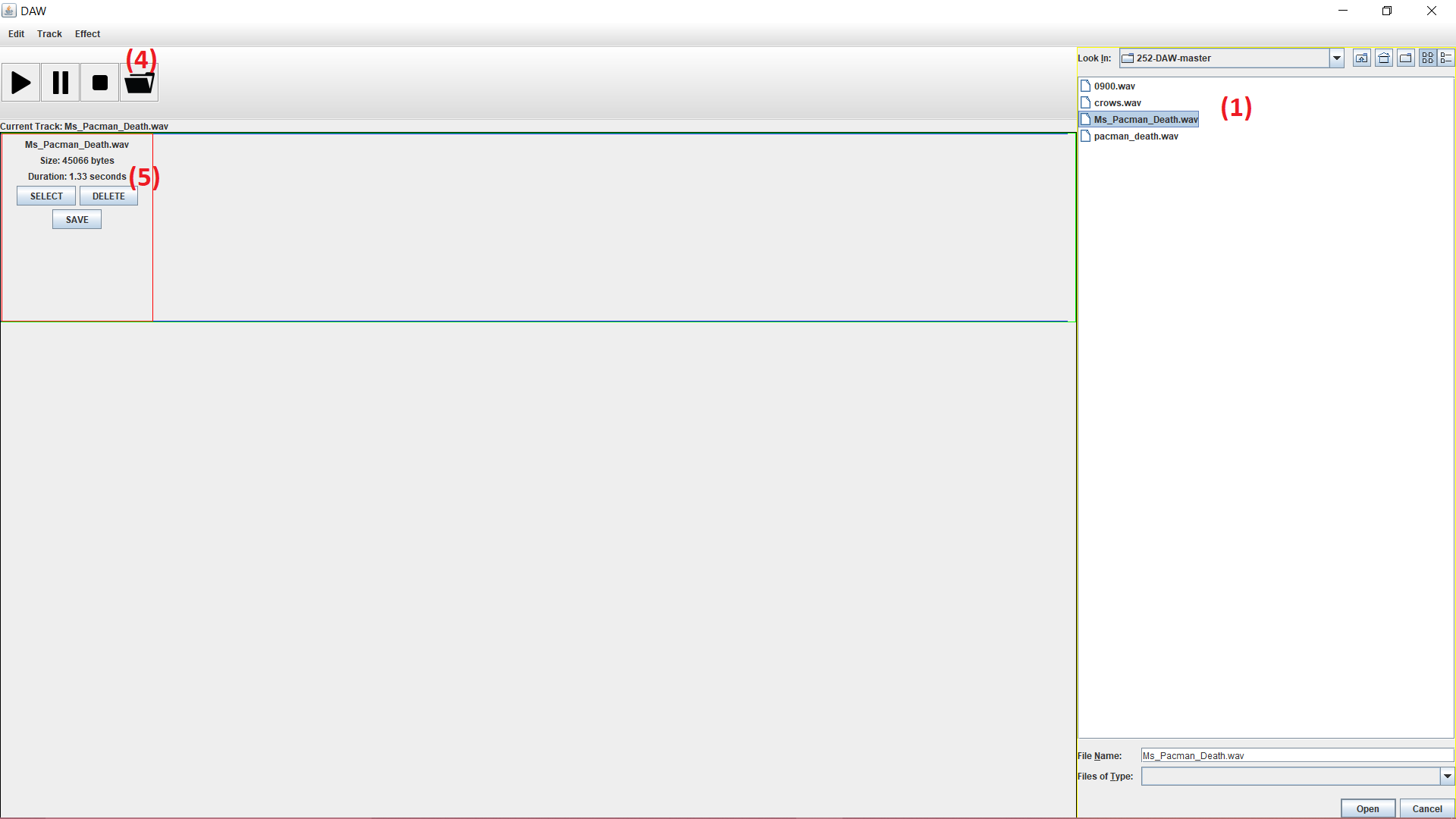
Welcome to the Digital Audio Workstation. It is our hope that this manual will provide detailed instructions on how to use the basic functions of this program and give general explanations regarding the inner-workings of the code.

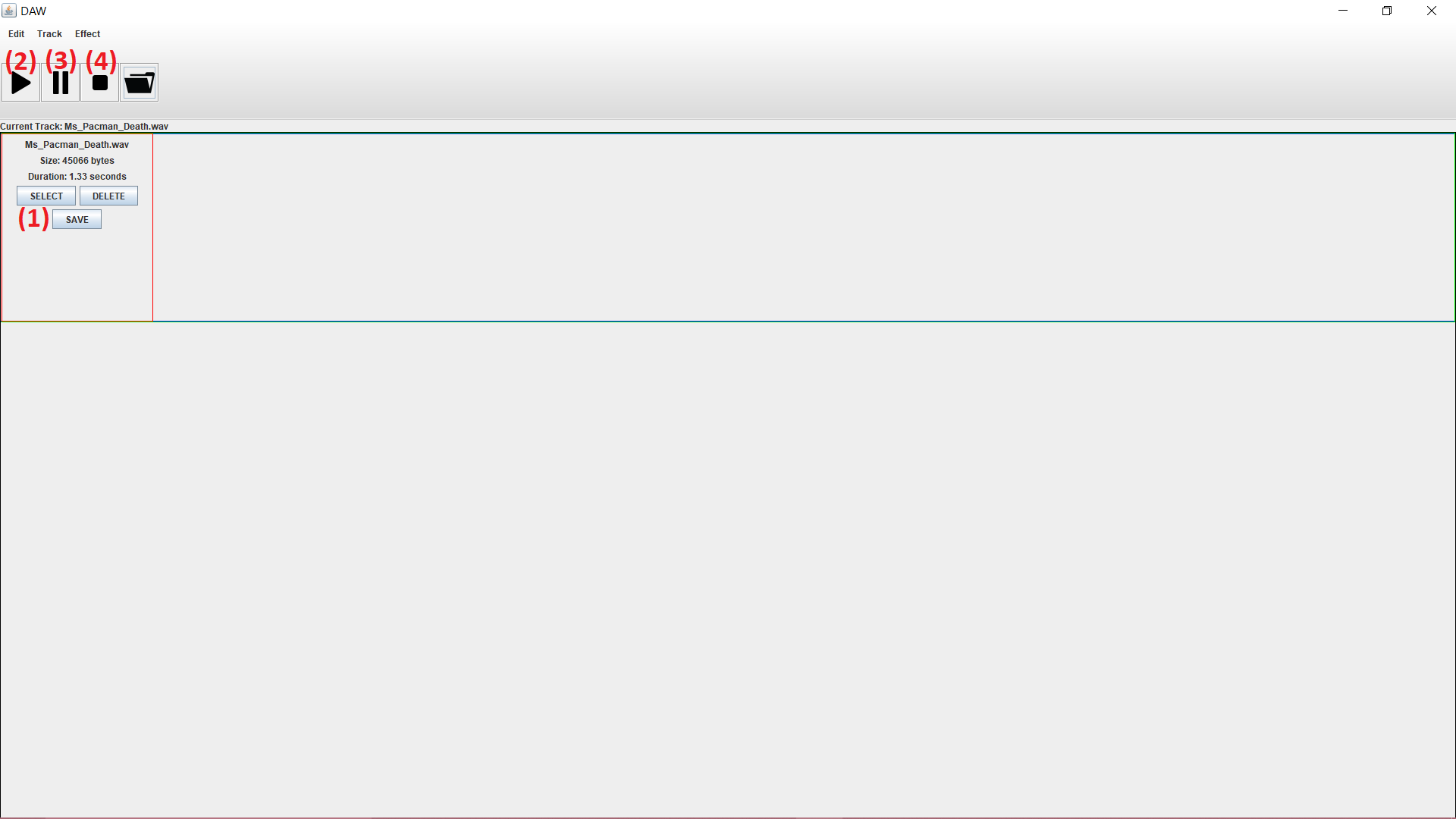
Getting Started:

When the program is started for the first time, the user is greeted with an empty track list. To add .wav files to this list, the user must:

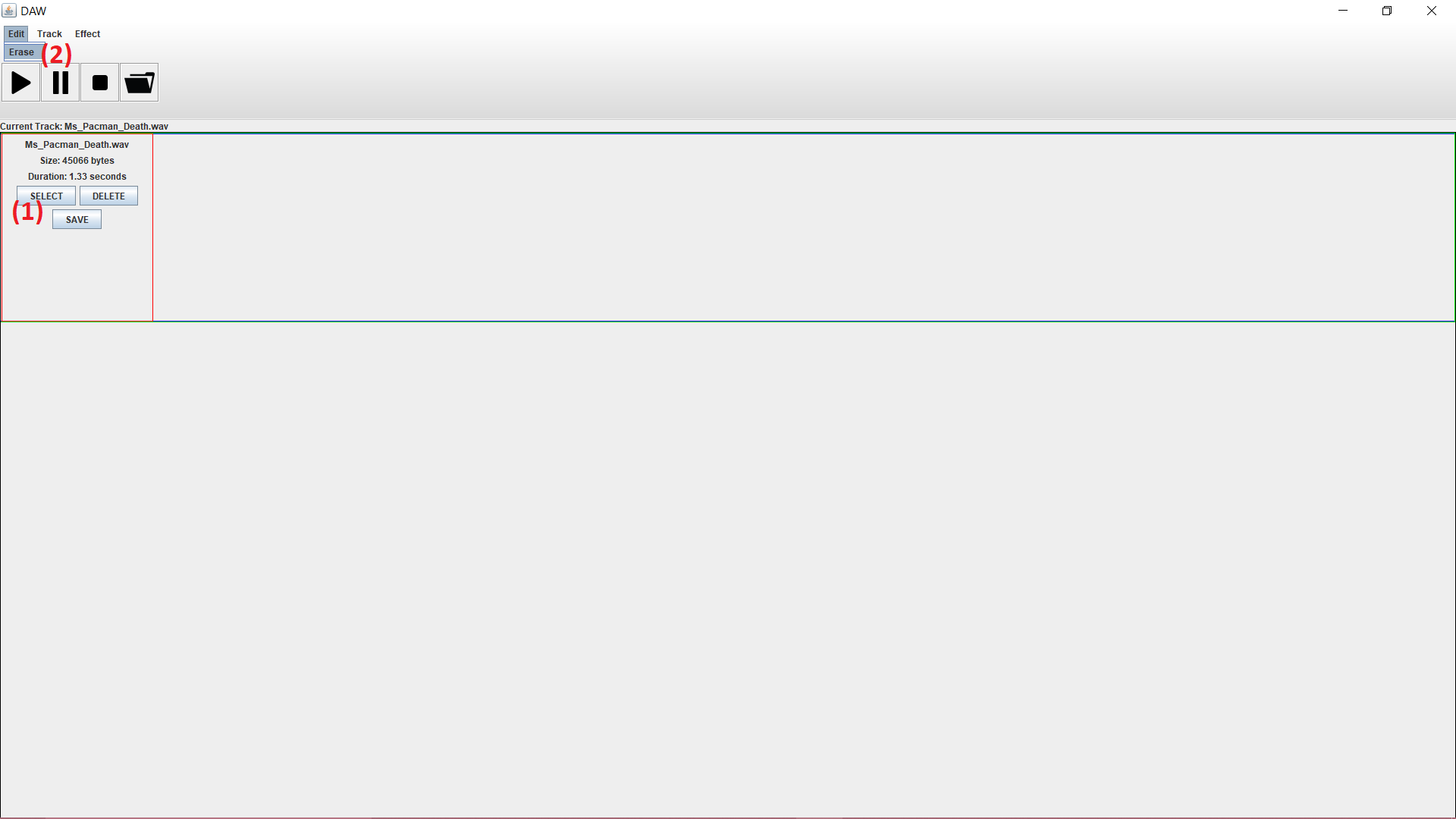
1. Find a .wav file from the explorer on the right side of the screen
2. Once the desired file is found, double click on it to bring it into the DAW’s track list
3. Multiple files can be loaded into the track list, each time a file is double clicked it will be added to the track list
4. The file explorer can be removed from the side of the screen by clicking the file button
5. The track can be removed from the track list by clicking the “Delete” button

Playing a File:

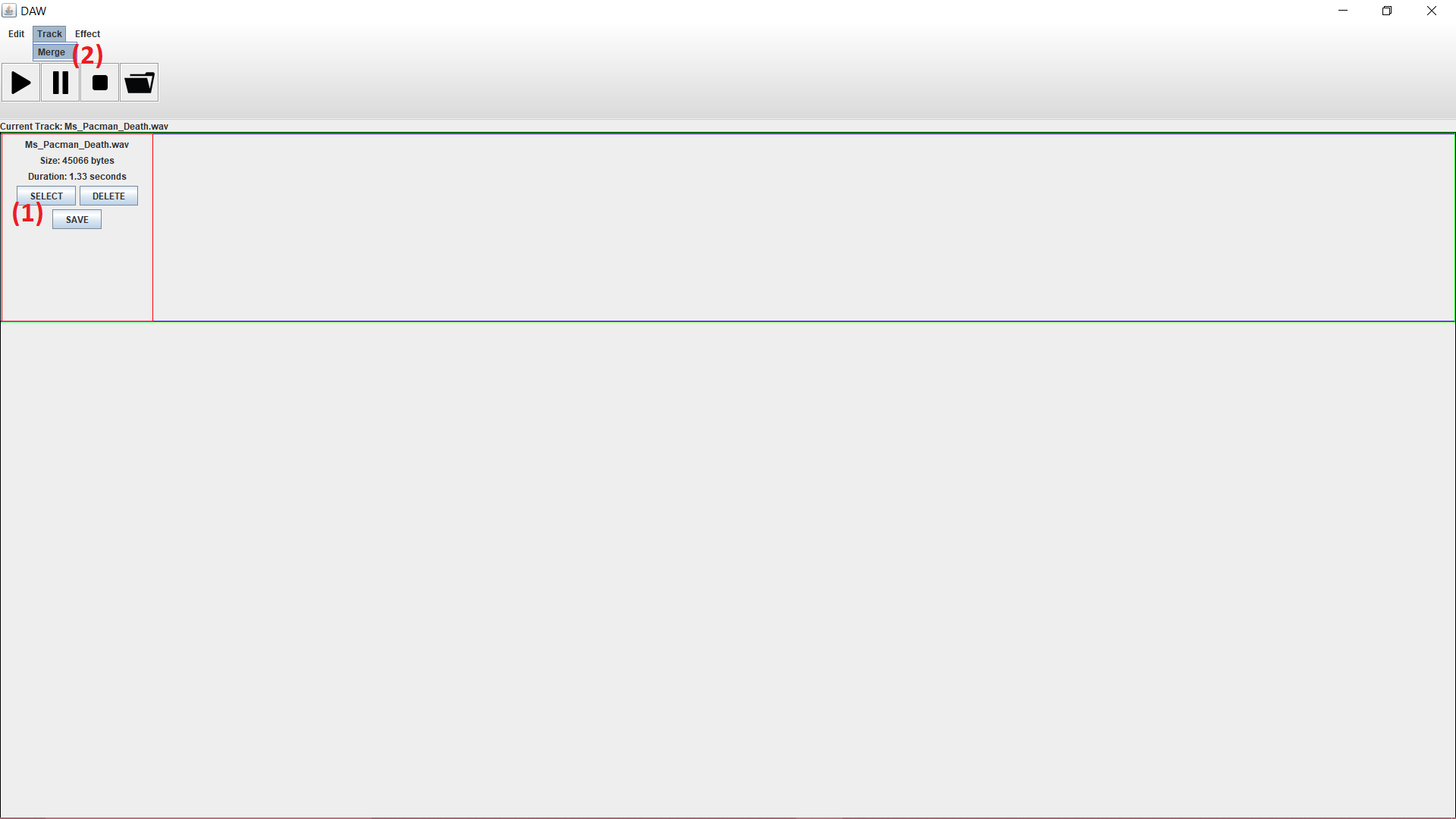
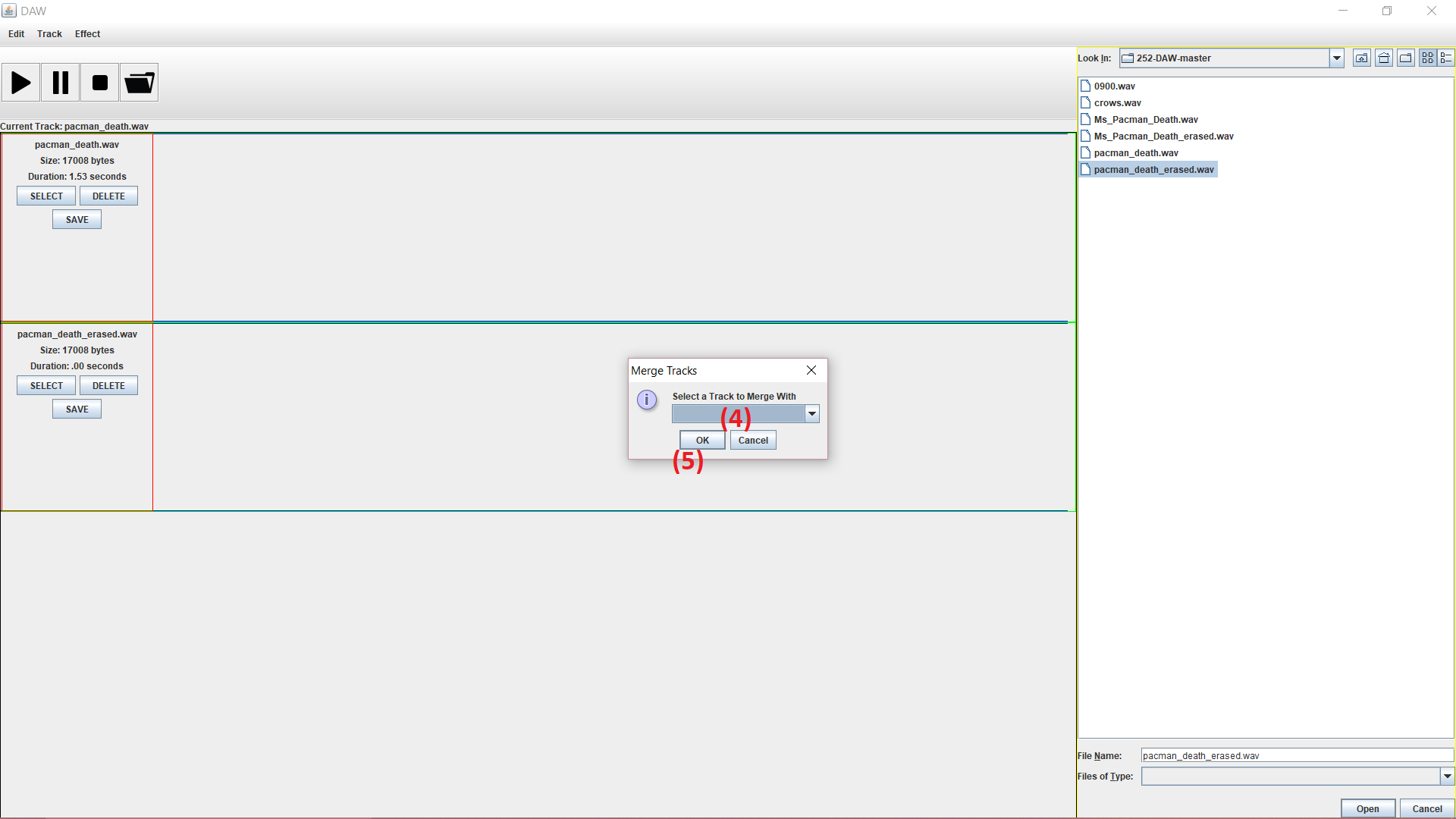
Once a track is loaded into the track list (see “Getting Started”) the user can play, pause, and stop the program using the buttons in the Toolbar in the top quadrant of the window.

1. The user must first select the track they wish to play. This can be done by clicking “select”
2. If the Play button is clicked, the track will begin from the current position it is set at, the default being 0
3. If the Pause button is clicked, the track will stop playback and stay at the time where it was stopped. Playback will not resume until Play has been clicked again
4. If the Stop button is clicked, the track will stop playback and set the position (in milliseconds) of the track to 0 

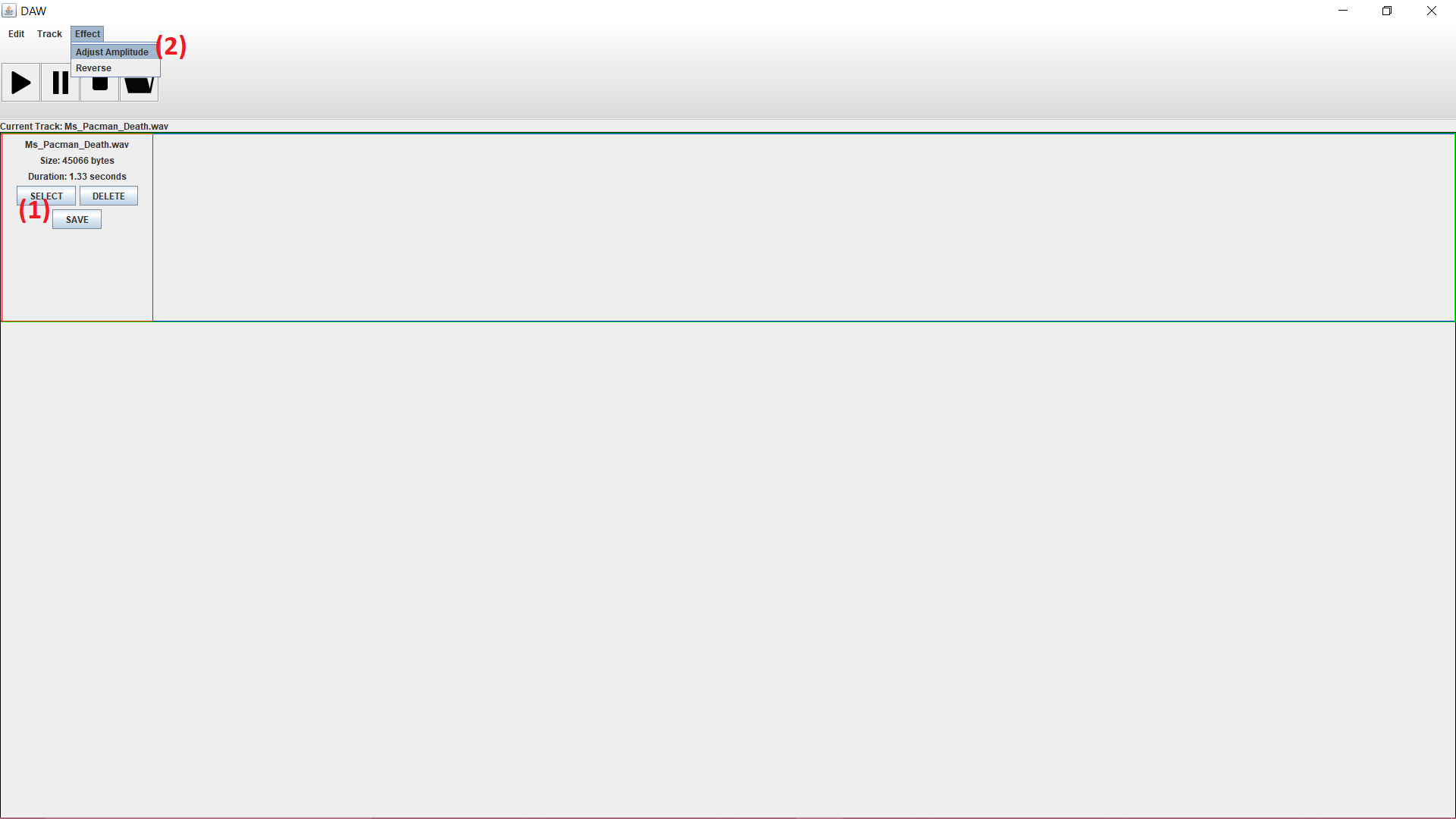
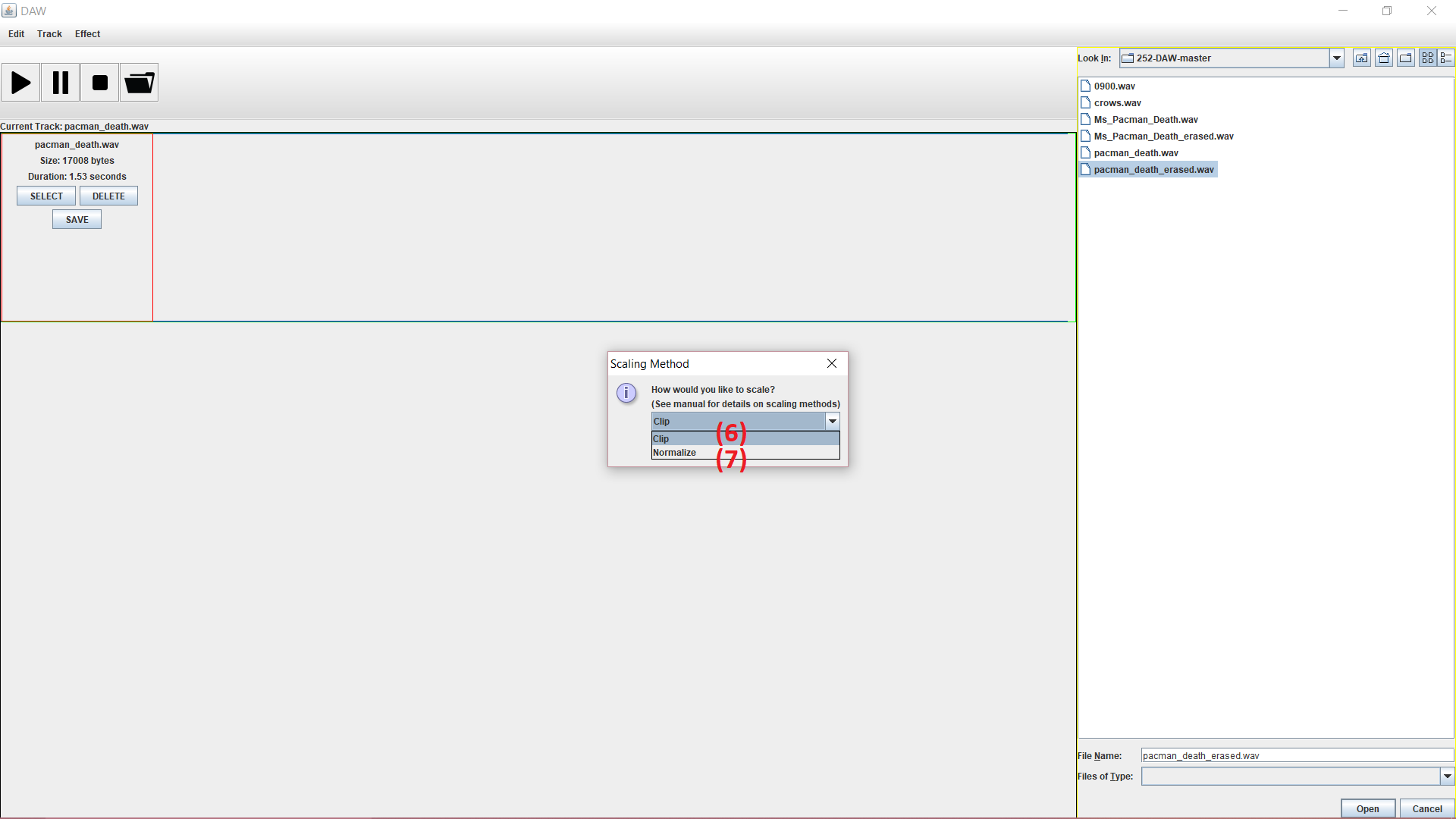
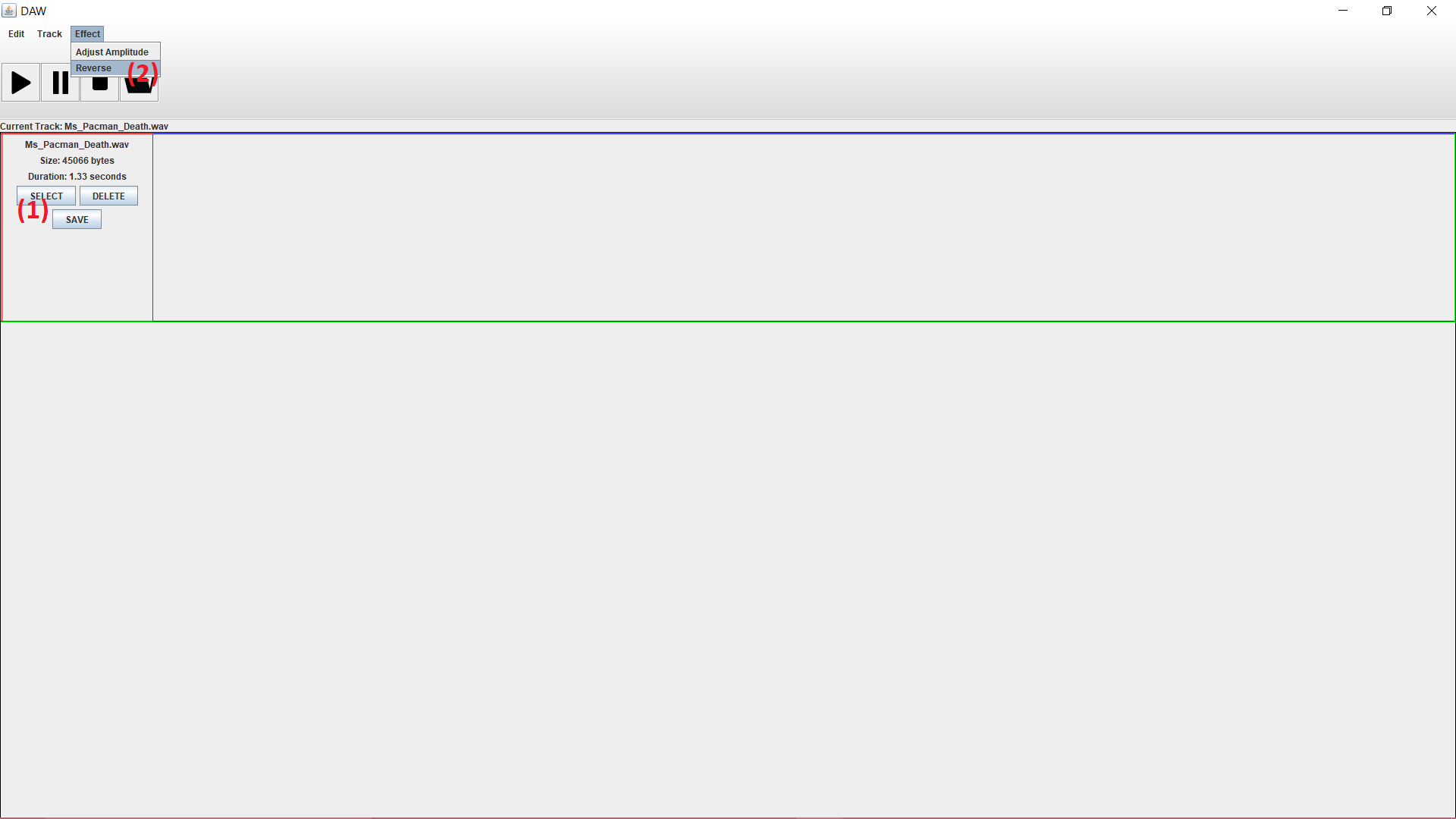
Edit Menu:

1. Erase
   1. This option will silence a track, essentially making an empty .wav file
      1. The user must first select a track in the tracklist to erase. To do this, the “select” button must be pressed on the desired track
      2. After selecting a track to erase, the user can then go to Edit → Erase to remove the sound from the file
      3. The new, empty track will be saved in the same location as the original file
      4. The new track will show up in the track list

Track:

1. Merge
   1. This function will allow the user to bring two sounds together, essentially playing them on top of each other.
      1. There must be at least two tracks in the tracklist to use this function
      2. To begin, the user must select a track using the “select” button
      3. After making a selection, the user can go to Track → Merge to begin merging
      4. The user will be shown a popup with a drop down menu. Here is where the desired track to be merged with can be chosen from
      5. After choosing a track to merge the selected track with, the user clicks “Ok”
      6. The program will perform the merge and then save the new track to a file in the same location as the file that was merged to the original
      7. The new track will show up in the track list

Effect Menu:

1. Adjust Amplitude
   1. This option allows the user to change the amplitude of the sound (the extent of its vibration)
      1. The user must first select a track in the tracklist to adjust. To do this, the “select” button must be pressed.
      2. After the desired track is selected, the user can then go to Effect → Adjust Amplitude to begin adjusting amplitude 
      3. Clicking “Adjust Amplitude” will bring up a window with a box for input
      4. In the box, the user inputs a percentage they would like to raise or lower the amplitude by (<100 for lowering, >100 for raising)
      5. After inputting a percentage to adjust the sound by, the user will be shown a drop down menu with two options
      6. The first option will allow the audio to be clipped (if the amplitude goes beyond the 16 bit scope of the .wav file)
      7. The second option will allow the audio to be normalized
      8. After choosing a desired percentage and one of the options, the user can click “Ok” to make the changes to the selected track
      9. The program will make the adjustments and save them to a new file in the same location as the original file
      10. The new track will show up in the track list
2. Reverse
   1. This option will reverse the audio of the selected track
      1. The user must first select a track in the tracklist to reverse. To do this, the “select” button must be pressed
      2. After selecting a track, the user can go to Effect → Reverse, this will perform the function on the selected audio
      3. The program will reverse the sound and save it to a new file in the same location as the original file
      4. The new track will show up in the track list

Classes

DAW (main class)

AudioDisplayContainer extends JPanel implements ActionListener

* This JPanel holds two other JPanel classes, AudioFileInfo and AudioFileVisualDisplay

AudioFileInfo extends JPanel implements ActionListener

* This is where basic file info including name, length, and size are displayed
* This is also where the select, delete, and save buttons are located

AudioFileVisualDisplay extends JPanel implements ActionListener

* This is where real time info about the file will be displayed

FileExplorerWindow extends JFileChooser implements ActionListener

* This window holds the file explorer used for finding and importing .wav files into the program

MainDisplayWindow extends JPanel

* This class holds all audio swing components are held
* The MainDisplayWindow acts as the hub for all file editing, so algorithms for various functions such as reverse and adjust amplitude are performed here

MenuBar extends JMenuBar implements ActionListener

* This class holds the drop-down menus for Edit, Track, and Effect

ProgramFrame extends JFrame

* This is where swing components are added
* This is also where objects that are considered “global” are held because it is the highest level of the program

ToolBar extends JToolBar implements ActionListener

* This is where the controls for the track (Play, Pause, and Stop) are held
* This is also where the show/hide button for the file explorer is held

Algorithms

1. Erase
   1. This algorithm works by
      1. Reading the audio data information from a FileInputStream
      2. Smashing each pair of bytes into a short
      3. Setting that short to 0
      4. Extracting new byte pairs from the short
      5. Writing the new zeroed byte pairs to the FileOutputStream
2. Adjust Amplitude (Normalize)
   1. This algorithm works by
      1. Reading the audio data information from a FileInputStream and finding the largest byte (maxByte)
      2. Modifying the user-given scaling ratio so that no byte can be made greater than 127
      3. Re-reading the audio data in byte pairs and smashing them into shorts
      4. Multiplying the shorts by the adjusted scaling ratio
      5. Extracting the new byte pairs from the shorts
      6. Writing the new bytes into a FileOutputStream
3. Adjust Amplitude (Clip)
   1. This algorithm works by
      1. Reading the audio data information from a FileInputStream
      2. Reading in pairs of bytes and smashing them into shorts
      3. Multiplying the shorts by the user-given scaling ratio
      4. Extracting the new byte pairs from the short
      5. Setting any bytes that go over 127 to 127
      6. Writing the new bytes into a FileOutputStream
4. Reverse
   1. This algorithm works by
      1. Reading the audio data information from a FileInputStream
      2. Reading in byte pairs and smashing them into shorts
      3. Reads the shorts into a short array
      4. Taking the first and last shorts in the array and flipping them so that the first is last and last is first
      5. Moving to the second and second to last byte pairs and flipping those
      6. Continuing to move in through the pairs and flipping them until the entire array is reversed
      7. Extracting the byte pairs from the newly arranged shorts
      8. Writing the new bytes into a FileOutputStream
5. Merge
   1. This algorithm works by
      1. Reading the audio data information from two FileInputStreams
      2. Smashing byte pairs from each FileInputStream into two separate shorts
      3. Adding the shorts of FileInputStream 1 and 2 together
      4. Extracting byte pair from the new short
      5. Writing the new byte pair to the FileOutputStream