Software Construction | Spring 2016
Sofia Wyetzner | Ivy Malao | Zoë Naidoo | Annie Chen | Rachel Pinsker | Zakir Gowani

impromptu

interactive music notation

Team members: Ivy Malao, Zoë Naidoo, Annie Chen (Mengyuan Chen), Rachel Pinsker, Zakir Gowani, Sofia Wyetzner

MILESTONE 4.A

1. What we plan to implement in the 2nd iteration.

- Upload mp3 audio file (**frontend** -- Zakir)
- Implement clef, key, time signature on PDF display (**frontend** -- Rachel)
- Process and render chords (**frontend** -- Rachel and **backend** -- Ivy/Zoë)
    - Chord, Note, and Rest will now be subclasses of the superclass Event
- Support multiple MIDI format types, e.g. when no note off event (**backend** -- Annie)
- Support one-track/single-instrument mp3 audio file by generating Tune object directly from mp3 audio file (**backend** -- Ivy and Zoë):
    - Extract note onsets, durations, and frequencies from mp3 audio file
    - Calculate note pitches from frequencies
- Record instrument and store as mp3 file (**frontend** -- Sofia)
- Edit sheet music (**frontend**):
    - Adding/Deleting notes -- Zakir
    - Editing a note's pitch and duration -- Rachel
    - Displaying notes of a given measure for editing -- Sofia
- Save files as impromptu sheet music. This means downloading as a json file to later upload and re-convert into a Tune object for display. Frontend for creating json file, downloading json file, and saving uploaded json file and backend for reading a json file to make a Tune object. (**frontend** -- Sofia and **backend** -- Annie)

2. A brief description about how the work will be divided among 2 or 3 pairs of people in your team.

Please see the names in parenthesis above for our planned work division.

3. Unit test cases

https://github.com/rpinsker/impromptu

Relevant files:

- **tests/iter2backendtests.py**
- **tests/flaskFrontendTests.py** (new tests are below line 238)

Note: The verbose flag (-v) allows you to run the test with more detail to see which tests pass and fail. For this iteration, most tests will fail as much of the code is not yet implemented and the class structure is not yet updated.

Backend tests: When inside the src folder, run: `python ../tests/iter2backendtests.py -v`

Frontend tests: When inside the src folder, run: `python ../tests/flaskFrontendTests.py -v`