

**TAB2XML**

Design Document

**Group 6**

**Elmira Onagh**

**Irsa Nasir**

**Long Lin**

**Harjap Randhawa**

**Daniel Di Giovanni**



Winter 2022

**Table of Content**

[**1.** **Introduction** 2](#_Toc100599834)

[**2.** **Visualization of Sheet Music** 2](#_Toc100599835)

[**1.1.1.** **Maintenance Scenario** 2](#_Toc100599836)

[**1.2.** **Instrument: *Guitar*** 3](#_Toc100599837)

[**1.2.1.** **Sequence Diagram** 4](#_Toc100599838)

[**1.2.2.** **UML Class Diagram** 5](#_Toc100599839)

[**1.3.** **Instrument: *Drum*** 6](#_Toc100599840)

[**1.3.1.** **Sequence Diagram** 6](#_Toc100599841)

[**1.3.2.** **UML Class Diagram** 7](#_Toc100599842)

[**1.4.** **Instrument: *Bass*** 8](#_Toc100599843)

[**1.4.1.** **UML Class Diagram** 8](#_Toc100599844)

[**1.4.2.** **Sequence Diagram** 9](#_Toc100599845)

[**3.** **Playing the tablature** 10](#_Toc100599846)

[**a.** **Activity Diagram** 10](#_Toc100599847)

[**2.2.** **UML class diagram: *Guitar*** 11](#_Toc100599848)

[**2.3.** **UML class diagram: *Drum*** 11](#_Toc100599849)

[**2.4** **UML class diagram: *Bass*** 12](#_Toc100599850)

[**4.** **Printing the Music sheet** 13](#_Toc100599851)

[**4.1.** **Sequence Diagram** 13](#_Toc100599852)

[**4.2.** **Activity Diagram** 14](#_Toc100599853)

[**5.** **Go to measure** 15](#_Toc100599854)

[**5.1.** **Activity Diagram** 15](#_Toc100599855)

# **Introduction**

TAB2XML is an application used to convert musical tablatures into MusicXML and then in turn, into music sheets. In this document, we will focus on the design of Previewing music Sheet and additional functionalities such as playing and printing music sheets. For this purpose, there are three main packages in our system: GUI, GUI.draw, and instrument. The GUI package contains the controller for the Preview music sheet button and MusicPlayer class used for playing the music. GUI.draw contains classes needed to display music sheet elements and Instrument packages contain classes corresponding to the supported tablatures (currently: Guitar, Drum, and Bass

# **Visualization of Sheet Music**

Based on the instrument specified in the input tablature, three different objects are created: *Guitar*, *Drum*, and *Bass*. In the following section, we look at different diagrams related to the creation and relationships of these classes (Figure 1). To allow our system to support tablatures for a new instrument, we can create a new class that instantiates an object corresponding to this new instrument.

#### **Maintenance Scenario**

For new elements to be displayed on the music sheet for the guitar tablature, we can either add them as a different method in one of the existing classes or create a new class and add a reference to it in the *Guitar* class. The initial point of drawing elements starts from *drawMeasureNotes ()* of the *Guitar* class and by following the path (many different methods are called in this method to display different elements) the new reference can be added in an appropriate place. To add new elements to existing classes we can simply create a new method in that class to support displaying this new element. For example, let us say we want to add a different type of clef. We can add a method in *DrawClef* class that instead of drawing “TAB,” creates this new type of clef.

The same thing can be done for Drum and Bass classes.

Diagram

Description automatically generated

**Figure 1.** Overall activity diagram of the visualizing a tablature as a music sheet.

### **Instrument: *Guitar***

If the input tablature is a guitar tablature, then a *Guitar* class is instantiated. The creation and displaying of the elements of the tablature in form of a music sheet are done through the *drawGuitar* method.

In section 1.1.1 the sequence diagrams depicting the sequence of events taken to visualize guitar notes can be found. The Diagrams are broken down into parts to ease the understanding and visualization.

In section 1.1.2 UML Class diagram is included to show the interactions between the *Guitar* class and other classes that result in displaying the musical elements on the screen.

#### **Sequence Diagram**

A screenshot of a computer

Description automatically generated with medium confidence

**Figure 2.** Overall Sequence diagram of displaying guitar tablature.

Graphical user interface

Description automatically generated

**Figure 3.** Sequence diagram describing the events taken in *drawGuitar* method of *Guitar* class.

#### **UML Class Diagram**

Graphical user interface

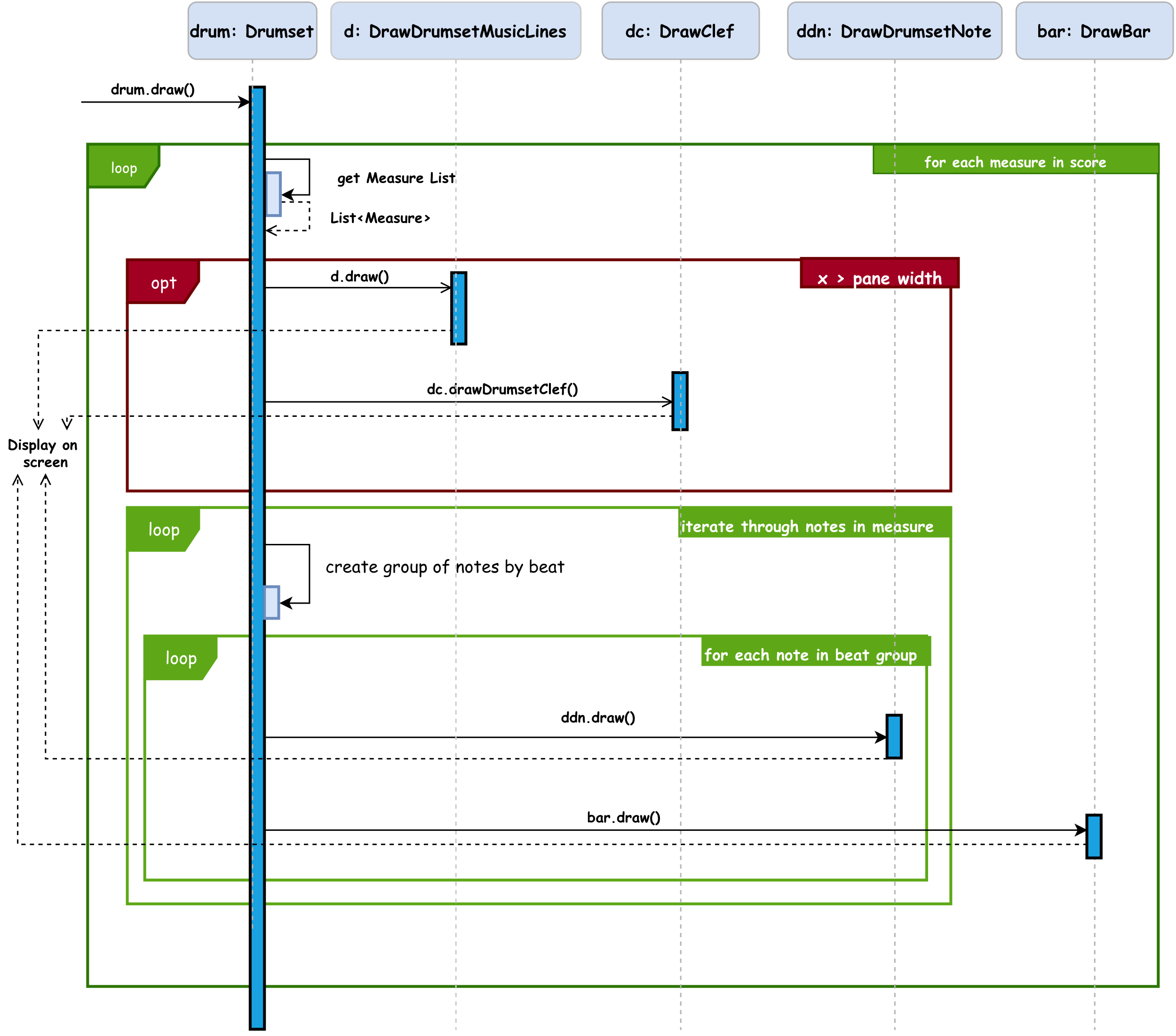
Description automatically generated

**Figure 4.** Class diagram of the *Guitar* class and its interactions. The green coloured *Guitar* class belongs to the *instrument* package while the blue classes belong to the *GUI.draw* package. The public, private, protected attributes and operations are denoted by “+,” “-,” and “#” respectively.

### **Instrument: *Drum***

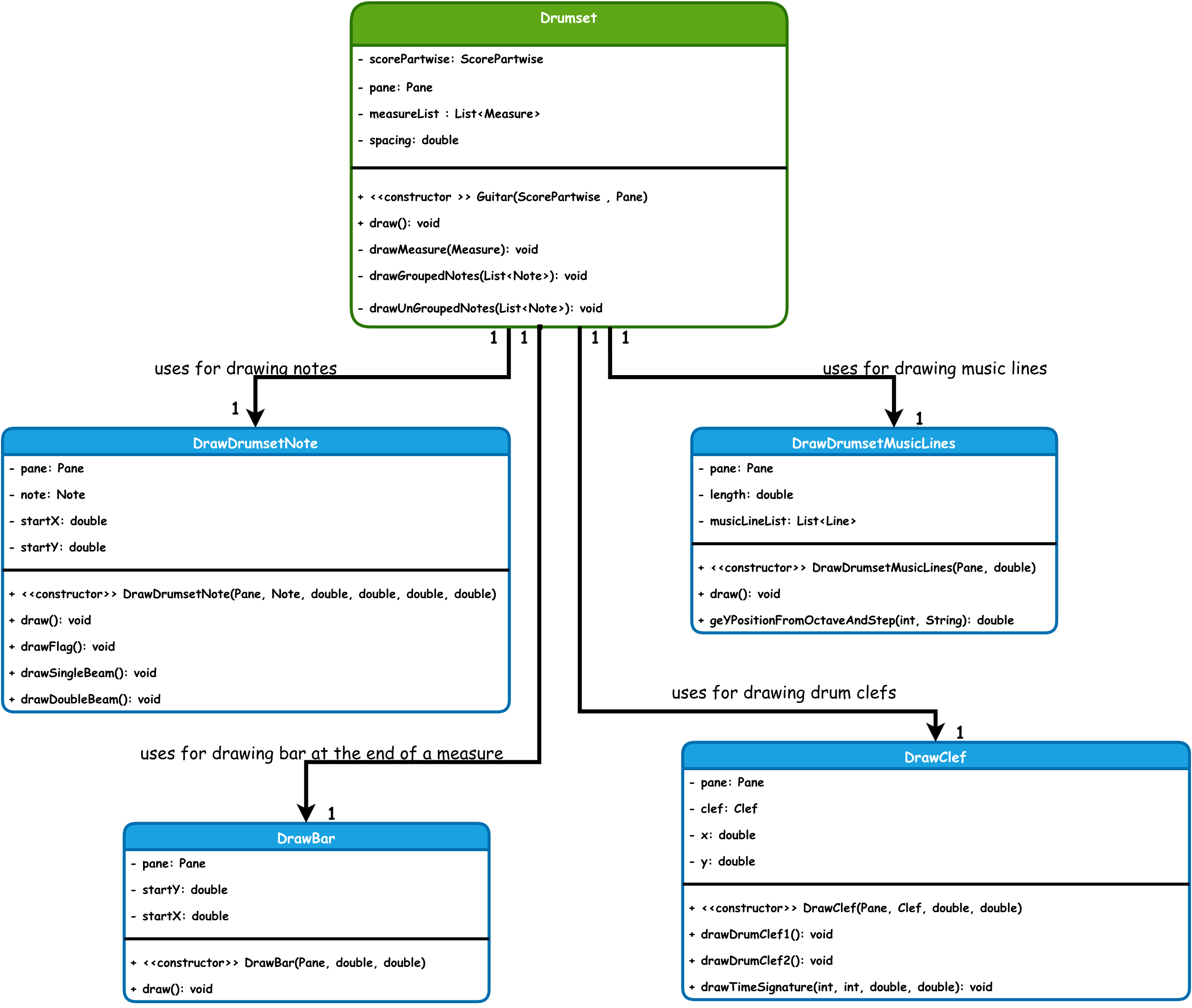
The following two diagrams are sequence and class diagrams for the *Drumset* class, showing its workflow and the other classes it uses/interacts with.

#### **Sequence Diagram**



**Figure 5**. Sequence diagram of the *Drumset* class and its interactions with the classes *DrawDrumsetMusicLines,* *DrawClef*, *DrawDrumsetNote*, and *DrawBar*.

#### **UML Class Diagram**



**Figure 6.** Class Diagram of the *Drumset* class and the classes it uses.

### **Instrument: *Bass***

#### **UML Class Diagram**

**A picture containing text

Description automatically generated**

**Figure 7.** Class diagram indicating the attributes and methods of *Bass* class and classes that it interacts with

#### **Sequence Diagram**

Diagram, timeline

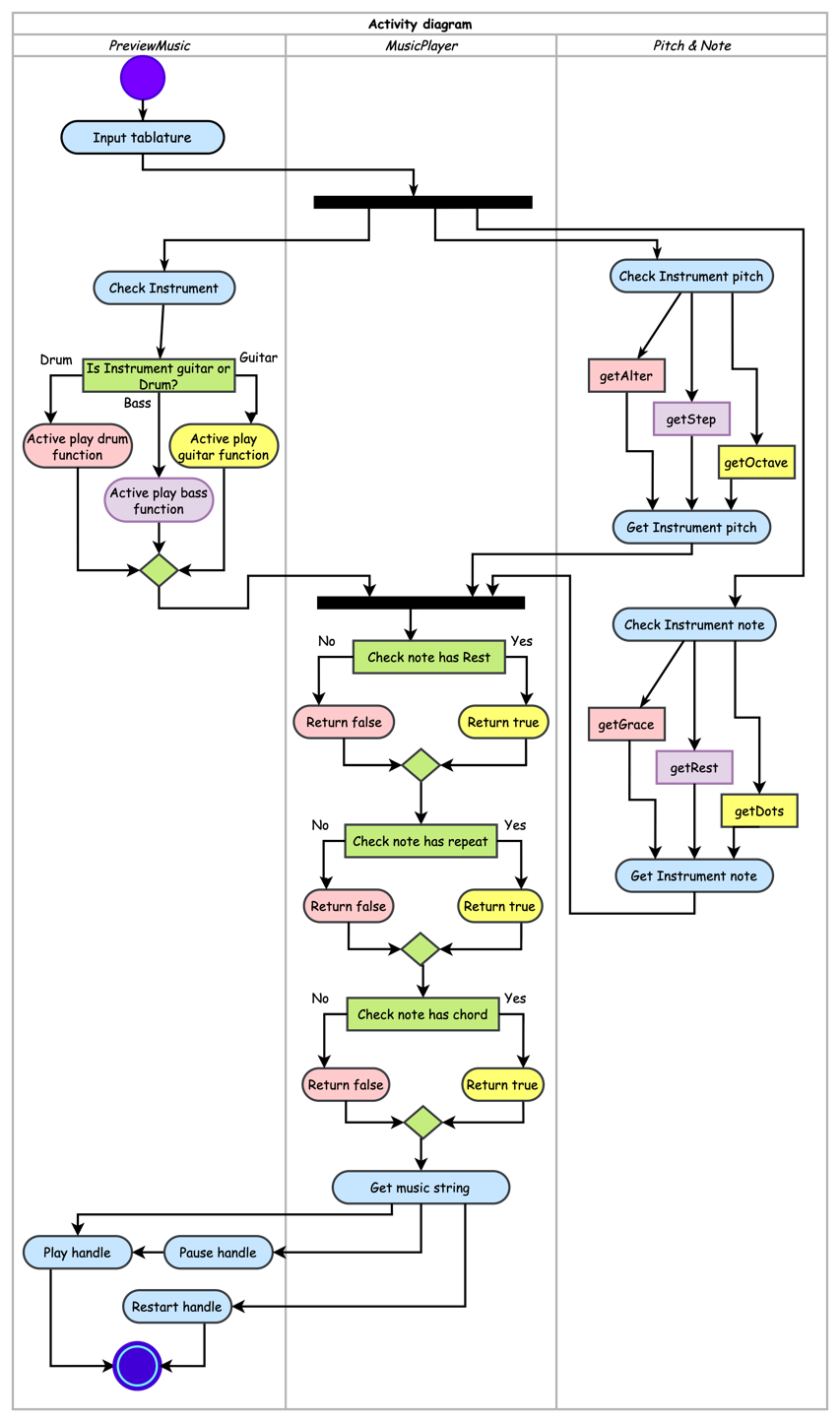
Description automatically generated with medium confidence

**Figure 8.** Sequence diagram of the for the Bass tablature

# **Playing the tablature**

In this section, we will cover the design diagrams related to the Play functionality of the system. In section 2.1 an Activity diagram describing the overall events taken to play the notes is included. In the following sections, UML class diagrams depicting the specific methods of *MusicPlayer* class used for each instrument can be found.

### **Activity Diagram**

1. 

**Figure 8**. Activity diagram of play music note function for guitar and drum

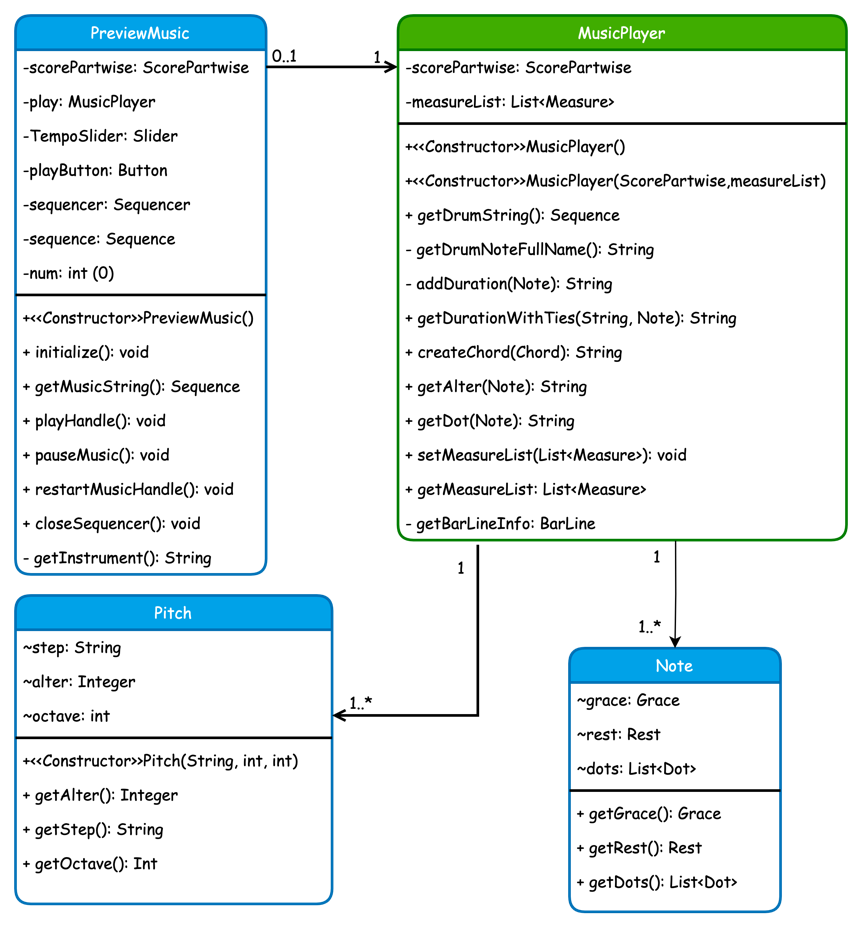
### **UML class diagram: *Guitar***

A screenshot of a cell phone

Description automatically generated with medium confidence

**Figure 9**. The public, private, protected attributes and operations are denoted by “+,” “-,” and “~” respectively.

### **UML class diagram: *Drum***

1. 

**Figure 10**. The public, private, protected attributes and operations are denoted by “+,” “-,” and “~” respectively.

### **UML class diagram: *Bass***

A screenshot of a cell phone

Description automatically generated with medium confidence

**Figure 11**. The public, private, protected attributes and operations are denoted by “+,” “-,” and “~” respectively.

# **Printing the Music sheet**

In this section, we will discuss the sequence and activity diagrams describing the events taken to print/ save the music sheet from the moment that the user clicks on the print button on the preview window.

### **Sequence Diagram**

Diagram

Description automatically generated

**Figure 12**. Sequence diagram for the print function.

### **Activity Diagram**

Diagram

Description automatically generated

**Figure 13.** Activity diagram for the print function.

# **Go to measure**

This section includes an activity diagram describing the event taken from when a user enters a specific measure number to highlight the measure.

### **Activity Diagram**

A picture containing diagram

Description automatically generated

**Figure 14.** Activity Diagram showing the events that result in highlighting a measure.

**Group 6**

**Elmira Onagh**

**Irsa Nasir**

**Long Lin**

**Harjap Randhawa**

**Daniel Di Giovanni**