THE AUDITRANSCRIBE FILE FORMAT SPECIFICATION

For Application Version 0.3.0, Revision 2

File Version: 302

This document serves two purposes.

1. To give a detailed explanation for the AudiTranscribe file format and how to use it.
2. To assist developers and contributors in understanding how to process and format their data into the required specification.

We separate the remainder of the document into the following parts.

General File Format 2

Heading Data Section 3

Section 1. Q-Transform Data Section 3

Section 2. Audio Data Section 4

Section 3. GUI Data Section 4

Section 4. Music Notes Data Section 5

End-Of-File Section 5

General File Format

An AudiTranscribe file ends with the extension `.audt`. The file is a binary file containing binary data that can be understood by AudiTranscribe internals.

Notes on the convention of the format described below:

- A sequence of 4 bytes, such as `12 34 56 AB`, will be written as one 16-bit hexadecimal number, such as `0x123456AB`.
- Sections of data separated by two newlines (not in byte form, but just what is shown) should be **joined together** in the actual file.
- Boxed sections with dotted borders denote arbitrary data. These sections are labelled with the supposed data that should be in these sections.
  - Text in Arial (and not `Courier New`) also denotes arbitrary data.

Here's an example file.

```
0x41554449 0x5452414E 0x53435249 0x42450A0A
0xAD75C1BE 0x00000ABC 0x0000EF12 0xE05E05E5

0x00000001
0xC05EDCCC 0xCCCCCCCD 0x40C34A45 0x87E7C06E
0x0013579B
Q-Transform data goes here
0xE05E05E5

0x00000002
0x002468AC
Audio data goes here
0x40e58880 0x00000000
0x000493E0
0x00000142 [Original audio file name]
0xE05E05E5

0x00000003
0x0000000B
0x00000009
0x00029300 0x29394010
0x00000001 0x03918939
0x00000004 0x0812BA23
0x00018324
0xE05E05E5

0x00000004
0x00000018
Times to place note rectangles goes here
0x00000018
Note rectangles' durations go here
0x00000018
Note rectangles' note numbers go here
0xE05E05E5
```

```
0xE0FE0FEF 0xE0FE0FEF
0x0139ACE9
```

- The 4 bytes `0xE05E05E5` are used to delimitate data sections.
- The 8 bytes `0xE0FE0FEF 0xE0FE0FEF` are used to mark the end of the file.
- The last value `0x0139ACE9` is the checksum for the file. This checksum is the sum of all bytes of the file, but modulo 2^{32} (4,294,967,296). This is equivalent to getting the byte representation of the integer sum of all the bytes in the file.

Heading Data Section

The header section corresponds to the following 32 bytes in the example file given.

```
0x41554449 0x5452414E 0x53435249 0x42450A0A
0xAD75C1BE 0x00000ABC 0x0000EF12 0xE05E05E5
```

- The first 16 bytes, `0x41554449 0x5452414E 0x53435249 0x42450A0A`, decodes to the string "`AUDITRANSCRIBE\n\n`" when decoded in ASCII. The `\n`'s present are line breaks.
- The following 4 bytes (`0xAD75C1BE`) are the AudiTranscribe signature. After the two line breaks, these 4 bytes are to follow the first 16 bytes.
- The next 4 bytes (`0x00000ABC`) correspond to the version number of the file format specification.
- The following 4 bytes (`0x0000EF12`) describe the version number of the LZ4 compression algorithm.
- The final 4 bytes (`0xE05E05E5`) correspond to the end-of-section delimiter.

These 32 bytes must be present in every AudiTranscribe file, the first 20 of which **must be what is shown above**.

Section 1. Q-Transform Data Section

The Q-Transform data section corresponds to the following lines in the example file given.

```
0x00000001
0xC05EDCCC 0xCCCCCCCD 0x40C34A45 0x87E7C06E
0x0013579B
Q-Transform data goes here
0xE05E05E5
```

- The first 4 bytes (`0x00000001`) correspond to the section ID. The Q-Transform data section has **section ID 1**.
- The next 8 bytes (`0xC05EDCCC 0xCCCCCCCD`) correspond to the smallest magnitude value present in the (decompressed) Q-transform matrix.
- The following 8 bytes (`0x40C34A45 0x87E7C06E`) correspond to the largest magnitude value present in the (decompressed) Q-transform matrix.
- The next 4 bytes (`0x0013579B`) correspond to the number of bytes that are used to store the LZ4 compressed Q-Transform data.
- The next `0x0013579B` bytes represent the LZ4 compressed Q-Transform data. The program will decode this data to retrieve the original Q-Transform data from the LZ4 compressed version.
- The final 4 bytes (`0xE05E05E5`) correspond to the end-of-section delimiter.

Section 2. Audio Data Section

The audio data section corresponds to the following lines in the example file given.

```
0x00000002
0x002468AC
Audio data goes here
0x40e58880 0x00000000
0x000493E0
0x00000142  [Original audio file name]
0xE05E05E5
```

- The first 4 bytes (`0x00000002`) correspond to the section ID. The audio data section has **section ID 2**.
- The next 4 bytes (`0x002468AC`) correspond to the number of bytes that are used to store the audio data.
- The following `0x002468AC` bytes correspond to the compressed audio data.
- The next 8 bytes after that (`0x40e58880 0x00000000`) represent the sample rate of the audio file.
- The next 4 bytes (`0x000493E0`) represent the total duration of the audio **in milliseconds** as an integer.
- The next 4 bytes (`0x00000142`) denote the number of bytes that are used to store the original audio file name.
- The following `0x00000142` bytes correspond to the original audio file name.
- The final 4 bytes (`0xE05E05E5`) correspond to the end-of-section delimiter.

Section 3. GUI Data Section

The GUI data section corresponds to the following lines in the example file given.

```
0x00000003
0x0000000B
0x00000009
0x00029300 0x29394010
0x00000001 0x03918939
0x00000004 0x0812BA23
0x00018324
0xE05E05E5
```

- The first 4 bytes (`0x00000003`) correspond to the section ID. The GUI data section has **section ID 3**.
- The next 4 bytes (`0x0000000B`) correspond to the music key index. This will be the index of the music key in the dropdown menu shown in the application.
- The following 4 bytes (`0x00000009`) represent the time signature index. This will be the index of the time signature in the dropdown menu shown in the application.
- The next 8 bytes (`0x00029300 0x29394010`) represent the beats per minute.
- The following 8 bytes (`0x00000001 0x03918939`) represent the offset seconds.
- The next 8 bytes (`0x00000004 0x0812BA23`) represent the playback volume.
- The next 4 bytes (`0x00018324`) represent the current time when playing the audio file **in milliseconds** as an integer. This is when playback will continue.
- The final 4 bytes (`0xE05E05E5`) correspond to the end-of-section delimiter.

Section 4. Music Notes Data Section

The music notes data section corresponds to the following lines in the example file given.

```
0x00000004
0x00000018
Times to place note rectangles goes here
0x00000018
Note rectangles' durations go here
0x00000018
Note rectangles' note numbers go here
0xE05E05E5
```

- The first 4 bytes (`0x00000004`) correspond to the section ID. The music notes data section has **section ID 4**.
- The next 4 bytes (`0x00000018`) denote the number of bytes used to store the array containing the times to place note rectangles.
- The following `0x00000018` bytes correspond to the array containing the times to place note rectangles.
- The next 4 bytes (`0x00000018`) denote the number of bytes used to store the array containing the note rectangles' durations.
- The following `0x00000018` bytes correspond to the array containing the note rectangles' durations.
- The next 4 bytes (`0x00000018`) denote the number of bytes used to store the array containing the note rectangles' note numbers.
- The following `0x00000018` bytes correspond to the array containing the note rectangles' note numbers.
- The final 4 bytes (`0xE05E05E5`) correspond to the end-of-section delimiter.

It is important to note that the lengths of all the arrays provided must be the same. Also, elements at the same index of different arrays correspond to the **same note rectangle**.

End-Of-File Section

The end-of-file section should **always** consist of 12 bytes.

```
0xE0FE0FEF 0xE0FE0FEF
0x0139ACE9
```

- The first 8 bytes (`0xE0FE0FEF 0xE0FE0FEF`) are the end of file delimiter. These 8 bytes should only appear at the end of the file.
- The last 4 bytes (`0x0139ace9`) correspond to the checksum value, as explained in the General File Format.

- END OF DOCUMENT -