Testing Document

EECS 2311 | Software Development Project

Group 4:

Walid AlDari
Hoshner Tavadia
Shaharyar Choudhry
Matthew Patrus

“The document describes the test cases run, the way they were derived, and the way they were implemented. Discussion on why this should be sufficient testing is included. Professional presentation.”

Music Player Test Cases

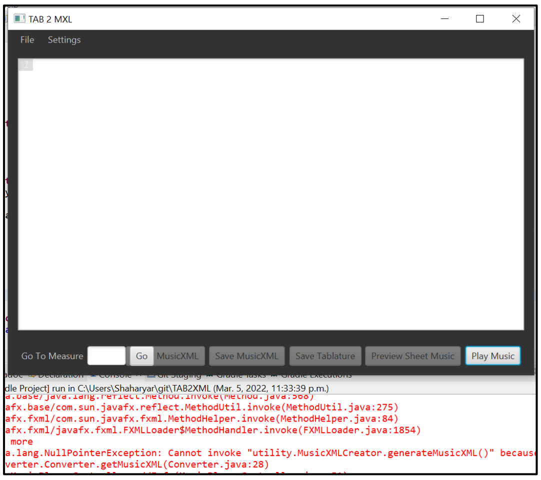
1) **Test Case 1**: Test to see if the “Play Music” button works when an empty output in entered

How was this derived: Initially, the “play music” button was able to be pressed even if the user entered no input. Therefore, clicking the play music button would create a null pointer exception. We fixed this problem by only allowing the user to play music once the user has entered a reasonable input

How this was tested: We ran the code with an empty input and clicked the “Play Music” button. We were expecting a warning message to pop up with a message saying “Please enter valid music tab”. We then compared the outputs with what we were looking for

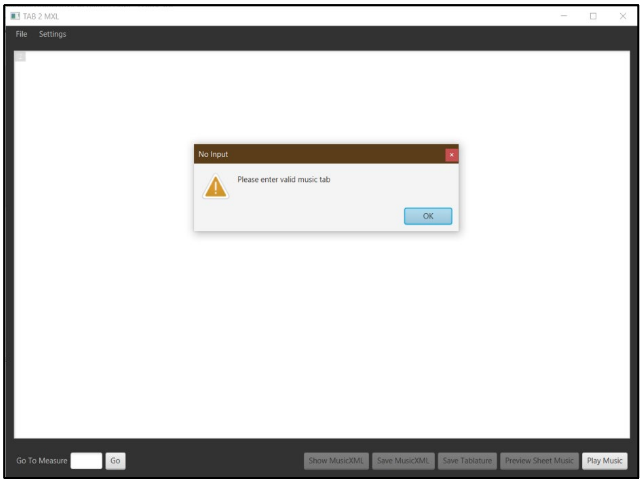
Further test: We wanted to see what happened had we entered multiple empty spaces. This resulted in an output as shown below in the “Before” image. To fix this we added a strip function to fix this

Before:



```
dle Project] run in C:\Users\Shaharyar\git\TAB2XML (Mar. 5, 2022, 11:33:39 p.m.)
a.base/java.lang.reflect.Method.invoke(Method.java:568)
afx.base/com.sun.javafx.reflect.MethodUtil.invoke(MethodUtil.java:275)
afx.fxml/com.sun.javafx.fxml.MethodHelper.invoke(MethodHelper.java:84)
afx.fxml/javafx.fxml.FXMLLoader$MethodHandler.invoke(FXMLLoader.java:1854)
 more
a.lang.NullPointerException: Cannot invoke "utility.MusicXMLCreator.generateMusicXML()" becaus
verter.Converter.getMusicXML(Converter.java:28)
```

After:



Please enter valid music tab

2) **Test Case 2:** Test to see if the “Play” button, on the music player, works when an input of Guitar text-tablature is used.

How was this derived: It was a function that needed to be implemented as stated by the “customer” and the way we implemented it was to use a play button in which, when pressed, would play music audio.

How was this tested: When pressing the play button we check to see if the audio of the music is played.

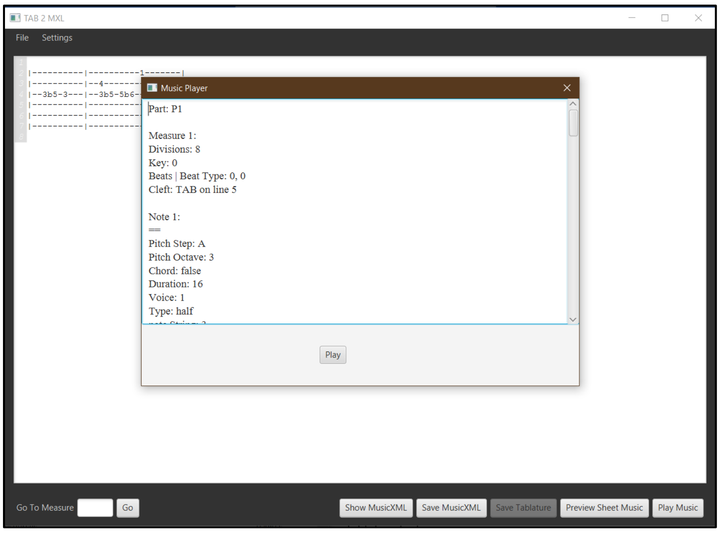
3) **Test Case 3:** Test to see if the music-player GUI output accurately matches the expected output

How was this derived: The text area is meant to display information about the users inputted tablature music. The information includes measure number, number of divisions, key, pitch step, duration, part number, and more. It is crucial for the correct information to be displayed as the user would like to learn more about the specific notes and measures.

How was this tested: We checked to see if the expected output is equal to the actual output. So when running the test we compared the two strings of information. We repeated this multiple times with different inputs of music tablature text.

This is sufficient as it handles all possible inputs and handles the cases when an unexpected output is encountered.

Expected Output:



```
|----------|----------1-------|
|----------|--4-------
|--3b5-3---|--3b5-5b6-
|----------|----------
|----------|----------
|----------|----------
```

```
Part: P1

Measure 1:
Divisions: 8
Key: 0
Beats | Beat Type: 0, 0
Cleft: TAB on line 5

Note 1:
==
Pitch Step: A
Pitch Octave: 3
Chord: false
Duration: 16
Voice: 1
Type: half
```

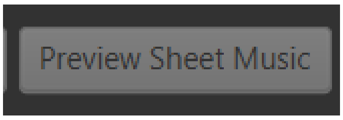
Sheet Music Creator Test Cases

1) **Test Case 1**: Test to see if the "Preview Sheet Music" button becomes enabled when a non-empty output in entered

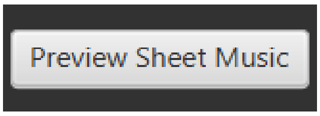
How was this derived: Initially, the "Preview Sheet Music" button was disabled if the user entered input. When the user enters a ASCII music tab. It enables the "Preview Sheet Music" button along with other buttons.

How this was tested: We ran the code with an empty input and found the button to be disabled. When a correct tablature was inserted from the test cases, the button became enabled, and we were able to click it.

Disabled Button:



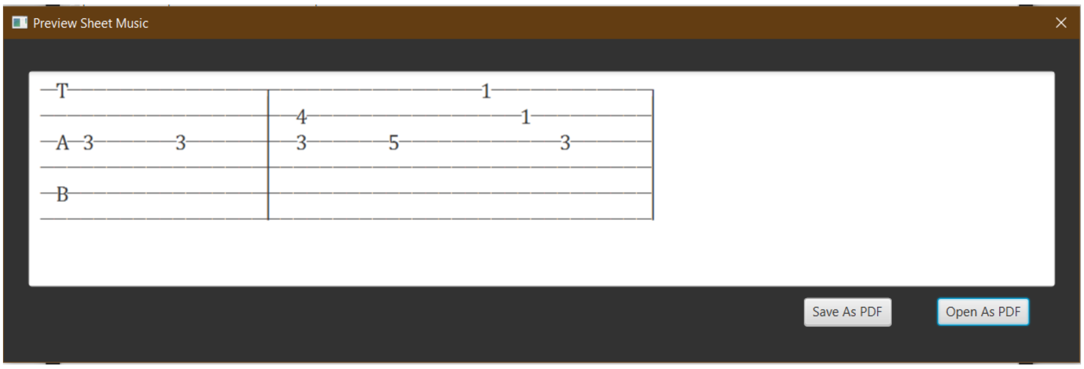
Enabled Button:



2) **Test Case 2:** Test to see if the "Preview Sheet Music" button, on the main screen, works when an input of Guitar text-tablature is used.

How was this derived: It was a function that needed to be implemented as stated by the "customer" and the way we implemented it was when it was clicked another window would open and display the music tablature.

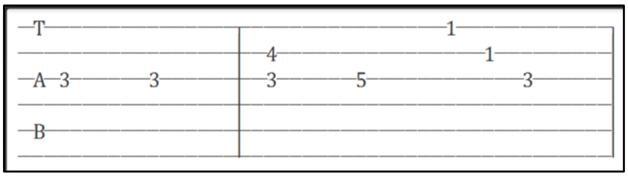
Expected Output:



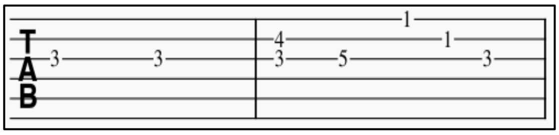
3) **Test Case 3:** Test to see if the Music Tablature Produced by the software is accurate.

How was this derived: We compared the music tablature created by our software to a music tablature created by MuseScore3.0 and SoundSlice web application. Our results were 90% similar to these applications.

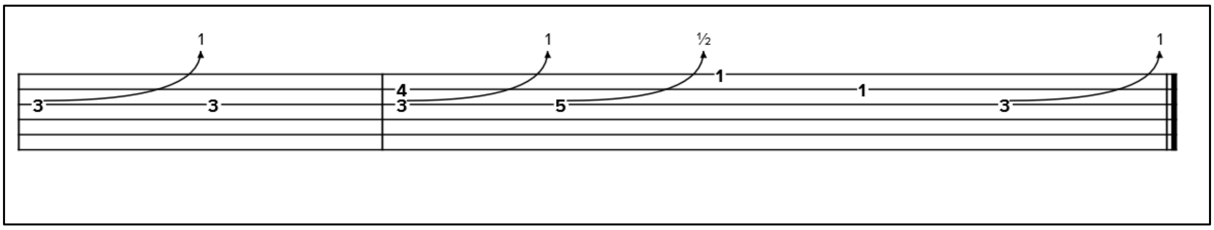
Our Software:



MuseScore3 Results:

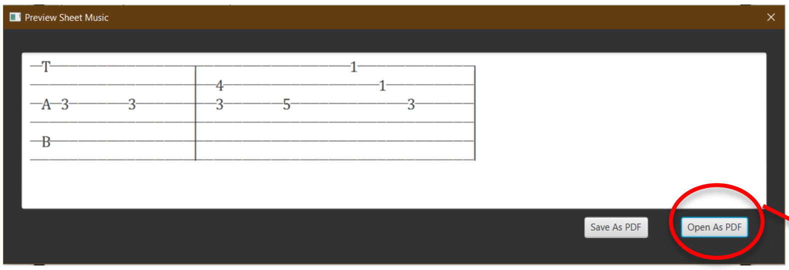


SoundSlice Results:



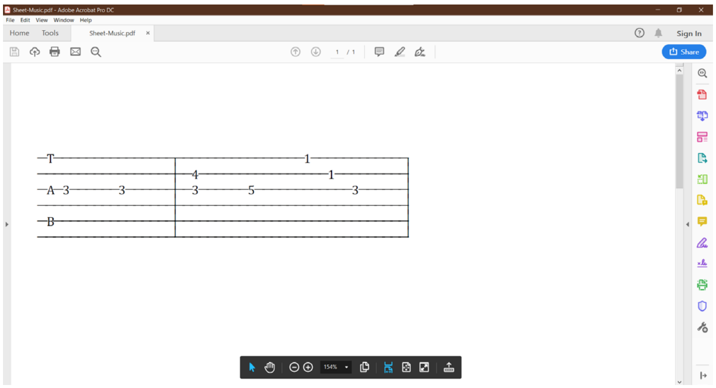
4) **Test Case 4:** Test to see if our program can view the file as a PDF document.

How was this derived: We tested the feature in our software to export the music tablature as a PDF or open it as a PDF file.



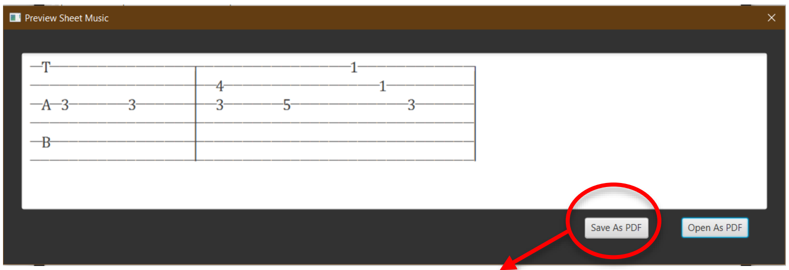
Open Tablature As PDF Document

Expected Output: The software was able to generate a temporary PDF file and open it using the default PDF Viewer.


Sheet-Music.pdf - Adobe Acrobat Pro DC
File Edit View Window Help
Home Tools Sheet-Music.pdf
Sign In
Share
1 / 1
154%

5) **Test Case 5:** Test to see if our program can export the file as a PDF document.

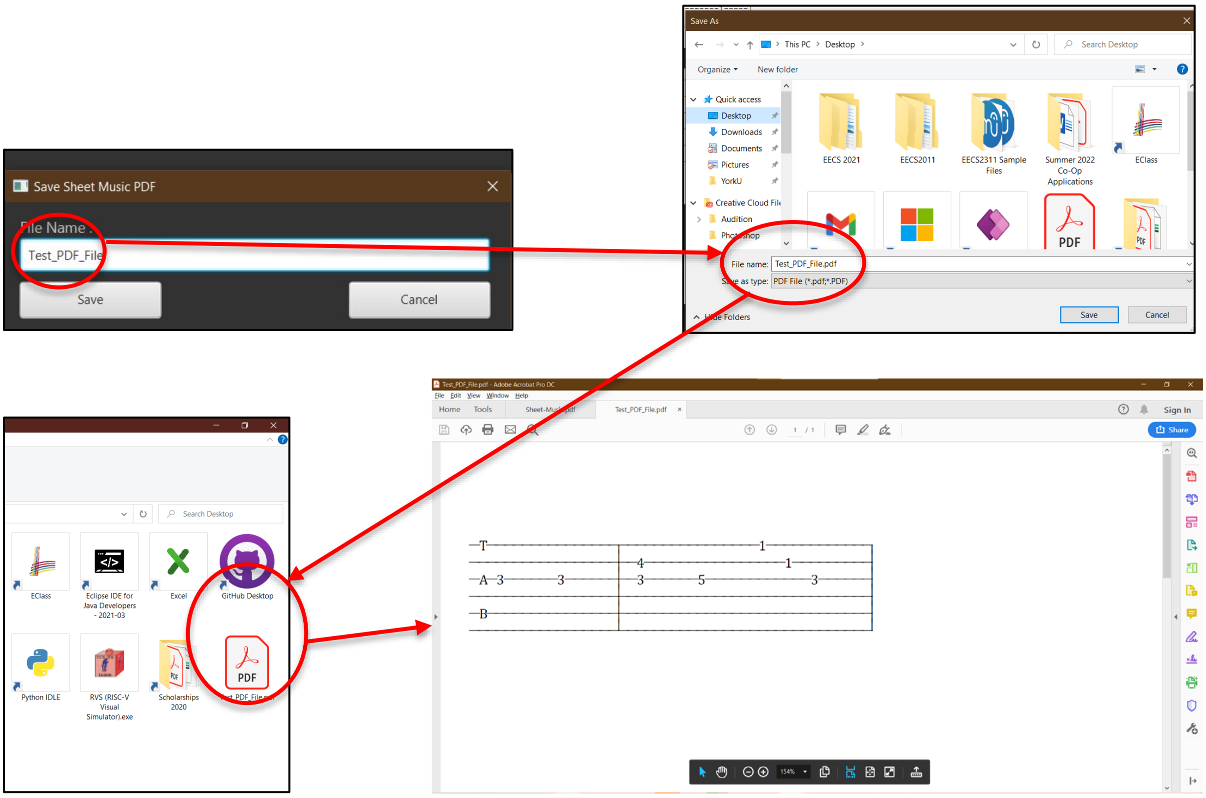
How was this derived: We tested the feature in our software to export the music tablature as a PDF.



Save Tablature As PDF Document

The software was able to open a window to enter the file name and then would open up a file directory to allow the user to select a destination and save the file.

Output:



File Name: Test_PDF_File

File name: Test_PDF_File.pdf

Save as type: PDF File (*.pdf;*.PDF)