offer a high-level description of how the auditory clock system works. The clock system offers audio feedback in response to a user’s keystrokes derived from 5 keyboard keys: ‘J’, ‘K,’ ‘L’, ‘;’, and ‘spacebar’. The application is written in Python3. It makes use of two external packages: PyAudio and readchar. PyAudio enables the programmer to invoke commands to playback audio files to the user. The readchar package allows for the programmer to acquire and manipulate single-keystroke inputs from the user.

sound.py

Sound.py enables PyAudio functionality within our main program file: set_clock.py. The class Play, the function combine_wav_files(), and the function cleanup() are the tools provided within this source file. The programmer may instantiate a Play object by providing the directory of an audio file to the Play object’s instantiating argument. Once a Play object is instantiated, the constructor will play the sound file. The Play class manages a ‘PyAudio stream’ that keeps track of the last played sound file. The cleanup() function is used to close the PyAudio stream. The combine_wav_files() function enables the programmer to concatenate multiple audio files together into a new file.

set_clock.py

The function main() invokes the operation of the four primary functions in set_clock.py’s functionality: create_sound_filenames(), verify_sound_filenames(), create_menu_globals(), and run_menu(). The function create_sound_filenames() declares numerous global variables representing the location of each sound file in the file system. These sound files are used for the auditory menus.

The function verify_sound_filenames() is disabled by default. If the programmer wishes to enable this function they may uncomment the function call in main(). Once the programmer enables the function call, verify_sound_filenames() will verify that all sound files can be successfully loaded and played. It instantiates PyAudio Play objects for each of the global variables created in the function create_sound_filenames(). By playing all of the sound files at the start of the program, we can ensure that the pathnames are correct before testing further functionalities of our program. Due to the nature of the PyAudio stream, the very last sound played by this function should be the sound played at the ‘starting menu’ of set_clock.py. Regardless of whether this function is enabled or not, the Intro audio will play.

The function create_menu_globals() is then called by main(). The purpose of this function is to create global constants for setting keystroke controls and global variables for the day of week, hour of day, and minute of day.

Lastly, the main() function invokes the run_menu() function. This function runs the Set Clock menu in an endless loop until the user exits. It makes use of the readchar package to acquire user keystroke input. The run_menu() function essentially runs the ‘main menu’. In this menu, the user can use inputted keystrokes to access submenus (Set Day, Set Hour, Set Minutes, and Set AM/PM sub-menus). Once the user selects a sub-menu, using the select key, run_menu() will call run_submenu(). The run_submenu() function allows the user to select a specific option for their selected sub-menu (If the user selected (Set Day), the sub-menu provide options consisting of all of the days of the week). When the user selects an option within a sub-menu, the program gives control back to the run_menu() function. The user has access to an auditory help menu, using the help key. The user is capable of quitting the program - and thereby exiting the endless loop.