tonic

tonic - play music together

Brief Description

Tonic is a software for the digitization and editing of music scores. It allows musicians to capture sheet music in digital form, automatically recognize the musical notation, and adjust the recognized notes. This facilitates handling sheet music for various instruments and allows exporting the scores in a printable format such as PDF.

Background

Many musicians who work with different instruments need to manually edit sheet music since some instruments are played in different keys (e.g., woodwind instruments not tuned in the key of C). This process is prone to errors and time-consuming. **Tonic** aims to automate and simplify this process by enabling the scanning, recognition, and editing of sheet music.

Objectives

1. **Sheet Music Scanning:** Tonic can import high-quality sheet music from .pdf or image files (e.g., JPG, PNG).

2. **Automatic Note Recognition:** The software can recognize simple musical notations from the scanned sheet music.

3. **Transposition:** The system should enable automatic transposition of notes into different keys to adapt them for various instruments.

4. **Export:** Tonic provides the option to export the edited sheet music as printable PDF files.

Extended Objectives:

1. **Sound Output:** The software should allow playback of the recognized and edited notes as an audio file (sound).

2. **Instrument Selection:** Tonic should allow users to select a musical instrument to play the notes in the corresponding timbre.

Installation

Clone this repository to your local machine:

```
git clone https://github.com/tonicmusic/tonic.git
```

Optical Music Recognition (OMR) Options

For Optical Music Recognition (OMR), Tonic provides two methods:

Template Matching

Template Matching is an experimental feature built into tonic that offers basic OMR functionality. While it allows users to detect and process simple musical notations, it is prone to frequent errors. For more accurate and robust results, we recommend using Audiveris.

Audiveris Integration

Audiveris is integrated into Tonic, providing advanced Optical Music Recognition (OMR) capabilities. Audiveris offers comprehensive recognition features for complex scores and is a powerful option for users needing detailed analysis.

Java Requirement: Audiveris requires Java to run. Ensure that the Java Runtime Environment (JRE) is installed on your system before using Audiveris with Tonic. You can download the latest version of Java here.

- **Download Audiveris**: GitHub Repository
- **More Information**: Audiveris Website

Note: For the best experience viewing and playing back MusicXML files generated by tonic, we recommend installing **MuseScore 4**, a free and widely-used music notation software. MuseScore 4 allows you to easily view, edit, and listen to the MusicXML files generated by tonic. See here for Downloading and info: MuseScore Download Page.
Check and update the path to your Musescore executable in 'pdfgenerator.py'.

Prepare a virtualenv and install the required libraries.

```
cd tonic/backend
pip install setuptools
python -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
```

The requirements file lists all dependencies. Currently, the following libraries are used.

- argparse
- matplotlib
- midiutil
- music21
- numpy

- opencv-python-headless
- wand
- pillow

PlantUML Diagrams

This project includes diagrams created with PlantUML. To view or edit these diagrams, you'll need to install the PlantUML plugin in your IDE or text editor (e.g., Visual Studio Code, IntelliJ IDEA, or Atom). Additionally, the plugin may require a Java runtime environment to function properly.

Working with tonic

Try the help to learn how to use `tonic`.

```
python src/main.py --help

usage: main.py [-h] -i INPUT [-o OUTPUT]

tonic, play music together

options:
  -h, --help                      show this help message and exit
  -i, --input INPUT               Image input file [pdf, png, mxml].
  -o, --output OUTPUT             Output filename. Supported suffixes are [mxml,
pdf]
  -n, --interval INTERVAL         Interval for transposition (if input is
MusicXML).
  -t, --title TITLE               Optional title for transposed piece (if input is
MusicXML).
  -c, --clef   CLEF               Clef for transposed piece. Choose between 'bass'
and 'treble'.
  -ins, --instrument INSTRUMENT   Choose an instrument for the part. Options
include: Piano, Guitar, Flute, Bassoon, Tenorsaxophone
```

At least an input file must be provided.

The suffix of the input and output files determine the mode of `tonic`. It is possible to

- Convert a PNG or PDF file to MusicXML
- Transpose a MusicXML
- Export a MusicXML to PDF

Examples

- To convert a music sheet image file (PNG or PDF) into MusicXML format:

  ```
  python src/main.py -i input/song.png -o output/song.xml
  ```

- Export a MusicXML to PDF.

  ```
  python src/main.py -i output/song.xml -o output/song.pdf
  ```

- Directly export an input image file in PNG or PDF format to PDF.

  ```
  python src/main.py -i input/song.png -o output/song.pdf
  ```

- Transpose a MusicXML source.

  ```
  python src/main.py -i input/song.xml -o output/transposed_song.xml -n=-M2
  ```

- To transpose a MusicXML document and add a custom title:

  ```
  python src/main.py -i input/song.xml -o output/transposed_song.xml -n=-M2 --
  title "Transposed Title"
  ```

- To convert an image file (PNG or PDF) to MusicXML format and transpose it immediately:

  ```
  python src/main.py -i input/song.pdf -o output/transposed_song.xml -n=-M2 --
  title "Transposed Title"
  ```

- To convert an image file (PNG or PDF) to MusicXML, transpose it, and specify the desired musical instrument:

  ```
  python src/main.py -i input/song.pdf -o output/transposed_song.xml -n=-M2 --
  title "Transposed Title" -ins "Bassoon"
  ```