Design Document

By: Zaeem Israr, Kevin Banh, Amir Khademi

Table of Contents

The following short paragraphs describe the classes and their basic functionality.

Node.java

This class allows creation of a Node object. A Node object contains information such as text to be spoken, tags, more information regarding the tags (such as pause duration) and how nodes are linked together (parent nodes and child nodes). A combination of nodes based on their tags, text and how they are connected are a representation of a scenario, but not in a text file format.

FileGeneration.java

This class allows the creation of a FileGeneration object. After adding all the nodes representing a scenario to the FileGeneration object, there is a method that allows the generation of a text file representing the scenario. The method works by traversing the connected nodes, storing the text and appropriate tags based on each node's content and writing to a file at the end.

FileGernationSimplified.java

This is a static class. Instead of using a FileGeneration object, creating all the nodes, adding the nodes to the FileGeneration object and et cetera to generate the text file, this class simplifies the process. The class, given the correct information, will create the nodes using Node constructor and automatically create a FileGeneration object, add the nodes and create the text file.

Client.java

The client class contains the main method and simply creates the main window of the GUI. It creates an object of the main window, sets its layout to grid layout and sets the frame to visible. The client class has no other function.

MainFrame.java

The file frame class simply describes a jframe with three buttons labeled “New File”, “Record Audio” and “Load File”. The New File button creates an instance of the File Frame class, the Record Audio button creates an instance of the Audio Frame class while the Load File button creates an instance of the Load Form class.

FileFrame.java

The file frame class creates a frame with two text fields to enter the number of cells and buttons, a “Create Node” button and a “Save” button. This frame also creates a scrollable panel on which nodes created by the node panel class are created. The nodes of the node panel class are created as an array of panels which are placed on an invisible panel which allows the scrollable panel to function properly. Pressing the “Add Node” button creates an instance of the Node Panel class. Pressing the “Save” button uses the methods in FileGeneration and FileGenerationSimplified to create the file.

NodePanel.java

The node panel class describes the representation of a node in the GUI. This class exists as a panel and cannot be initialized on its own without a jFrame class. The class describes vaious text fields to enter text, select parent and merge nodes and select tags using a combo box. The class also provides getter methods that provide references to the text fields and combo boxes for easy interaction between different frame classes.

AudioFrame.java

The audio frame class simply creates a GUI with a text field where the name of the audio file to be created is typed. It also creates a button labeled “Start” and “Stop” which are used to start and stop recording. This class uses methods provided in the Sound Recorder class to create the audio file.

LoadForm.java

The load form creates a jFrame with an editable text area and a “Save” button. The text area reads the text from the output file and displays it. Users can edit the text as desired and press the save button. Pressing the save button would overwrite the existing save file with the edited version. The class makes use of file readers, scanners and writers to allow reading, editing and writing the file.

ErrorForm.java

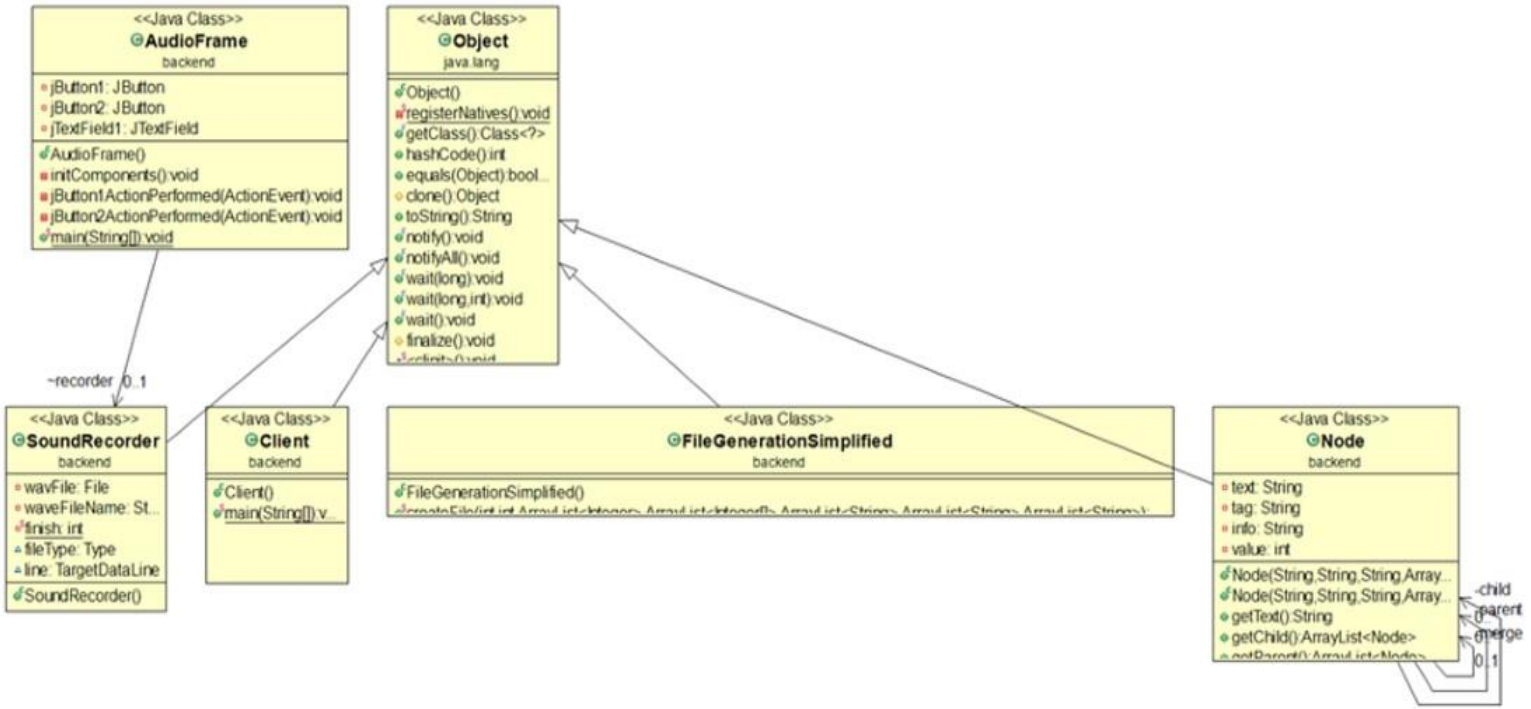
The error form class creates a small window with a text area. The text area displays the error messages generated when creating the text file. This is handled by a getter method that provides reference to the text area. The error messages are generated in the File Frame class.

SounRecorder.java

The sound recorder class handles the audio files. It provides three methods to start recording, stop recording and naming the generated file. The file frame class has no other purpose.

UML Diagram:

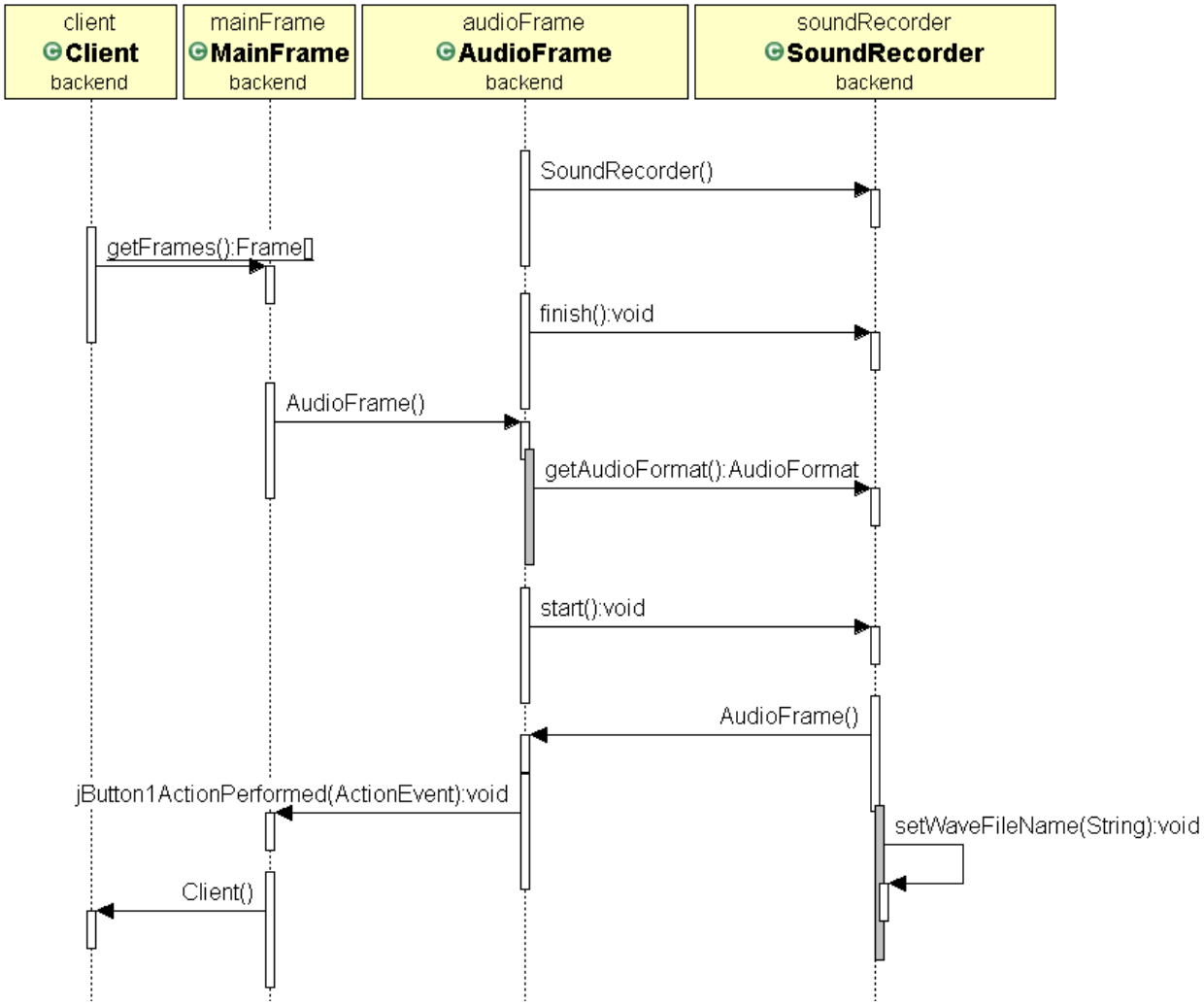
The UML diagram of the classes described above is as follows:


<<Java Class>>
AudioFrame
backend
jButton1: JButton
jButton2: JButton
jTextField1: JTextField
AudioFrame()
initComponents():void
jButton1ActionPerformed(ActionEvent):void
jButton2ActionPerformed(ActionEvent):void
main(String[]):void
<<Java Class>>
Object
java.lang
Object()
registerNatives():void
getClass():Class<?>
hashCode():int
equals(Object):bool...
clone():Object
toString():String
notify():void
notifyAll():void
wait(long):void
wait(long,int):void
wait():void
finalize():void
~recorder 0..1
<<Java Class>>
SoundRecorder
backend
wavFile: File
waveFileName: St...
finish: int
fileType: Type
line: TargetDataLine
SoundRecorder()
<<Java Class>>
Client
backend
Client()
main(String[]):v...
<<Java Class>>
FileGenerationSimplified
backend
FileGenerationSimplified()
<<Java Class>>
Node
backend
text: String
tag: String
info: String
value: int
Node(String,String,String,Array...
Node(String,String,String,Array...
getText():String
getChild():ArrayList<Node>
-child
-parent
0..1

Sequence Diagrams:

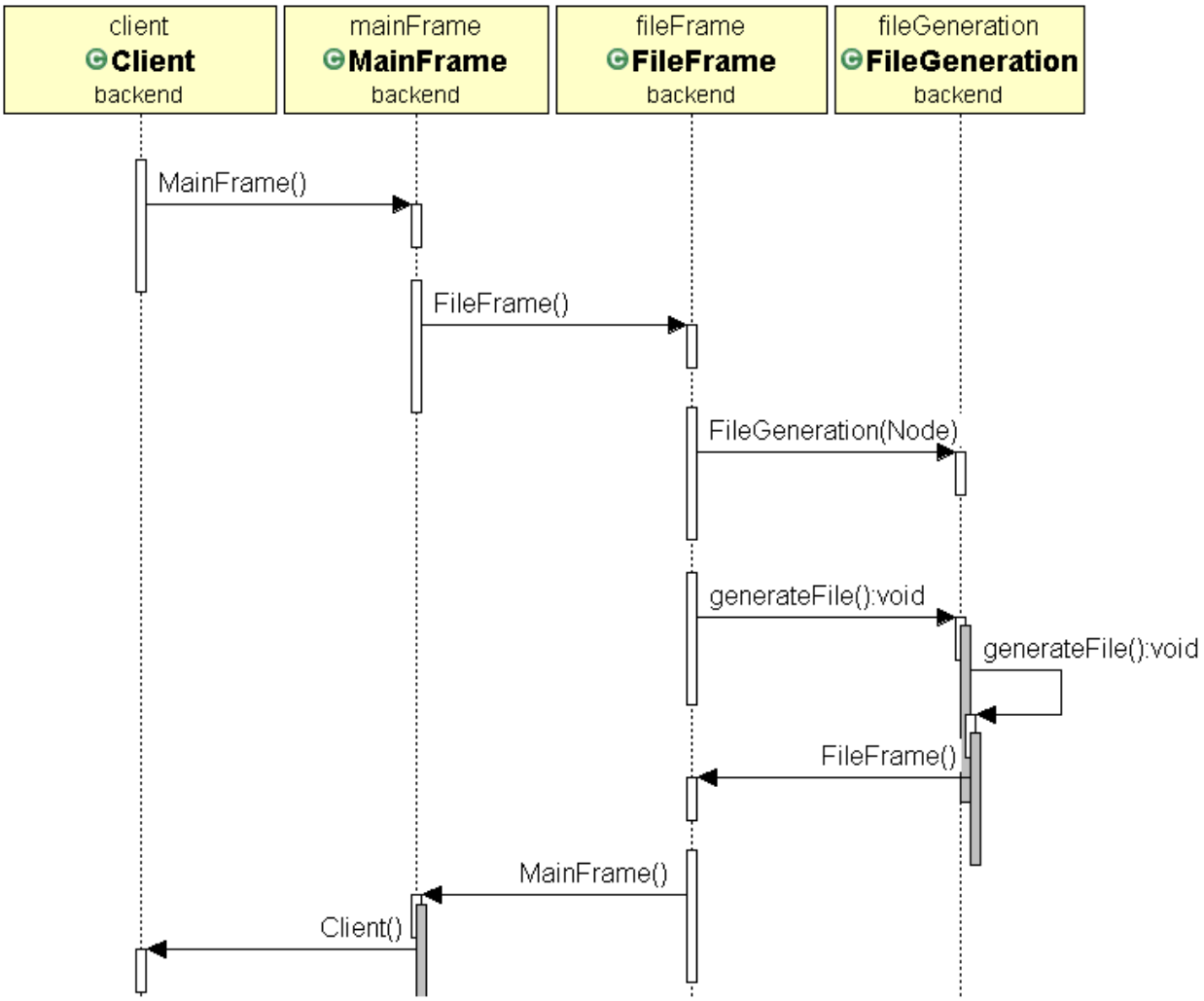
The following sequence diagram describes how the objects interact to create the audio file.

The client class crates the MainFrame object, which crates the AudioFrame abject upon clicking the button. The methods in SoundRecorder class are then called by the AudioFrame to create the .wav file.

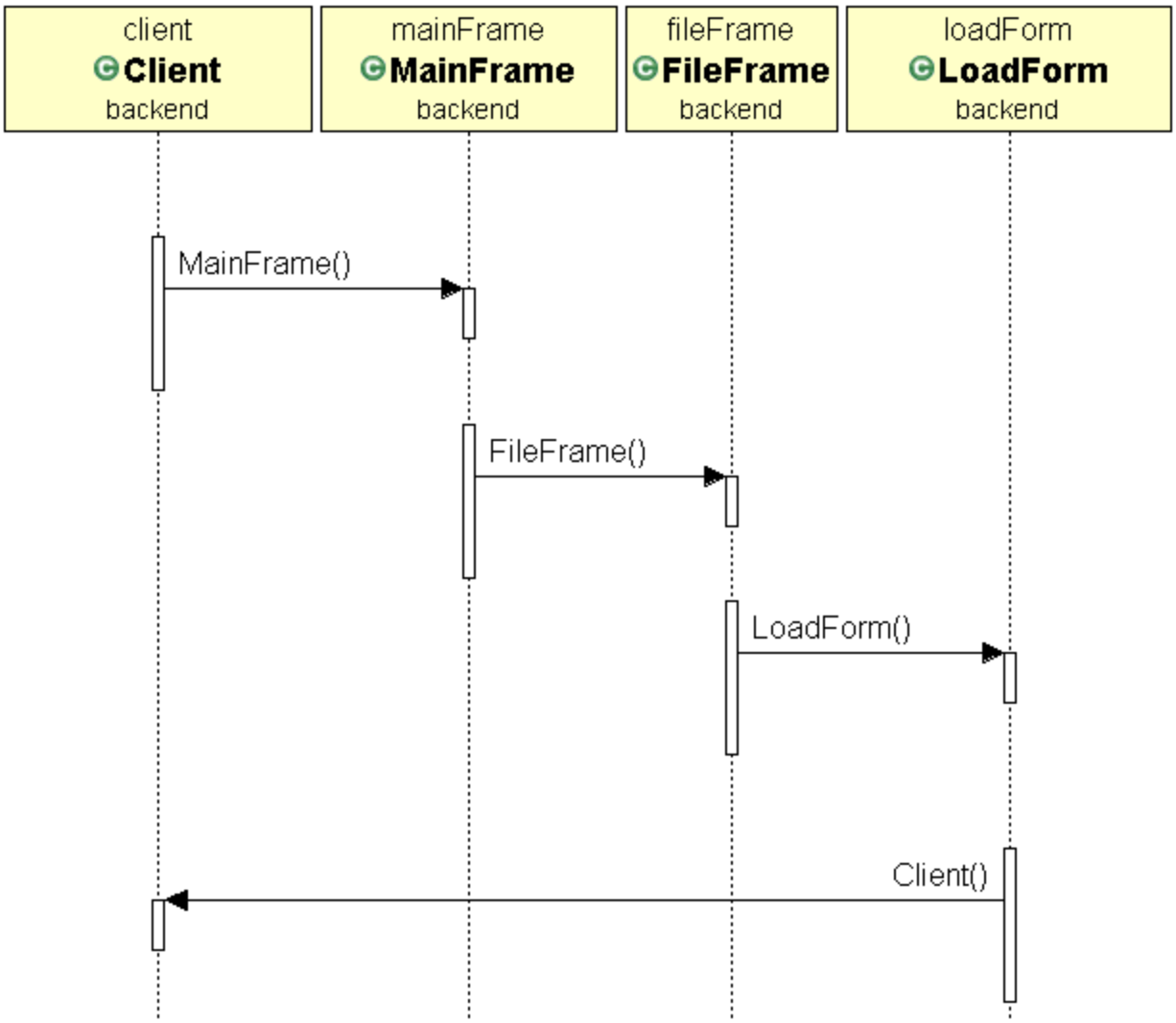


The following sequence diagram describes how the objects interact to create the text file.

The client class creates an object of MainFrame which upon pressing the save button in the FileFrame calls the methods on FileGeneration class to create the text file.



The following sequence diagram describes how objects interact to load and edit the file. The client class creates an object of The MainFrame which upon pressing the load button instantiates an object of LoadForm class. The scanner and writer in LoadForm class are then used to read and edit the text file.


client
Client
backend
mainFrame
MainFrame
backend
fileFrame
FileFrame
backend
loadForm
LoadForm
backend
MainFrame()
FileFrame()
LoadForm()
Client()