SynthSong

Course main assignment

Description

Design and implement a C++ console application that plays a melody with a custom synthesizer.

Within this assignment you can either focus on the creation of synthesizers (SYNTH FOCUS) in c++ or the generation of a melody (MELODY FOCUS). The first focus results in a **‘synthesizer demonstration application’**, where at least two customly designed and self-implemented synthesizers are demonstrated in a C++ console application. The second focus, the focus on the generation of a melody, results in a melody generator with synthesized playback, also in the form of a C++ console application.

Noteworthy, due to the two offered options regarding design focus, the specific deliverables and requirements for the SYNTH FOCUS and MELODY FOCUS are shared separately below.

Deliverables

Deliverables are submitted to an online git repository.

- An **overall description** of the application (two paragraphs), which should be informative regarding:
  - The chosen focus (custom synthesizers / melody generation)
  - Functional design, describing the user interaction flow
- **Functional flow diagram**, the user interaction flow together with the overall processes captured in a diagram
- SYNTH FOCUS
  - A **description** of the two synthesizer designs, accommodated with a **personal motivation** for these two chosen designs.
  - **2x Audio flow diagram** (one for each synthesizer design)
  - **2x** Audio flow captured in **pseudo code** (one for each synthesizer design)
- MELODY FOCUS
  - A **description** of the applied melody generation strategy, accommodated with a **personal motivation** for this strategy.
  - **Melody generation flow diagram**
  - **Melody generation flow** captured in **pseudo code**

- A basic **class diagram**, displaying inheritance and association relationships between classes
- **Codebase**
- **Selected learning goals and reflection**, a one page document (+/- 400 a 600 words) describing:
    - The three **selected learning goals**, provided with a **reflection** per learning goal
    - **Overall reflection**, covering:
        - The design and implementation *(e.g. proud about, possible improvements, etc.)*
        - Learned knowledge/skills *(besides the selected learning goals)*
        - Process *(e.g. remarkably good planning, struggles)*
    - Reflection transformed into **at least two take-aways**, relevant for the upcoming course

Requirements

Apart from the generic requirements, which can be found in the CSD git repository *(“beoordelingscriteria.pdf” covering topics like input validation, an executable and ‘sounding’ application)*, the implementation should meet the following requirements:

- The application is a console application which …
    - SYNTH FOCUS - … demonstrates two customly designed and self-implemented synthesizers.
    - MELODY FOCUS - … demonstrates a customly designed and self-implemented melody generator with synthesized playback.
- OOP is correctly applied.
    - Oscillator base class with at least three derived classes.
- SYNTH FOCUS
    - Synth base class with at least two derived classes.
    - The two synthesizer designs have been approved by the course leader or practicum supervisor.
    - The user can choose between two synthesizers
    - Both synthesizer designs provide each at least one customizable parameter that alters the timbre of the synthesizer *(e.g. overtone ratio, fm modulation frequency deviation, oscillator(s) type) (thus frequency and amplitude are not viewed as customizable parameter)*. Noteworthy, the customizable parameters per synthesizer should differ.
    - The selected synthesizer plays a (hardcoded) melody, or MIDI / OSC is used to control the synthesizer.
- MELODY FOCUS

    - The melody generation strategy has been approved by the course leader or practicum supervisor.
    - The melody generation should provide the user with at least two customizable parameters that are of influence to its process and outcome.
    - The generated melody is stored in an array of Note objects, whereby Note is a class that at least has the properties to capture a midi note, a duration in quarter notes and velocity.
    - The melody is played by a simple customly designed synthesizer.

Assessment specific criteria

Apart from the generic assessment criteria, which can be found in the CSD git repository *("beoordelingscriteria.pdf" covering topics like clear naming and comments)* the assessment this course will cover:

- Presentation

- Presentation
    - Demonstration: at least two user flows with two different parameters, together with demonstration of properly handled invalid input
    - Chosen focus and the custom design
    - Reflection and take-aways

- Design documentation
    - Readability and clarity of the required documents

- Codebase
    - OOP
    - SYNTH FOCUS - Syntheze / DSP correctly implemented
    - MELODY FOCUS - Melody strategy correctly implemented
    - Code flow is concise; no redundant function calls / lines
    - *Good practice* regarding naming, comments, style,
    - UI (clear communication & validation)

- Reflection
    - Coverage of the required elements
    - Readability and clarity