Ableton Programming Tasks: Context and Objective

In this document you will find two tasks. Both tasks are somewhat related to our product Ableton Live and the content add-ons called Packs. Please complete both tasks. We have tried to make the descriptions self-contained to ensure that no in-depth knowledge of Ableton Live or Packs is required for their proper completion.

The tasks should take no more than 1.5 hours each.

Your submission should include one Python file for the first task, and one text document for the second task, in a zip archive.

Task 1: Audio preview post-processing

We generate audio previews of each Live preset, to give Live users a taste of what a preset might sound like before loading it. After the preset has been rendered to audio, the sample is post-processed and compressed.

The proprietary sample conversion software used previously has been discontinued and needs to be replaced. Your task is to replace this functionality with a Python script, using ffmpeg to do the audio post-processing, making use of the audio filters functionality (https://ffmpeg.org/ffmpeg-filters.html#Audio-Filters). The script is intended to run on a Unix system.

Requirements:

- Your script should accept command line arguments as described in the specification. No extra input (e.g. command line options) should be necessary to successfully run the application.
- Your submission should only use the Python standard libraries and ffmpeg. Please do not use any libraries which require special installation.

Audio filter specification for ffmpeg:

- Trim sample to maximum duration of 2 seconds
- Compression applied with settings: threshold=0.1, ratio=2, attack=5
- 20 milliseconds fade out at end of sample

The output file should be an ogg file, with the base name remaining the same (for instance, `sample1.aif` becomes `sample1.ogg`).

You can assume that ffmpeg with ogg support is installed at `/usr/local/bin/ffmpeg`, and that the Python script will only be run with valid inputs.

Usage

The arguments for the script are files paths (relative or absolute) to each audio file to be processed.

Example:
```
$ python preview_postprocess.py samples/kick.aif samples/snare.aif
samples/extra/cowbell.aif
```

Output

One ogg file for each input file, processed with the filters specified, located in the same directory as the corresponding source aif file.

```
$ ls -R samples/
samples/:
kick.aif kick.ogg snare.aif snare.ogg

./samples/extra:
cowbell.aif cowbell.ogg
```

Task 2: `baseXmlHelper.py` code review

Many Ableton Live documents, such as presets and Live sets, are XML files. To make it easier to work with these XML files, we use a class called `xml_helper`, which can read and write XML files and provides many functions to manipulate XML data. Your task is to review the `baseXmlHelper.py` file which contains this class.

We'd like you to identify areas of improvement, and explain the reasoning behind the changes you propose. Any suggestion is welcome, whether it's implementation, code style, Python 3 compatibility, useful libraries, etc., but doesn't need to cover everything. The important part of this task is your explanation and reasoning and how you communicate it.

Requirements:

- Your review should be in a single text file or document, included with your submission.
- For each suggestion, state your proposed change, and then explain your reasoning for the change. You can include code in your answers if that is helpful.
- When referring to specific parts of the code, please give the line number range in question. This is not necessary for suggestions that apply to the whole file.