TAB TO MUSICXML CONVERTER AND PLAYER

Design Document

EECS 2311 — Group 8

ATTRIBUTION

Parth Sharma
Mohammad Khan
Greatlove Bariboloka
Kedamawi Mengitsu
Sina Aligholizadeh

Winter 2022

PURPOSE

This document will describe the design of the MusicXML converter application including how relevant classes interact with one another and the runtime behaviour of the application. The intended audience for this document is programmers with an understanding of OOP concepts.

INTRODUCTION

The section of the MusicXML converter that will be focused on in this paper is the part that converts MusicXML to sheet music and allows the user to play it.

PLAYING MUSIC

1

Title: Class diagram for playing music, figure 1

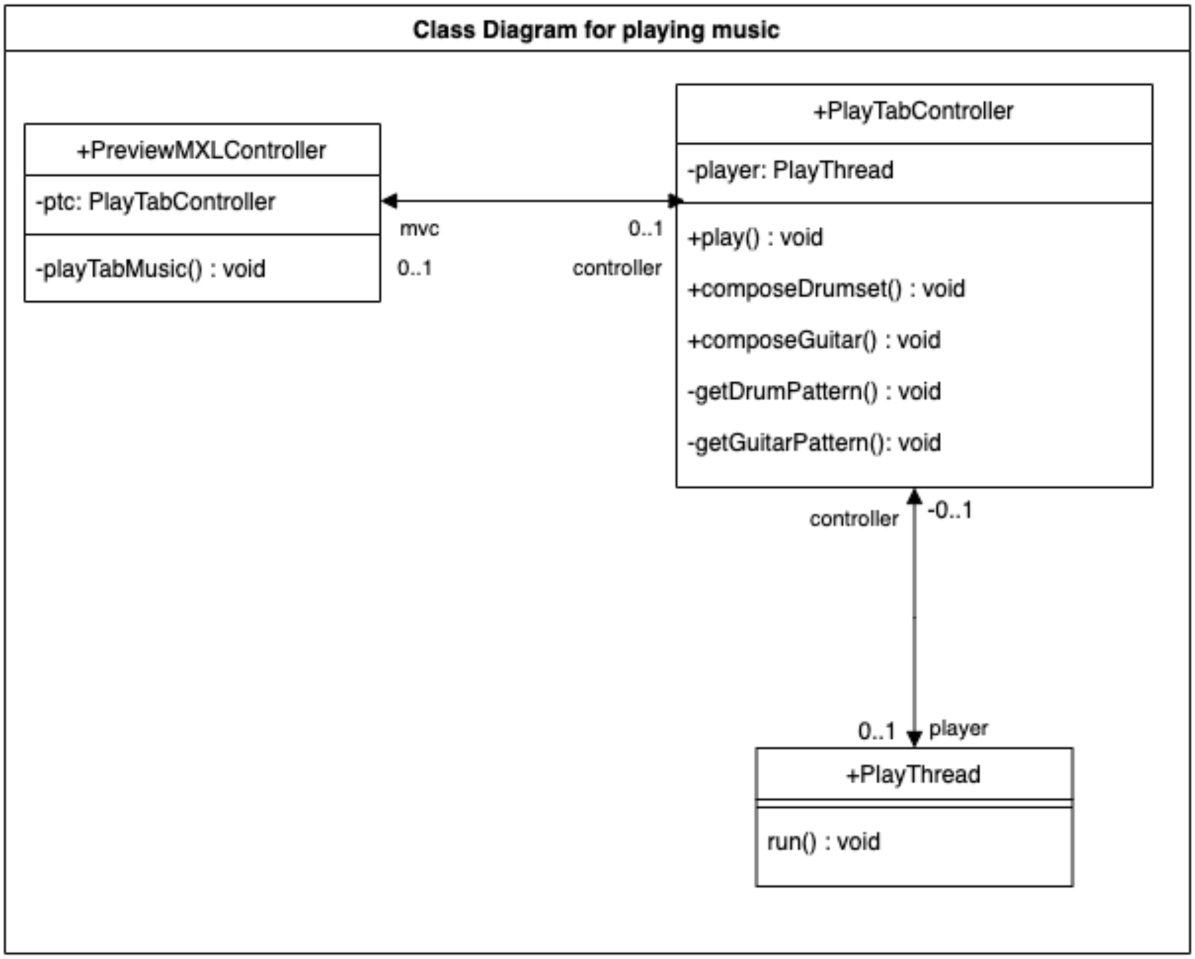

Figure 1: Class Diagram for playing music

Perspective: Main Application

2

Title: Sequence diagram for playing music, fig 2

Perspective: Application

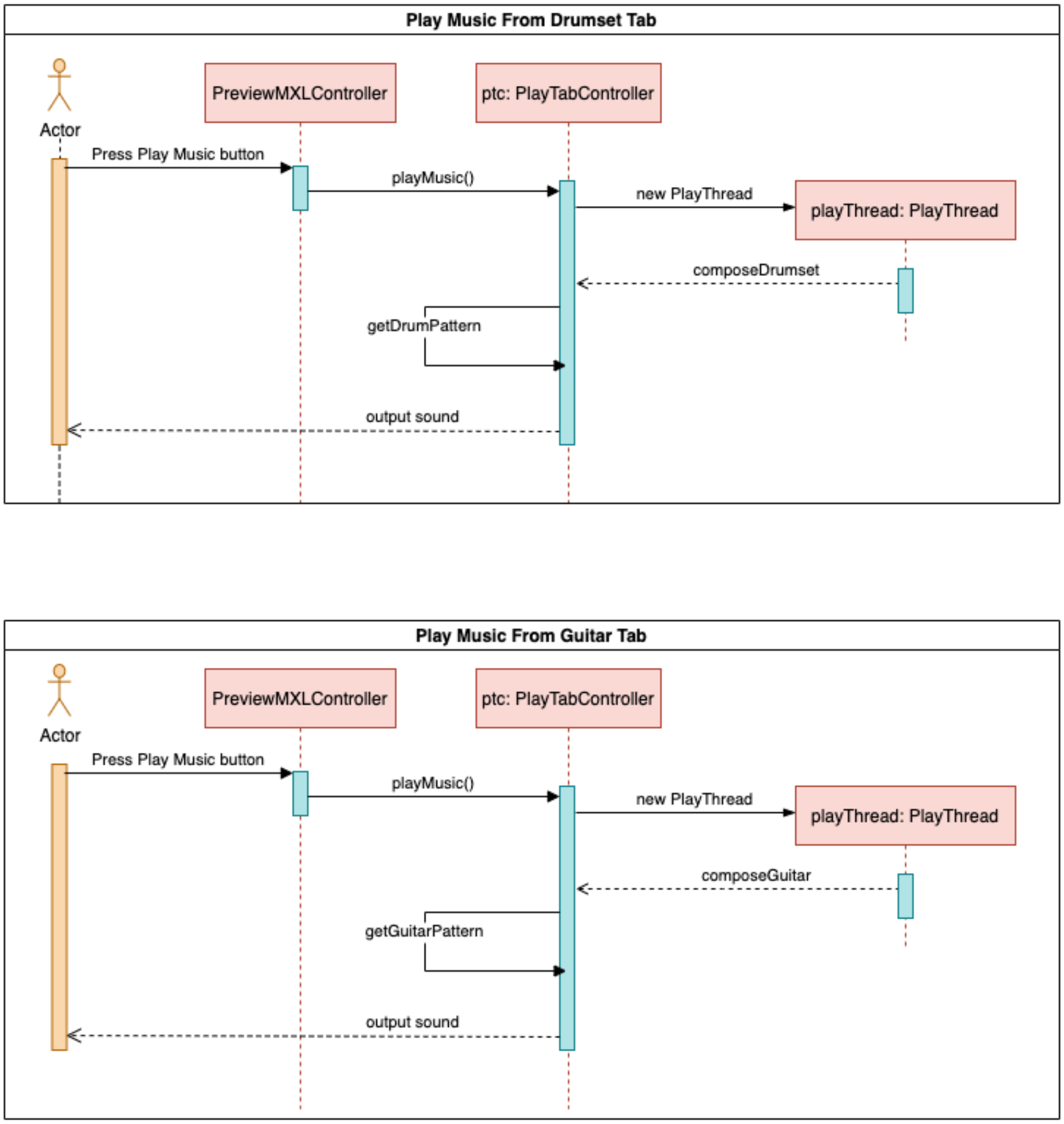

Figure 2: Sequence diagrams for playing music (drums and guitar/bass)

3

Title: Sequence diagram for playing drum tabs, fig 3

Perspective: Inside getDrumPattern method

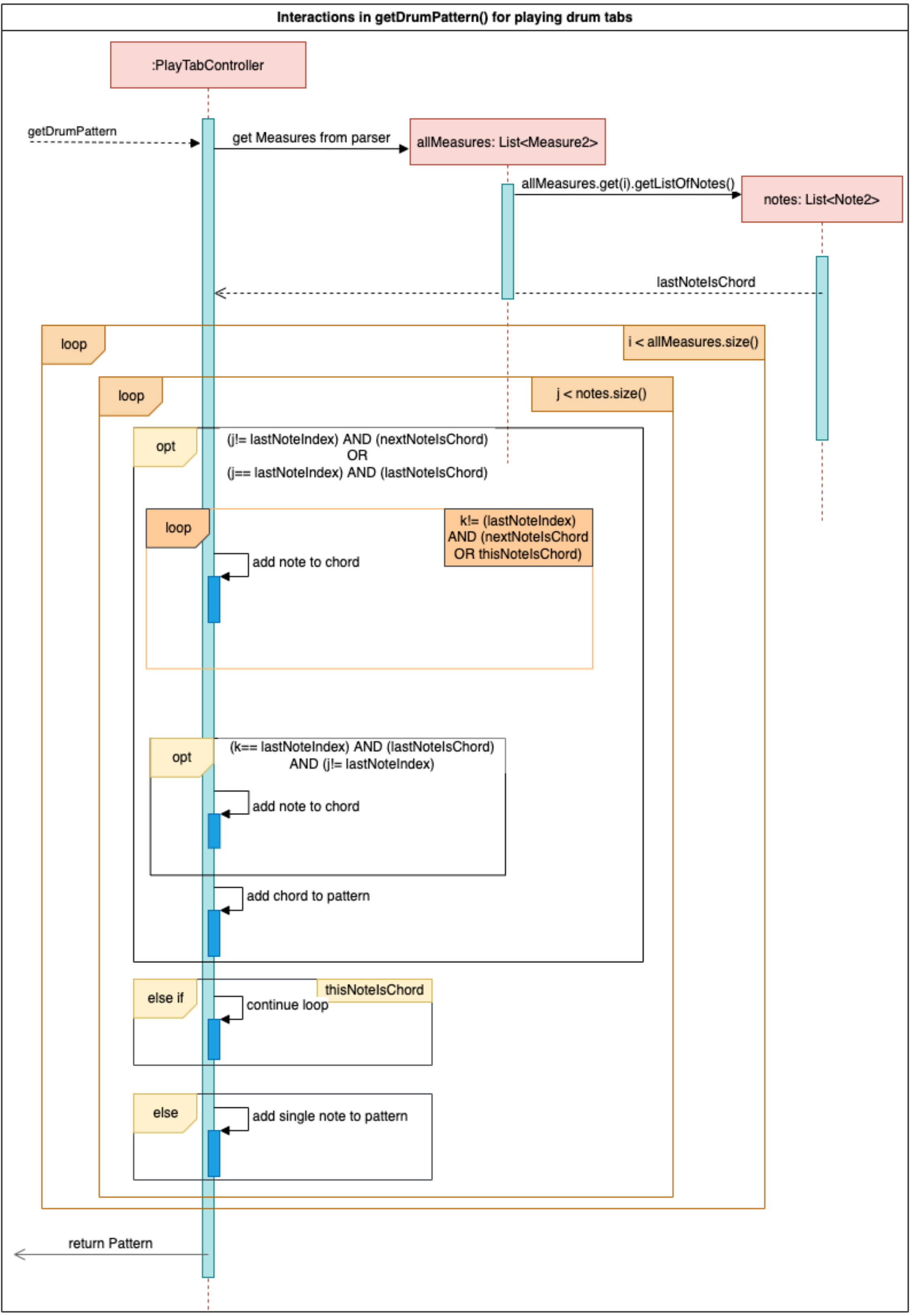

Figure 3: Sequence diagram for getDrumPattern()

4

Title: Activity diagram for playing sheet music, fig 4

Perspective: Main Application

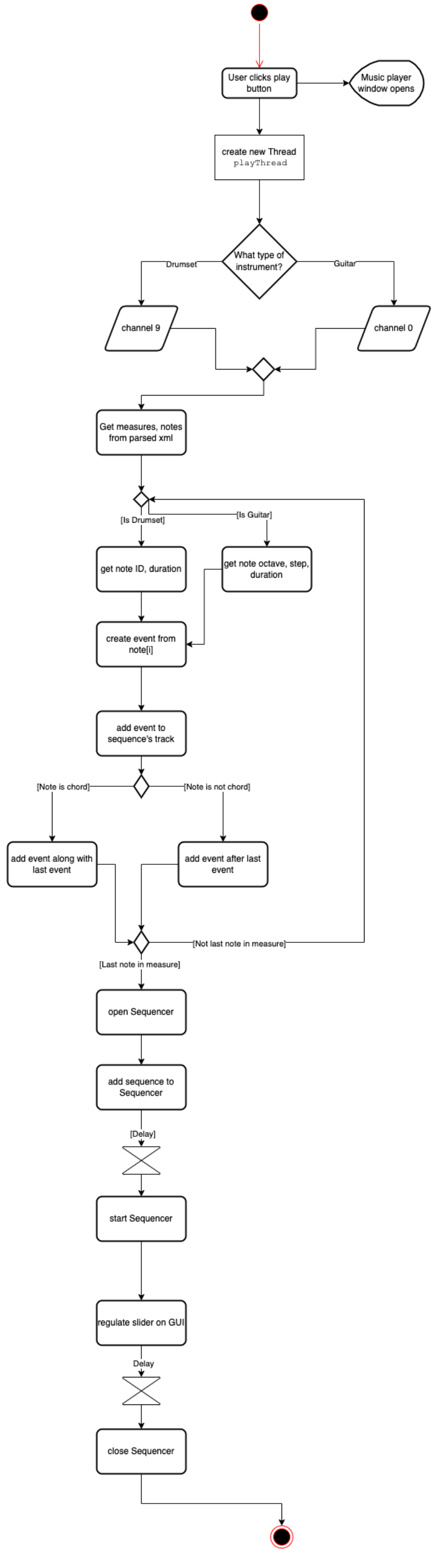
User clicks play button
Music player window opens
create new Thread playThread
What type of instrument?
Drumset
Guitar
channel 9
channel 0
Get measures, notes from parsed xml
[Is Drumset]
[Is Guitar]
get note ID, duration
get note octave, step, duration
create event from note[i]
add event to sequence's track
[Note is chord]
[Note is not chord]
add event along with last event
add event after last event
[Not last note in measure]
[Last note in measure]
open Sequencer
add sequence to Sequencer
[Delay]
start Sequencer
regulate slider on GUI
Delay
close Sequencer

Figure 4: Activity diagram for playing sheet music (Main App)

5

Title: Sequence diagram for playing guitar tabs, fig 5

Perspective: Inside getGuitarPattern() method

DISPLAYING SHEET MUSIC

1

Title: Class diagram for displaying sheet music, fig 6

Perspective: Main Application

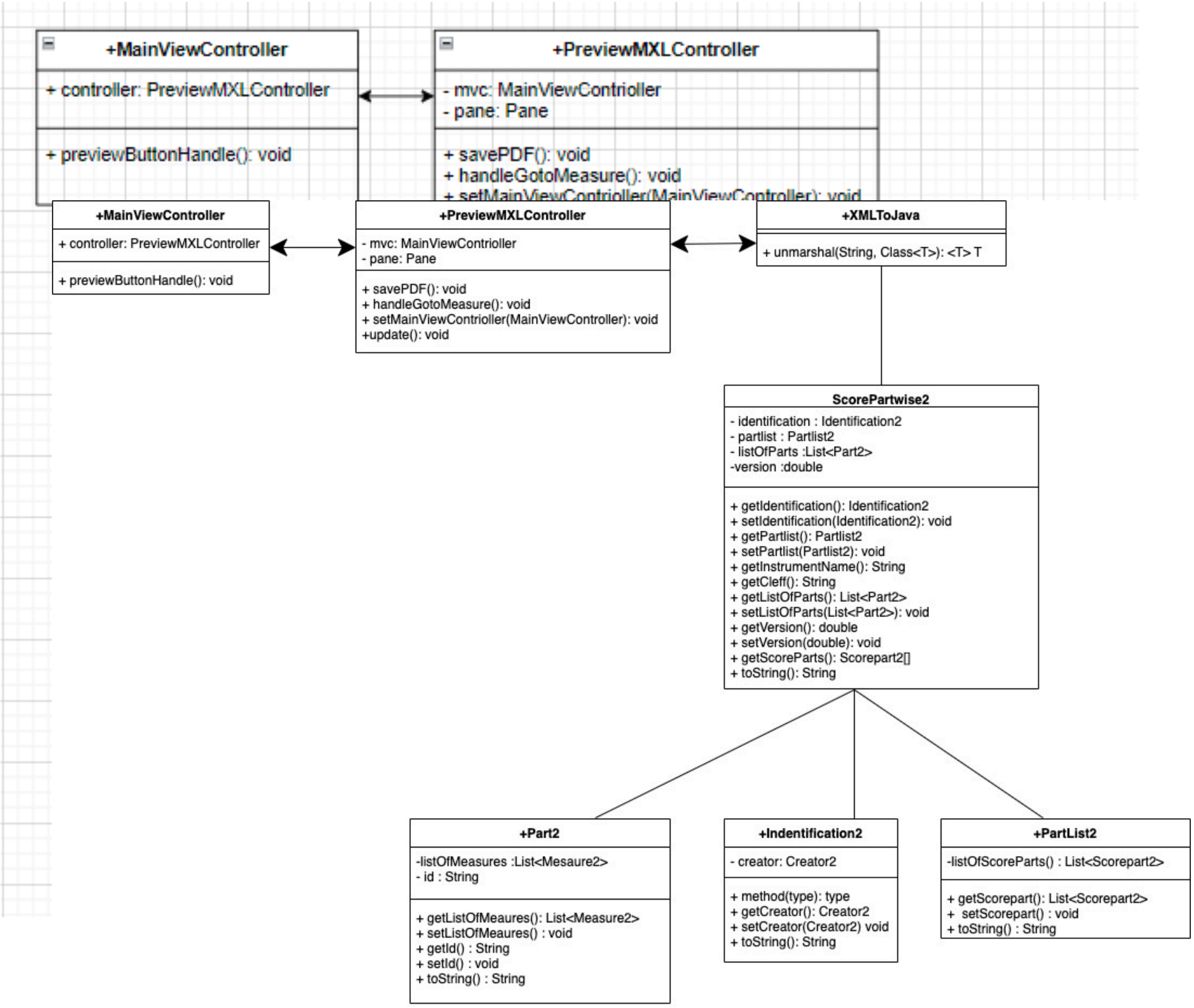
+MainViewController
+ controller: PreviewMXLController
+ previewButtonHandle(): void
+PreviewMXLController
- mvc: MainViewContrioller
- pane: Pane
+ savePDF(): void
+ handleGotoMeasure(): void
+ setMainViewContrioller(MainViewController): void
+MainViewController
+ controller: PreviewMXLController
+ previewButtonHandle(): void
+PreviewMXLController
- mvc: MainViewContrioller
- pane: Pane
+ savePDF(): void
+ handleGotoMeasure(): void
+ setMainViewContrioller(MainViewController): void
+update(): void
+XMLToJava
+ unmarshal(String, Class<T>): <T> T
ScorePartwise2
- identification : Identification2
- partlist : Partlist2
- listOfParts :List<Part2>
-version :double
+ getIdentification(): Identification2
+ setIdentification(Identification2): void
+ getPartlist(): Partlist2
+ setPartlist(Partlist2): void
+ getInstrumentName(): String
+ getCleff(): String
+ getListOfParts(): List<Part2>
+ setListOfParts(List<Part2>): void
+ getVersion(): double
+ setVersion(double): void
+ getScoreParts(): Scorepart2[]
+ toString(): String
+Part2
-listOfMeasures :List<Mesaure2>
- id : String
+ getListOfMeaures(): List<Measure2>
+ setListOfMeaures() : void
+ getId() : String
+ setId() : void
+ toString() : String
+Indentification2
- creator: Creator2
+ method(type): type
+ getCreator(): Creator2
+ setCreator(Creator2) void
+ toString(): String
+PartList2
-listOfScoreParts() : List<Scorepart2>
+ getScorepart(): List<Scorepart2>
+ setScorepart() : void
+ toString() : String

Figure 6: Class Diagram for the whole parser

2

Title: Class diagram for displaying guitar music notation, fig 7

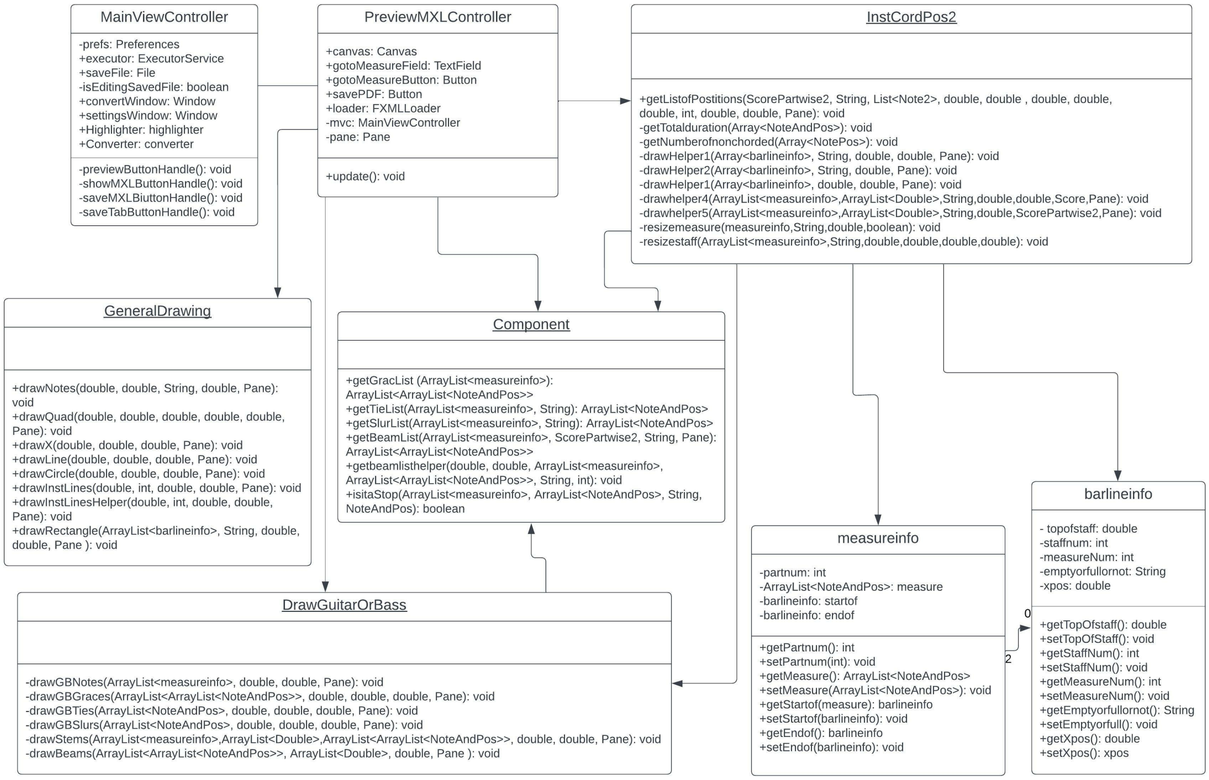
MainViewController
-prefs: Preferences
+executor: ExecutorService
+saveFile: File
-isEditingSavedFile: boolean
+convertWindow: Window
+settingsWindow: Window
+Highlighter: highlighter
+Converter: converter
-previewButtonHandle(): void
-showMXLButtonHandle(): void
-saveMXLBiuttonHandle(): void
-saveTabButtonHandle(): void
PreviewMXLController
+canvas: Canvas
+gotoMeasureField: TextField
+gotoMeasureButton: Button
+savePDF: Button
+loader: FXMLLoader
-mvc: MainViewController
-pane: Pane
+update(): void
InstCordPos2
+getListofPostitions(ScorePartwise2, String, List<Note2>, double, double , double, double, double, int, double, double, Pane): void
-getTotalduration(Array<NoteAndPos>): void
-getNumberofnonchorded(Array<NotePos>): void
-drawHelper1(Array<barlineinfo>, String, double, double, Pane): void
-drawHelper2(Array<barlineinfo>, String, double, Pane): void
-drawHelper1(Array<barlineinfo>, double, double, Pane): void
-drawhelper4(ArrayList<measureinfo>,ArrayList<Double>,String,double,double,Score,Pane): void
-drawhelper5(ArrayList<measureinfo>,ArrayList<Double>,String,double,ScorePartwise2,Pane): void
-resizemeasure(measureinfo,String,double,boolean): void
-resizestaff(ArrayList<measureinfo>,String,double,double,double,double): void
GeneralDrawing
+drawNotes(double, double, String, double, Pane): void
+drawQuad(double, double, double, double, double, Pane): void
+drawX(double, double, double, Pane): void
+drawLine(double, double, double, Pane): void
+drawCircle(double, double, double, Pane): void
+drawInstLines(double, int, double, double, Pane): void
+drawInstLinesHelper(double, int, double, double, Pane): void
+drawRectangle(ArrayList<barlineinfo>, String, double, double, Pane): void
Component
+getGracList (ArrayList<measureinfo>): ArrayList<ArrayList<NoteAndPos>>
+getTieList(ArrayList<measureinfo>, String): ArrayList<NoteAndPos>
+getSlurList(ArrayList<measureinfo>, String): ArrayList<NoteAndPos>
+getBeamList(ArrayList<measureinfo>, ScorePartwise2, String, Pane): ArrayList<ArrayList<NoteAndPos>>
+getbeamlisthelper(double, double, ArrayList<measureinfo>, ArrayList<ArrayList<NoteAndPos>>, String, int): void
+isitaStop(ArrayList<measureinfo>, ArrayList<NoteAndPos>, String, NoteAndPos): boolean
barlineinfo
- topofstaff: double
-staffnum: int
-measureNum: int
-emptyorfullornot: String
-xpos: double
+getTopOfstaff(): double
+setTopOfStaff(): void
+getStaffNum(): int
+setStaffNum(): void
+getMeasureNum(): int
+setMeasureNum(): void
+getEmptyorfullornot(): String
+setEmptyorfull(): void
+getXpos(): double
+setXpos(): xpos
measureinfo
-partnum: int
-ArrayList<NoteAndPos>: measure
-barlineinfo: startof
-barlineinfo: endof
+getPartnum(): int
+setPartnum(int): void
+getMeasure(): ArrayList<NoteAndPos>
+setMeasure(ArrayList<NoteAndPos>): void
+getStartof(measure): barlineinfo
+setStartof(barlineinfo): void
+getEndof(): barlineinfo
+setEndof(barlineinfo): void
0
2
DrawGuitarOrBass
-drawGBNotes(ArrayList<measureinfo>, double, double, Pane): void
-drawGBGraces(ArrayList<ArrayList<NoteAndPos>>, double, double, double, Pane): void
-drawGBTies(ArrayList<NoteAndPos>, double, double, double, Pane): void
-drawGBSlurs(ArrayList<NoteAndPos>, double, double, double, Pane): void
-drawStems(ArrayList<measureinfo>,ArrayList<Double>,ArrayList<ArrayList<NoteAndPos>>, double, double, Pane): void
-drawBeams(ArrayList<ArrayList<NoteAndPos>>, ArrayList<Double>, double, Pane): void

Figure 7: Class diagram for displaying guitar music notation

3

Title: Class diagram for displaying drums music notation, fig 8

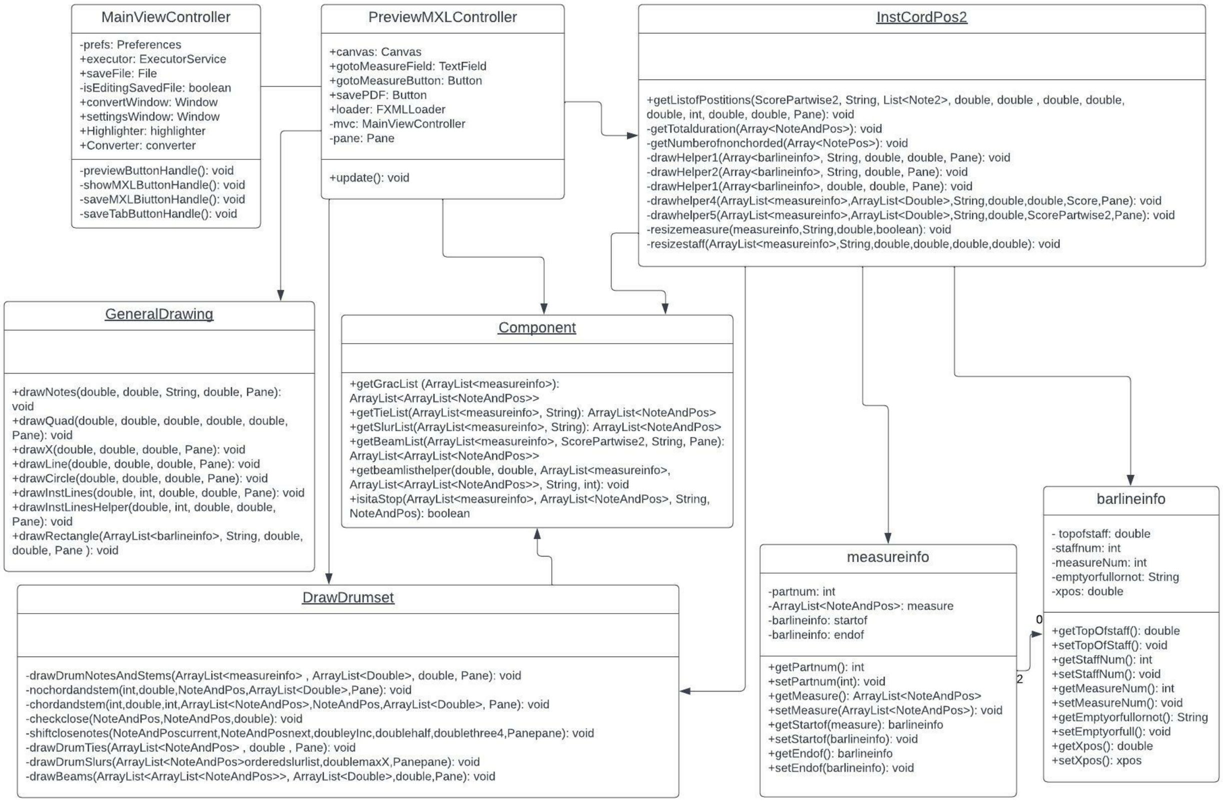
MainViewController
-prefs: Preferences
+executor: ExecutorService
+saveFile: File
-isEditingSavedFile: boolean
+convertWindow: Window
+settingsWindow: Window
+Highlighter: highlighter
+Converter: converter
-previewButtonHandle(): void
-showMXLButtonHandle(): void
-saveMXLBiuttonHandle(): void
-saveTabButtonHandle(): void
PreviewMXLController
+canvas: Canvas
+gotoMeasureField: TextField
+gotoMeasureButton: Button
+savePDF: Button
+loader: FXMLLoader
-mvc: MainViewController
-pane: Pane
+update(): void
InstCordPos2
+getListofPostitions(ScorePartwise2, String, List<Note2>, double, double , double, double, double, int, double, double, Pane): void
-getTotalduration(Array<NoteAndPos>): void
-getNumberofnonchorded(Array<NotePos>): void
-drawHelper1(Array<barlineinfo>, String, double, double, Pane): void
-drawHelper2(Array<barlineinfo>, String, double, Pane): void
-drawHelper1(Array<barlineinfo>, double, double, Pane): void
-drawhelper4(ArrayList<measureinfo>,ArrayList<Double>,String,double,double,Score,Pane): void
-drawhelper5(ArrayList<measureinfo>,ArrayList<Double>,String,double,ScorePartwise2,Pane): void
-resizemeasure(measureinfo,String,double,boolean): void
-resizestaff(ArrayList<measureinfo>,String,double,double,double,double): void
GeneralDrawing
+drawNotes(double, double, String, double, Pane): void
+drawQuad(double, double, double, double, double, Pane): void
+drawX(double, double, double, Pane): void
+drawLine(double, double, double, Pane): void
+drawCircle(double, double, double, Pane): void
+drawInstLines(double, int, double, double, Pane): void
+drawInstLinesHelper(double, int, double, double, Pane): void
+drawRectangle(ArrayList<barlineinfo>, String, double, double, Pane): void
Component
+getGracList (ArrayList<measureinfo>): ArrayList<ArrayList<NoteAndPos>>
+getTieList(ArrayList<measureinfo>, String): ArrayList<NoteAndPos>
+getSlurList(ArrayList<measureinfo>, String): ArrayList<NoteAndPos>
+getBeamList(ArrayList<measureinfo>, ScorePartwise2, String, Pane): ArrayList<ArrayList<NoteAndPos>>
+getbeamlisthelper(double, double, ArrayList<measureinfo>, ArrayList<ArrayList<NoteAndPos>>, String, int): void
+isitaStop(ArrayList<measureinfo>, ArrayList<NoteAndPos>, String, NoteAndPos): boolean
barlineinfo
- topofstaff: double
-staffnum: int
-measureNum: int
-emptyorfullornot: String
-xpos: double
+getTopOfstaff(): double
+setTopOfStaff(): void
+getStaffNum(): int
+setStaffNum(): void
+getMeasureNum(): int
+setMeasureNum(): void
+getEmptyorfullornot(): String
+setEmptyorfull(): void
+getXpos(): double
+setXpos(): xpos
measureinfo
-partnum: int
-ArrayList<NoteAndPos>: measure
-barlineinfo: startof
-barlineinfo: endof
+getPartnum(): int
+setPartnum(int): void
+getMeasure(): ArrayList<NoteAndPos>
+setMeasure(ArrayList<NoteAndPos>): void
+getStartof(measure): barlineinfo
+setStartof(barlineinfo): void
+getEndof(): barlineinfo
+setEndof(barlineinfo): void
0
2
DrawDrumset
-drawDrumNotesAndStems(ArrayList<measureinfo> , ArrayList<Double>, double, Pane): void
-nochordandstem(int,double,NoteAndPos,ArrayList<Double>,Pane): void
-chordandstem(int,double,int,ArrayList<NoteAndPos>,NoteAndPos,ArrayList<Double>, Pane): void
-checkclose(NoteAndPos,NoteAndPos,double): void
-shiftclosenotes(NoteAndPoscurrent,NoteAndPosnext,doubleyInc,doublehalf,doublethree4,Panepane): void
-drawDrumTies(ArrayList<NoteAndPos> , double , Pane): void
-drawDrumSlurs(ArrayList<NoteAndPos>orderedslurlist,doublemaxX,Panepane): void
-drawBeams(ArrayList<ArrayList<NoteAndPos>>, ArrayList<Double>,double,Pane): void

Figure 8: Class Diagram for displaying drums music notation

4

Title: Sequence diagram for displaying music notation, fig 9

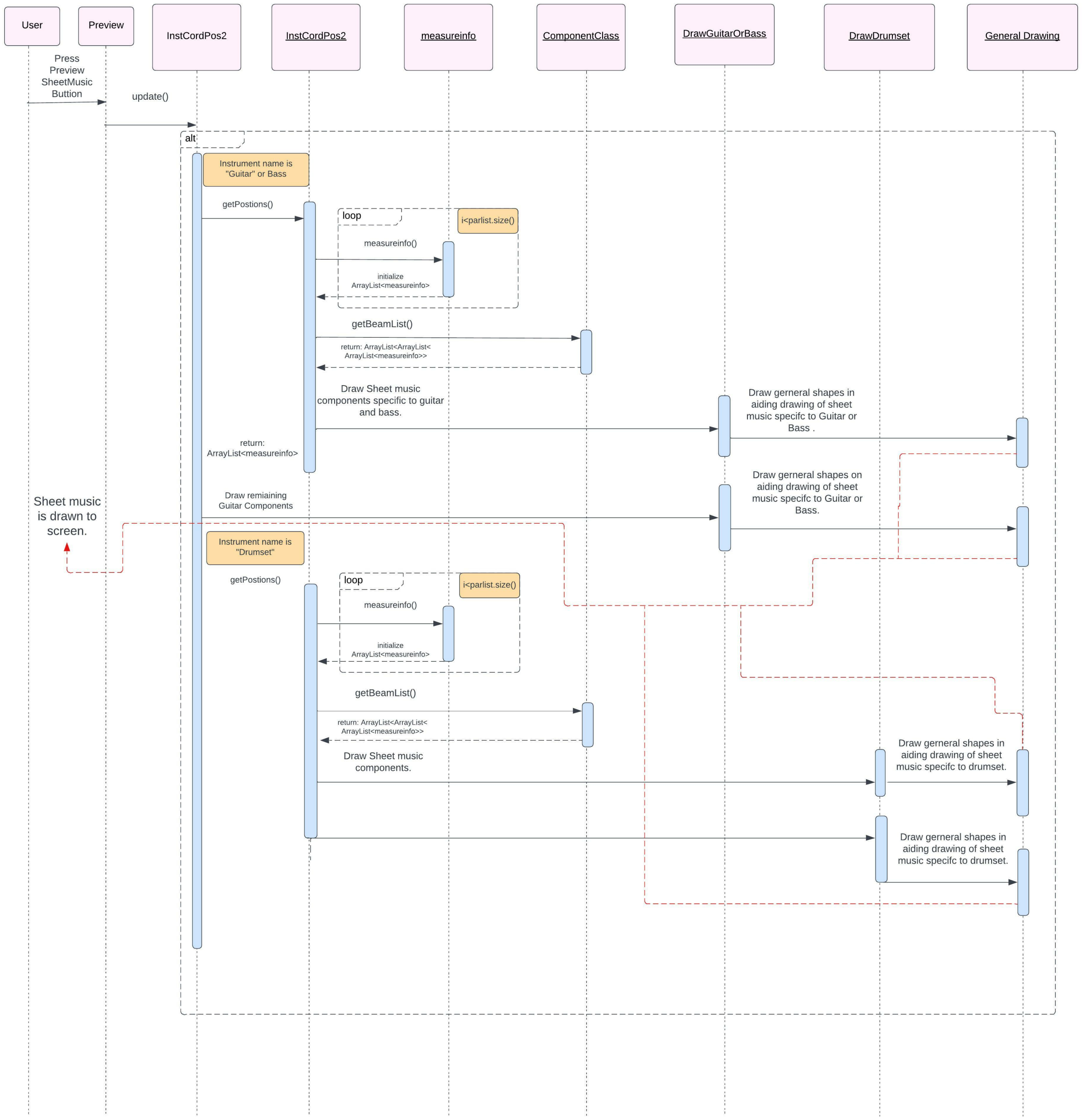
User
Preview
InstCordPos2
InstCordPos2
measureinfo
ComponentClass
DrawGuitarOrBass
DrawDrumset
General Drawing
Press Preview SheetMusic Buttion
update()
alt
Instrument name is "Guitar" or Bass
getPostions()
loop
i<parlist.size()
measureinfo()
initialize ArrayList<measureinfo>
getBeamList()
return: ArrayList<ArrayList< ArrayList<measureinfo>>
Draw Sheet music components specific to guitar and bass.
Draw gerneral shapes in aiding drawing of sheet music specifc to Guitar or Bass .
return: ArrayList<measureinfo>
Draw gerneral shapes on aiding drawing of sheet music specifc to Guitar or Bass.
Draw remiaining Guitar Components
Sheet music is drawn to screen.
Instrument name is "Drumset"
getPostions()
loop
i<parlist.size()
measureinfo()
initialize ArrayList<measureinfo>
getBeamList()
return: ArrayList<ArrayList< ArrayList<measureinfo>>
Draw Sheet music components.
Draw gerneral shapes in aiding drawing of sheet music specifc to drumset.
Draw gerneral shapes in aiding drawing of sheet music specifc to drumset.

Figure 9: Sequence diagram for displaying music notation

5

Title: Activity diagram for displaying music notation, fig 10

6

Title: Sequence diagram for printing music notation, fig 11

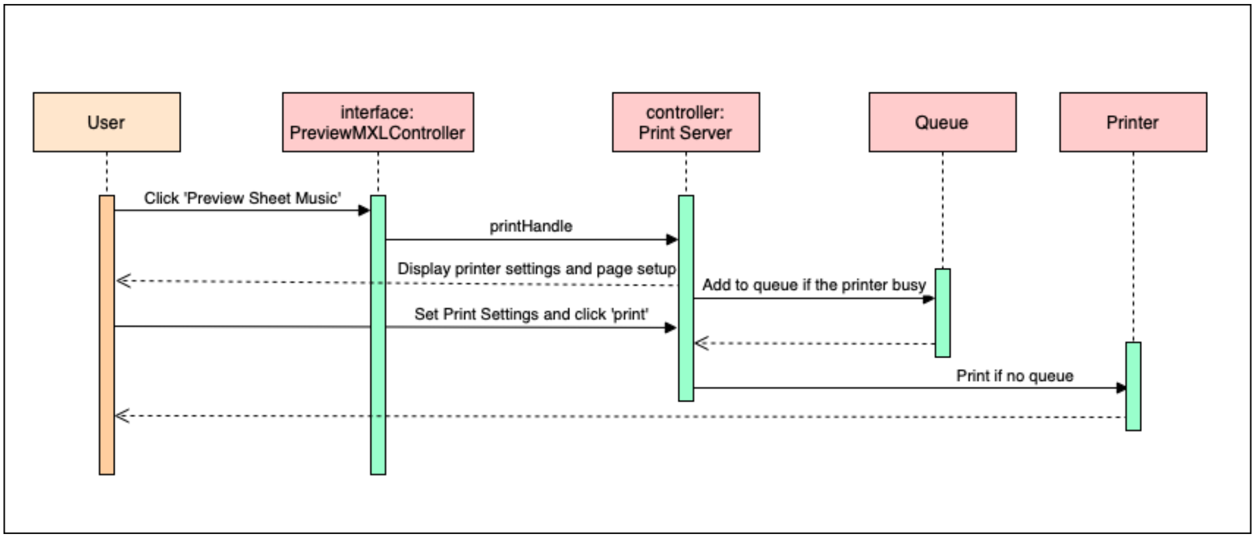
User
interface: PreviewMXLController
controller: Print Server
Queue
Printer
Click 'Preview Sheet Music'
printHandle
Display printer settings and page setup
Add to queue if the printer busy
Set Print Settings and click 'print'
Print if no queue

Figure 11: Sequence diagram for printing music notation

7

Title: Activity diagram for printing music notation, fig 12

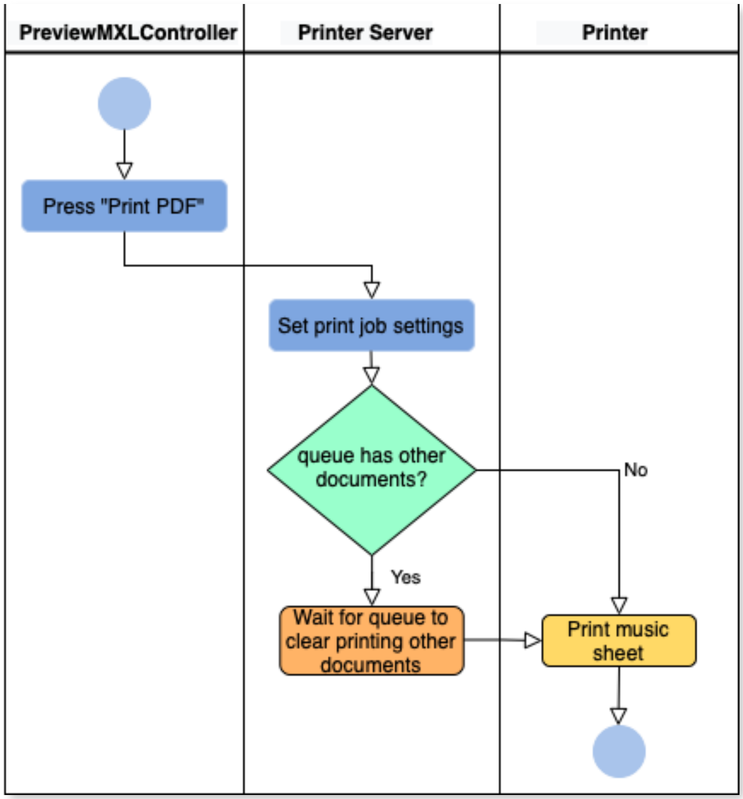

Figure 12: activity diagram for printing music notation

MAINTENANCE SCENARIOS

1. Supporting new Fugue Releases
    1.1. The jfugue library we are using goes obsolete: adjust the playing functionality accordingly; take advantage of the new features and upgrade app
    1.2. Jfugue is used to create patterns and play so modify the Pattern and Player instances in PlayTabController
2. To maintain app relevance.
    2.1. Includes adding playing and displaying functionality for other instruments like piano and violin
    2.2. Add in conditions in the PlayThread class for the name of the instrument
    2.3. Add composeInstrument() and getInstrumentPattern() methods, where Instrument is the name of the instrument being added
    2.4. Add needed information about the notes to the pattern in the getInstrumentPattern() method
3. Bugs reported by users
    3.1. Investigate the source of the bug, remake it and change relevant methods