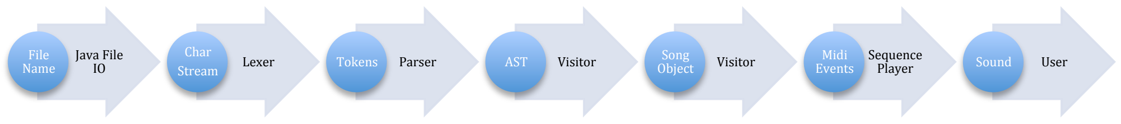
Project 1 ABC player

Overviews:

The Java file takes the filename of a ABC-music.abc file as an input. The Java Read class read the content into a String Object. Then the String is transfered to Lexer, which will create Tokens according to the grammar we provided. The parser parse the tokens, generating an abstract syntax tree afterwards. Then Visitor will process the AST into a Song object which can be interpreted by another Visitor. As the Song being processed, Notes will be added to SequencePlayer simultaneously. Finally, The SequencePlayer plays the music that User can hear.

The snapshot diagram as bellows:



Grammars & Abstract Syntax Tree :

Grammar has not been changed significantly. The AST was built based on the given BNF grammar file by using ANTLR.

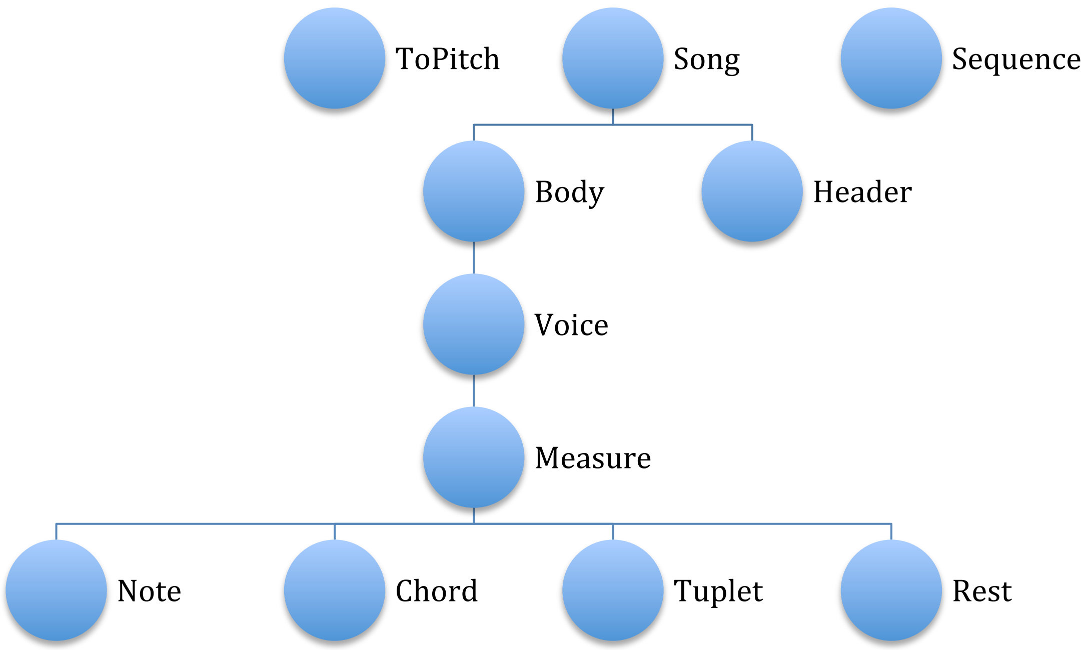
- We defined the implicit things in the given grammar ex, DIGIT
- We Changed LineFeed into End-Of-Line ::= newline | comment .

By translating the modified grammar into ANTLR to generate Parser and Lexer.

Traversing the AST to create Song object:

Once we have an AST, we can create a listener to traverse the AST to generate NOTE, CHORD, TUPLET, REST object which contain PITCH and DURATION, MEASURE, VOICE(List of Measures), MUSIC(List of VOICE) , HEADER then the SONG object which is the input of ToPitch, VIsitor class.

Abstract Syntax Tree:



Datatype definition:

Song(implement Sequence): represent a entire song, an output of a parser & an input of a visitor

- **Immutable**
- **Fields :**
  - **Body (List<Voice>)**
  - **Header**
- **Method :**
  - **Observor: getHeader(), gerBody();**

Sequence(Interface) : this represents any contiguous block of music which provides the method that invoke a visitor object to generate a sound that SequencePlayer can player

- **Method:**

Header: contains the fields which an ABC header store and help set the value of instance variables like NOTE, DURATION

- Immutable
- Fields :
  - tempo, title, index and etc;
- Methods:
  - getTempo, getMeter , getDefaultLength, getVoice, getKey

    - accept

Music (abstract class) : It represent the music parts of a song.

- Fields :
    - Pitch
    - Duration
- Method :
    -

Voice(implement Sequence) this contains a list of measures in a single voice

- **Immutable**
- **Fields :**
    - **List<Measure>**

Measure(implement Sequence): represent a repeat /Non-repeat measure and all the music inside it such as Single Note, Chord, Tuplet

- Immutable
- Fields : Repeat, NotRepeat
- Method :

Duration : represent the duration in terms of the music length

- Immutable
- Fields: startTickTIme (int), TickTIme(int)
- Method :
    - creator(startTickTIme (int), TickTIme(int))
    - observer : getStartTime, getTime
    - Producer: // calculate the LCM of all duration

Pitches:

#Already given in the original file

Note(implement Sequence): a note object which contains a single Note

- Immutable
- Fields:
    - Pitch
    - Accidentals (int)
    - Octave (int)

        - Duration

    Method : Creator(Pitches,Rest, Accidentals, Octave,)

Rest(implement Sequence):

- ■ Immutable
- ■ Fields:
    - Duration

    Method : Creator(Octave)

Chords (implement Sequence) : an object contains a list of Note object with same time duration

Immutable

Fields :

Notes   List<Notes>

Duration

Method : Creator: (SingleNotes, Duration..)

Observer: getNotesNum, getDuration, getType

Producer : changeTickTime()

Tuplet (implement Sequence): Tuplet is represent by a list of Note Object or Chord Object (due to Tuplet can contains chords)

- **Immutable**
- **Fields:**
    - ■ **List <Note> , List <chord>**

ToPitch : the visitor class that transverse the Song Object and add note to sequence player