Text2SpeechEditor

Sprint No 1 Report

Mpoz Ntoyran 2310

VERSIONS HISTORY

Date	Version	Description	Author
10/05/2020	Final		Mpoz Ntoyran

1 Introduction

This document provides information concerning the **1** sprint of the project.

1.1 Purpose

The following document is about the developing of the TextToSpeechEditor.This project is for transforming plain or encoded text to speech by using the FreeTts 1.2 library.The user can load or create a document witch also can be edited. Next he can transform the document to speech or can choose a line to transform to speech.Encoding the document is also available as a feature and transforming it after that to speech whole or by choosing a line.

1.2 Document Structure

The rest of this document is structured as follows. Section 2 describes out Scrum team and specifies the this Sprint's backlog. Section 3 specifies the main design concepts for this release of the project.

2 Scrum team and Sprint Backlog

Test's user story	1
Test file name	NewDocumentTest
Test description	In this test i test the US 1. By creating a author ,title which are of type String and a date. After the creation of the document we test if the titles authors and dates are the same.

Test's user story	2
Test file name	EditDocumentTest
Test description	In this test i test the US 2.After creating a document with the OpenDocument command we open it.By adding a new line to the content and executing the EditDocument command.With the OpenDocument command I reopen the document and test if the added line is in the contents of the file.

Test's user story	3
Test file name	SaveDocumentTest
Test description	In this test i test the US 3.By creating a new document and then executing the SaveDocument command ,after that we open the saved document with the OpenDocument command if the contetnts of the 2 document are the same then we have a successful test.

Test's user story	4
Test file name	OpenDocumentTest
Test description	In this test i test the US 4.First I create a file with all the field I need for a document and I write them.With only providing the the path of the created file i create an OpenDocument command and execute it.After i test if the the fields that I have are the same with the fields of the newly opened document.

Test's user story	5
Test file name	DocumentToSpeechTest
Test description	In this test i test the US 5.First I create a new document with some content. Secondly I change the audioMnager of the document with the FakeTextToSpeechApi which instred of transforming to speech it stores everything.After creating and executing a DocumentToSpeech command I check if the contents of the documents are the same as the contents of the FakeTextToSpeechApi.(FakeTextToSpeechApi is a TextToSpeechApi just to be more clear I refer to it as FakeTextToSpeechApi).

Test's user story	6
Test file name	LineToSpeechTest
Test description	In this test i test the US 6.By using a create document I change the audioManager.After creating and executing a LineToSpeech command I get the contents of the audioManager.Comparing the line from the contents of the document I got with the contents of the audioManager we get.

Test's user story	7
Test file name	ReverseDocumentToSpeechTest
Test description	In this test i test the US 7.After creating a document and changing the audioManager.I create and execute an ReverseDocumentToSpeech command.I get the contents of the audioManager if the contents are in the order described in the user story 7 the test is successful.

Test's user story	8
Test file name	ReverseLineToSpeechTest
Test description	In this test i test the US 8.As usually after creating a document and change its audioManager.I create a ReverseLineToSpeech command and execute it.I get the contents of the audioManager and if the contents of the audio manager are in the order described by the user story 8 the test is successful.

Test's user story	9
Test file name	EncodedDocumentToSpeech
Test description	In this test i test the US 9.In a document i created, i tune the encoding to Rot13 at the first test and AtBash for the second test.Also I change the audioManager.First I encode the document with the TuneEncodings command and after I play the document with EncodedDocumentToSpeech command.By encoding the content in an online encoder I check if the contents at the audioManager are the same with the encoded of the online encoder if they are the test is successful.

Test's user story	10
Test file name	PlayEncodedLineToSpeechTest
Test description	In this test i test the US 10.In a document create I create for the test I tune the encoding and change the audioManager.After creating and executing the PlayEncodedLineToSpeech.I check the contents of the audioManager with the encoded version of the selected line which I encoded online. If they are the same the test is successful.

Test's user story	11
Test file name	TuneEncodingsTest
Test description	In this test i test the US 11.By creating a document and a encodingStrategy.After creating and excuting the TuneEncodings command get the encoding of the document and check if the encodingStrategy I created is the same with the encodingStrategy I get from the document.

Test's user story	12
Test file name	TuneAudioTest
Test description	In the test i test the US 12.With a FakeTextToSpeechApi I create and int with a value after creating a TuneAudio command I execute it I get the value I change from the audioManager if the value from the audioManager is the same with the value I created then the test is successful.

2.1 Scrum team

Product Owner	Apostolos Zarras
Scrum Master	…
Development Team	Mpoz Ntoyran

2.2 Sprint Backlog

- [US-1] As a user, I want to create a new empty document, by giving its title and author. The application should automatically record the creation date.

- [US-2] As a user, I want to edit the contents of the document, via the application's user interface.

- [US-3] As a user, I want to save the contents of the document to disk by providing a particular filename. The application should automatically record the save date.

- [US-4] As a user, I want to open the contents of an existing document from disk by providing a particular file path, or by browsing the file system folders.

- [US-5] As a user, I want to transform the contents of the document to speech.

- [US-6] As a user, I want to select a line and transform it to speech.

- [US-7] As a user I want to transform the contents of the document to speech in reverse, i.e. play the last word of the last line first and so on.

- [US-8] As a user I want to select a line and transform it to speech in reverse, i.e. and play the last word first and so on.

- [US-9] As a user I want to encode the contents of the document and then transform them to speech.

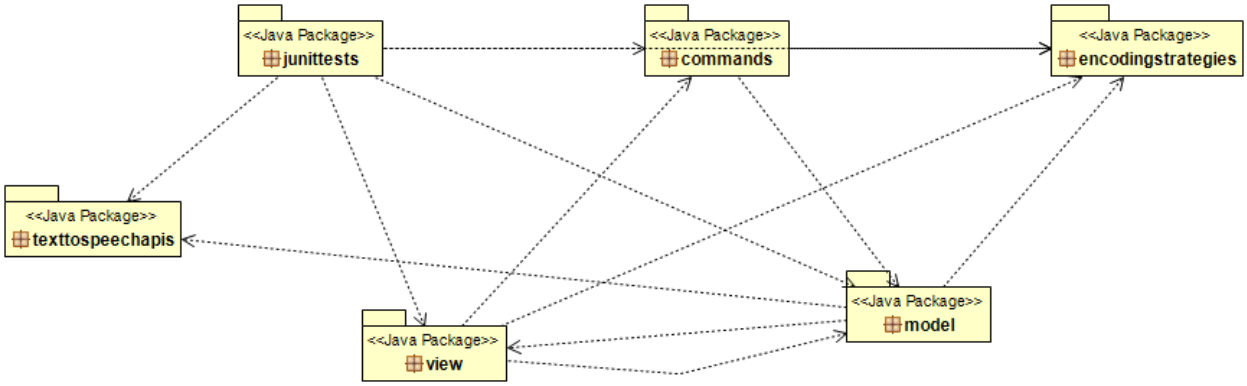
- [US-10] As a user I want to select a line, encode it and transform it to speech.

- [US-11] As a user I want to tune the text encoding technique. In particular the application should support at least the following encoding strategies:
    - Atbash: The Atbash cipher is formed by taking the alphabet and mapping it to its reverse, so that the first letter becomes the last letter, the second letter becomes the second to last letter, and so on.
    - Rot-13: Rot-13 is a letter substitution cipher that replaces a letter with the 13th letter after it, in the alphabet. Rot-13 is a special case of the Caesar cipher, which was developed in ancient Rome.

- [US-12] As a user I want to be able to tune the audio parameters, i.e., the volume, the speech rate and the pitch.

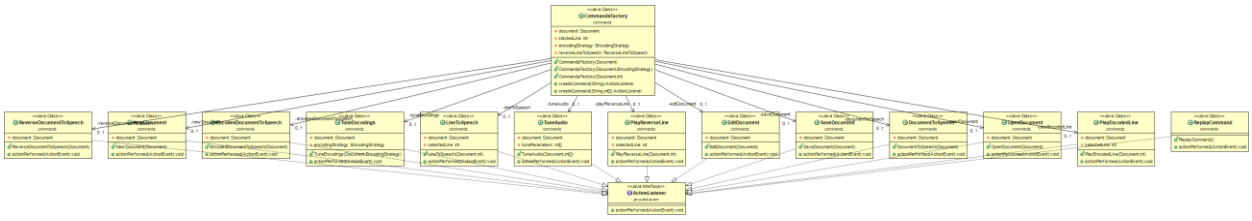
3 Design

3.1 Architecture

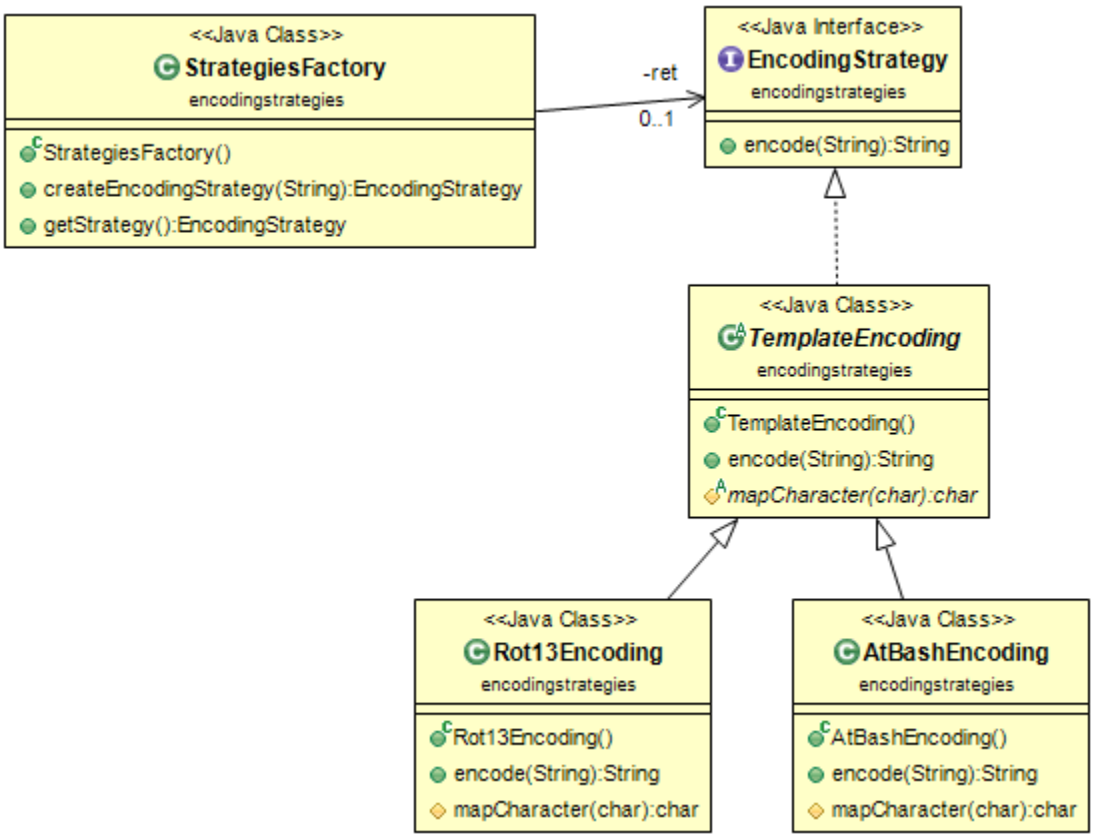


3.2 Design

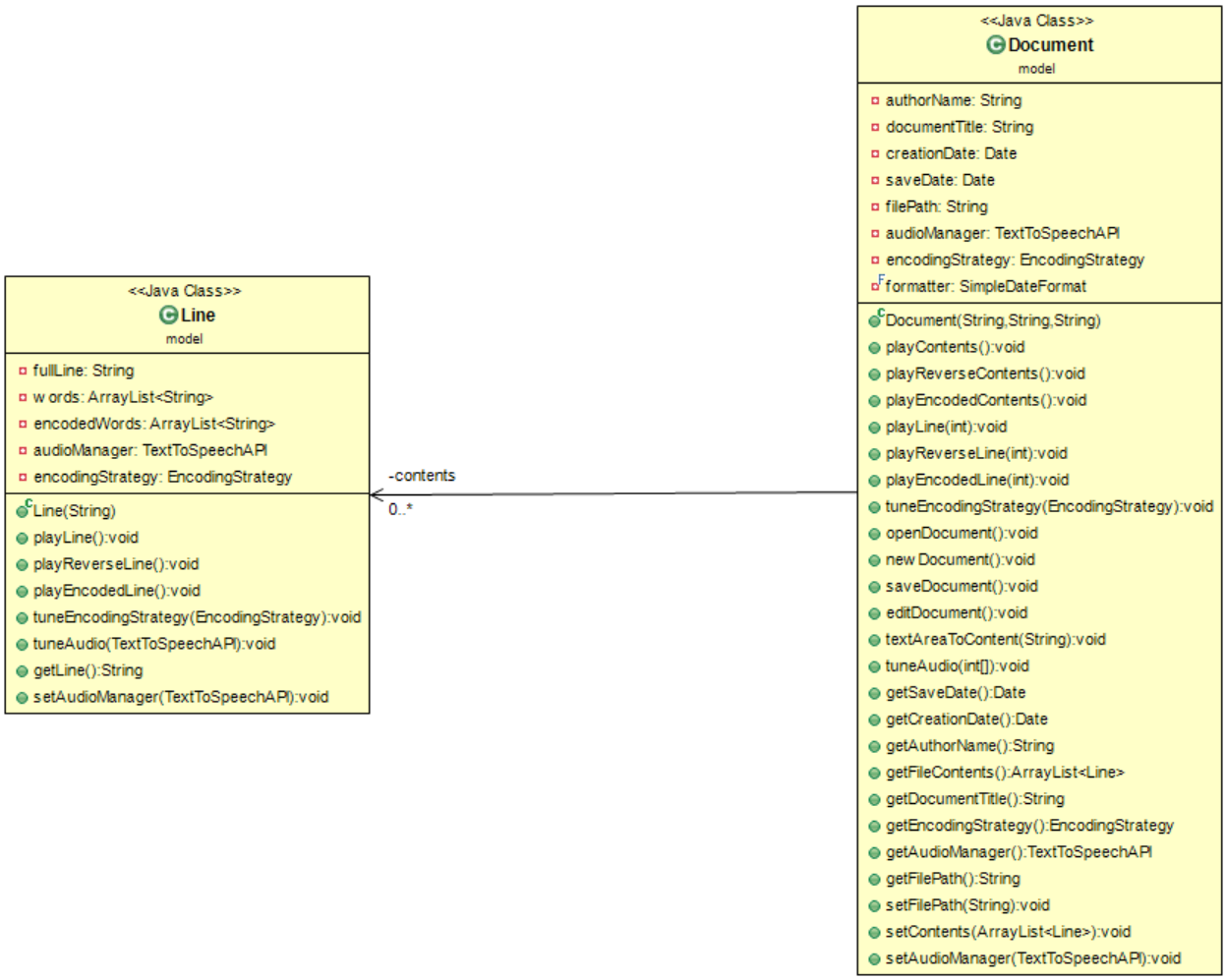
Commands package



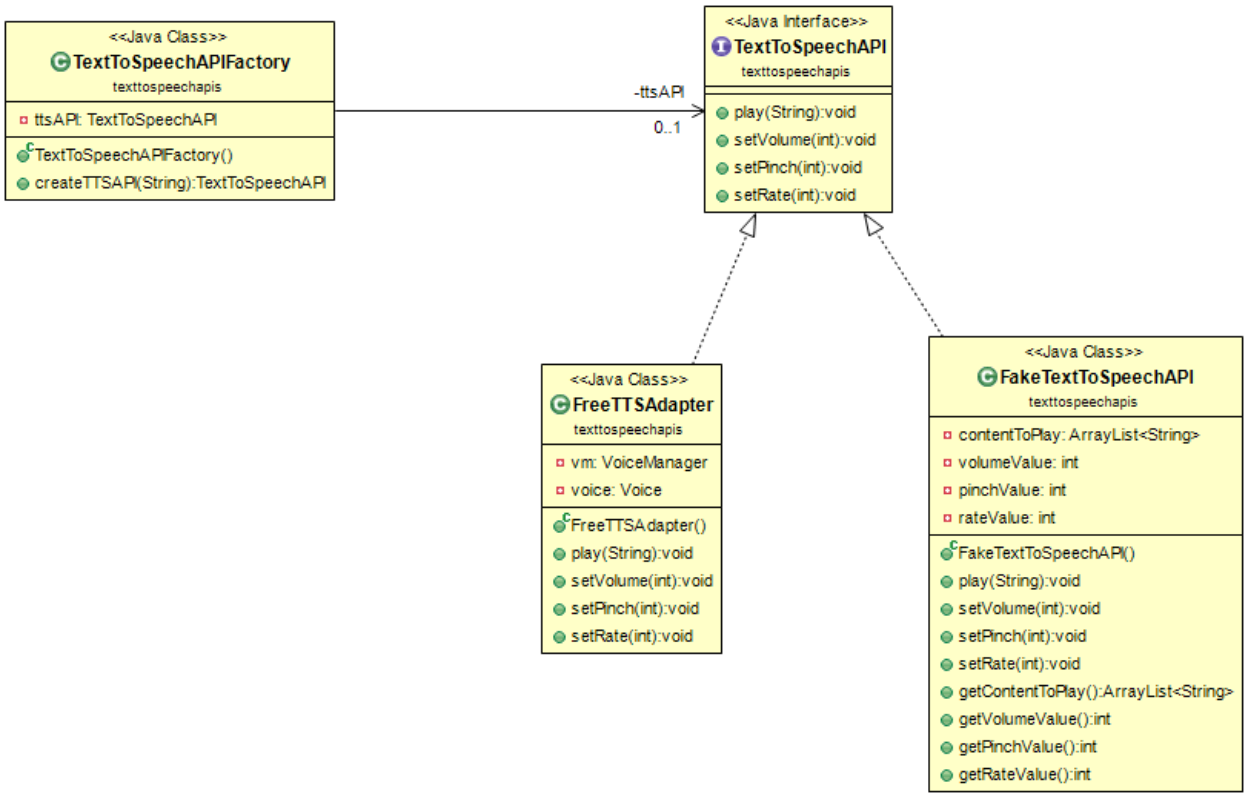
Encodingstrategies package



Model package


<<Java Class>>
Line
model
fullLine: String
words: ArrayList<String>
encodedWords: ArrayList<String>
audioManager: TextToSpeechAPI
encodingStrategy: EncodingStrategy
Line(String)
playLine():void
playReverseLine():void
playEncodedLine():void
tuneEncodingStrategy(EncodingStrategy):void
tuneAudio(TextToSpeechAPI):void
getLine():String
setAudioManager(TextToSpeechAPI):void
-contents
0..*
<<Java Class>>
Document
model
authorName: String
documentTitle: String
creationDate: Date
saveDate: Date
filePath: String
audioManager: TextToSpeechAPI
encodingStrategy: EncodingStrategy
formatter: SimpleDateFormat
Document(String,String,String)
playContents():void
playReverseContents():void
playEncodedContents():void
playLine(int):void
playReverseLine(int):void
playEncodedLine(int):void
tuneEncodingStrategy(EncodingStrategy):void
openDocument():void
newDocument():void
saveDocument():void
editDocument():void
textAreaToContent(String):void
tuneAudio(int[]):void
getSaveDate():Date
getCreationDate():Date
getAuthorName():String
getFileContents():ArrayList<Line>
getDocumentTitle():String
getEncodingStrategy():EncodingStrategy
getAudioManager():TextToSpeechAPI
getFilePath():String
setFilePath(String):void
setContents(ArrayList<Line>):void
setAudioManager(TextToSpeechAPI):void

TextToSpeechAPI_package



View package

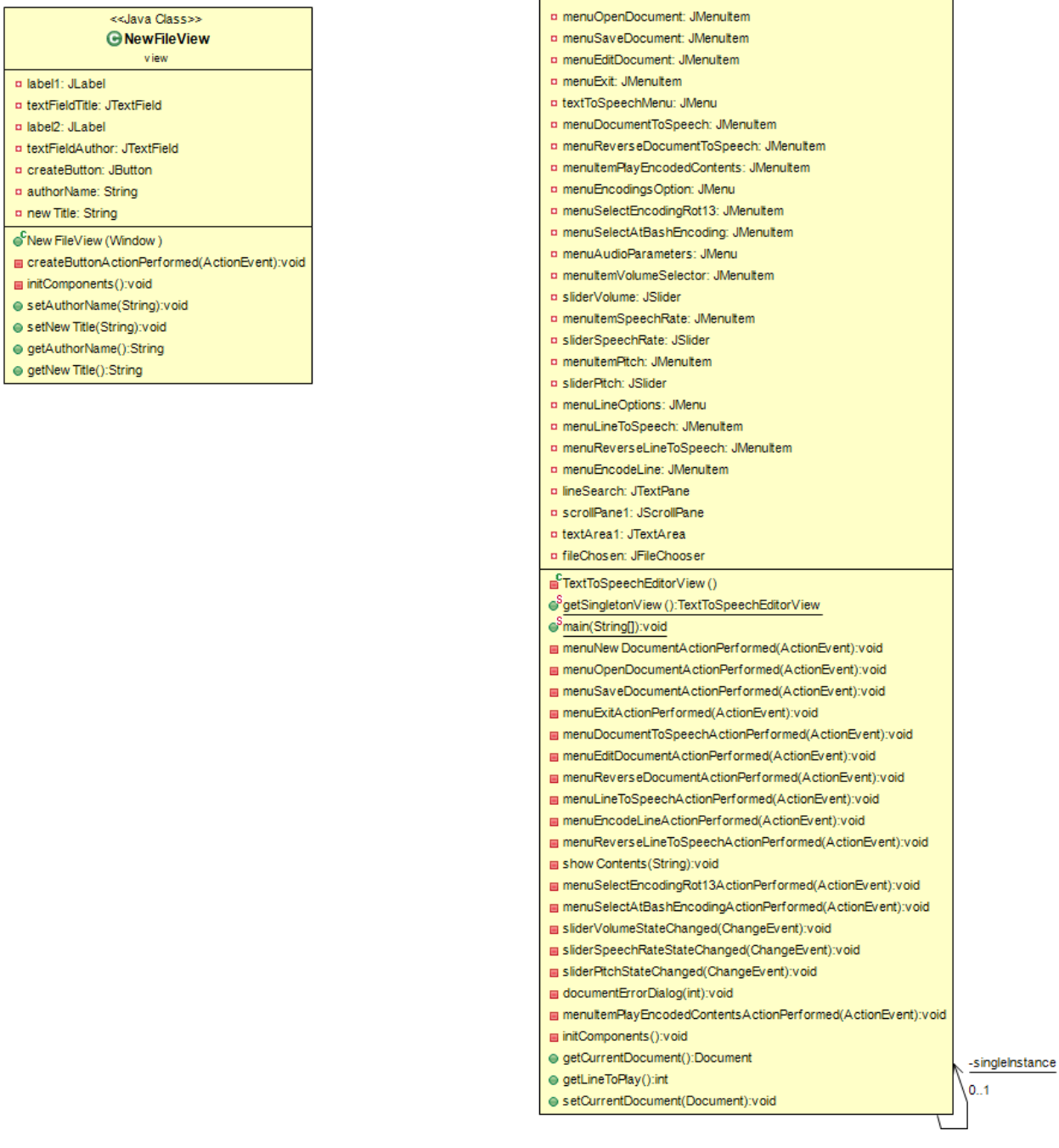

<<Java Class>>
NewFileView
view
label1: JLabel
textFieldTitle: JTextField
label2: JLabel
textFieldAuthor: JTextField
createButton: JButton
authorName: String
newTitle: String
New FileView (Window)
createButtonActionPerformed(ActionEvent):void
initComponents():void
setAuthorName(String):void
setNewTitle(String):void
getAuthorName():String
getNewTitle():String
<<Java Class>>
TextToSpeechEditorView
view
currentDocument: Document
commandsFactory: CommandsFactory
strategiesFactory: StrategiesFactory
command: ActionListener
menuBar1: JMenuBar
fileMenu: JMenu
menuNew Document: JMenuItem
menuOpenDocument: JMenuItem
menuSaveDocument: JMenuItem
menuEditDocument: JMenuItem
menuExit: JMenuItem
textToSpeechMenu: JMenu
menuDocumentToSpeech: JMenuItem
menuReverseDocumentToSpeech: JMenuItem
menuItemPlayEncodedContents: JMenuItem
menuEncodingsOption: JMenu
menuSelectEncodingRot13: JMenuItem
menuSelectAtBashEncoding: JMenuItem
menuAudioParameters: JMenu
menuItemVolumeSelector: JMenuItem
sliderVolume: JSlider
menuItemSpeechRate: JMenuItem
sliderSpeechRate: JSlider
menuItemPitch: JMenuItem
sliderPitch: JSlider
menuLineOptions: JMenu
menuLineToSpeech: JMenuItem
menuReverseLineToSpeech: JMenuItem
menuEncodeLine: JMenuItem
lineSearch: JTextPane
scrollPane1: JScrollPane
textArea1: JTextArea
fileChosen: JFileChooser
TextToSpeechEditorView ()
getSingletonView ():TextToSpeechEditorView
main(String[]):void
menuNew DocumentActionPerformed(ActionEvent):void
menuOpenDocumentActionPerformed(ActionEvent):void
menuSaveDocumentActionPerformed(ActionEvent):void
menuExitActionPerformed(ActionEvent):void
menuDocumentToSpeechActionPerformed(ActionEvent):void
menuEditDocumentActionPerformed(ActionEvent):void
menuReverseDocumentActionPerformed(ActionEvent):void
menuLineToSpeechActionPerformed(ActionEvent):void
menuEncodeLineActionPerformed(ActionEvent):void
menuReverseLineToSpeechActionPerformed(ActionEvent):void
show Contents(String):void
menuSelectEncodingRot13ActionPerformed(ActionEvent):void
menuSelectAtBashEncodingActionPerformed(ActionEvent):void
sliderVolumeStateChanged(ChangeEvent):void
sliderSpeechRateStateChanged(ChangeEvent):void
sliderPitchStateChanged(ChangeEvent):void
documentErrorDialog(int):void
menuItemPlayEncodedContentsActionPerformed(ActionEvent):void
initComponents():void
getCurrentDocument():Document
getLineToPlay():int
setCurrentDocument(Document):void
-singleInstance
0..1

Figure 1

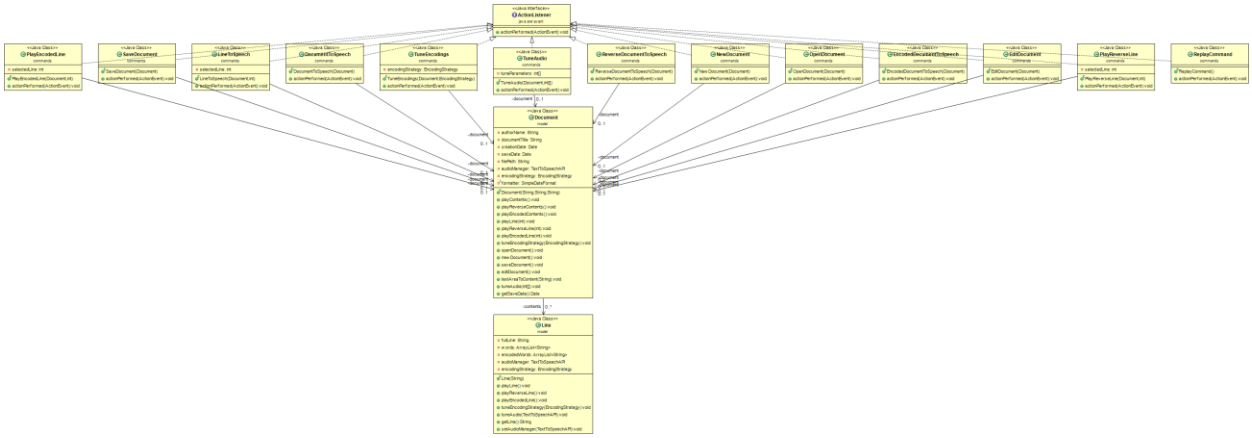


Figure_2

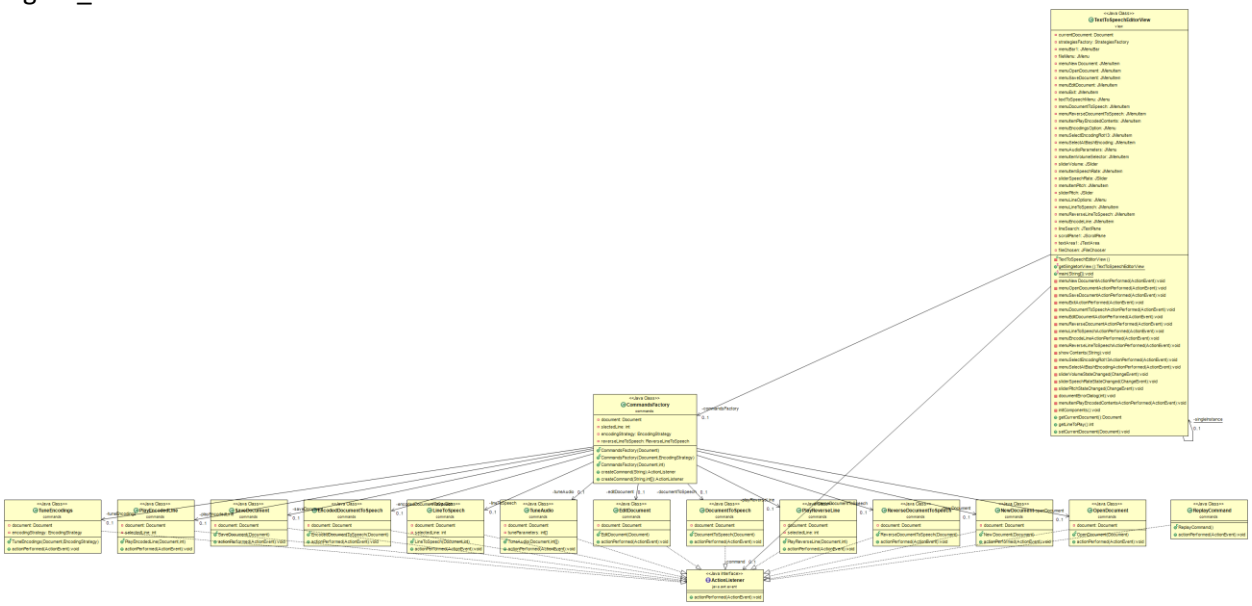
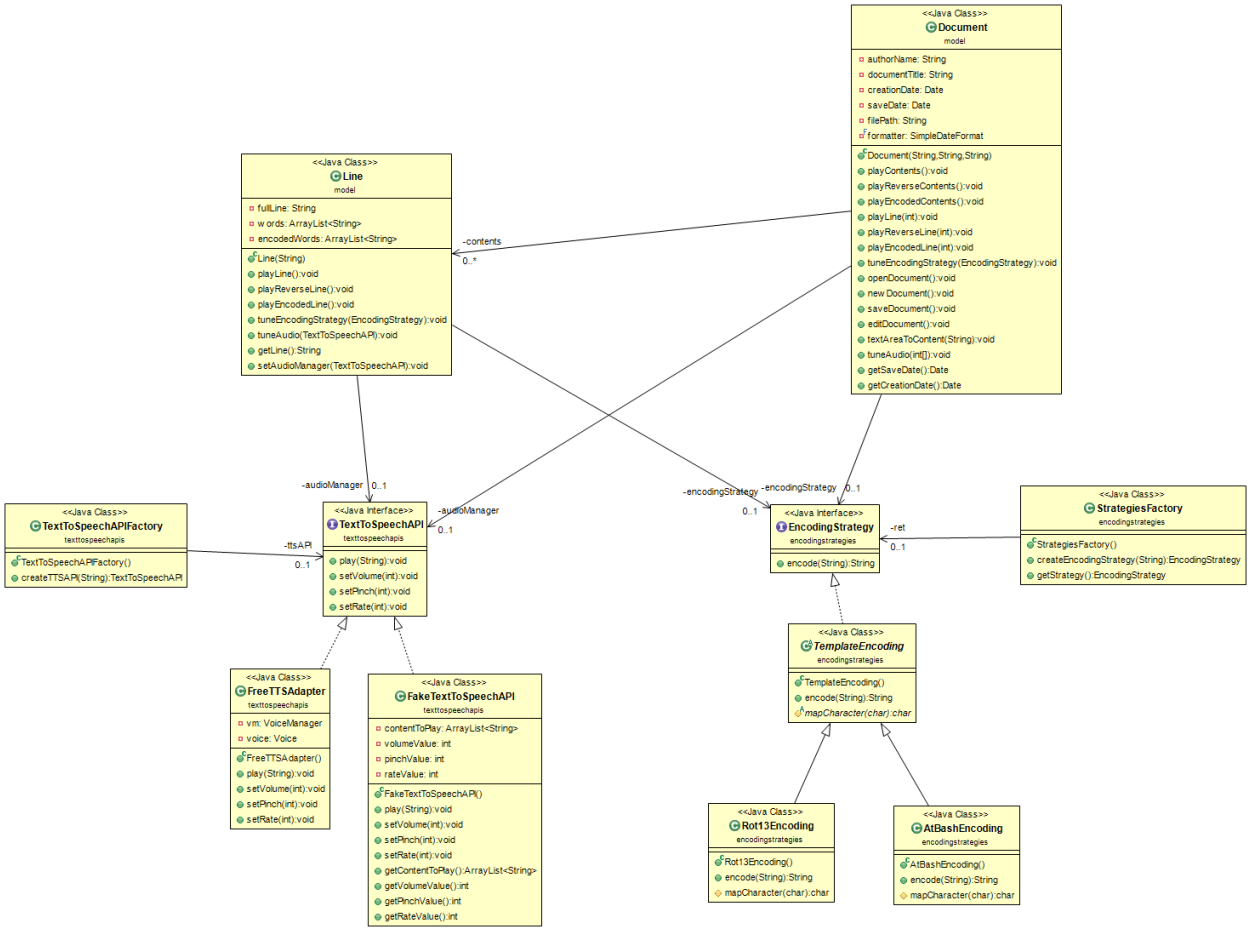


Figure 3


<<Java Class>>
Document
model
authorName: String
documentTitle: String
creationDate: Date
saveDate: Date
filePath: String
formatter: SimpleDateFormat
Document(String,String,String)
playContents():void
playReverseContents():void
playEncodedContents():void
playLine(int):void
playReverseLine(int):void
playEncodedLine(int):void
tuneEncodingStrategy(EncodingStrategy):void
openDocument():void
new Document():void
saveDocument():void
editDocument():void
textAreaToContent(String):void
tuneAudio(int[]):void
getSaveDate():Date
getCreationDate():Date
<<Java Class>>
Line
model
fullLine: String
words: ArrayList<String>
encodedWords: ArrayList<String>
Line(String)
playLine():void
playReverseLine():void
playEncodedLine():void
tuneEncodingStrategy(EncodingStrategy):void
tuneAudio(TextToSpeechAPI):void
getLine():String
setAudioManager(TextToSpeechAPI):void
-contents
0..*
-audioManager
0..1
-audioManager
0..1
-encodingStrategy
0..1
-encodingStrategy
0..1
<<Java Class>>
TextToSpeechAPIFactory
texttospeechapis
TextToSpeechAPIFactory()
createTTSAPI(String):TextToSpeechAPI
-ttsAPI
0..1
<<Java Interface>>
TextToSpeechAPI
texttospeechapis
play(String):void
setVolume(int):void
setPinch(int):void
setRate(int):void
<<Java Interface>>
EncodingStrategy
encodingstrategies
encode(String):String
-ret
0..1
<<Java Class>>
StrategiesFactory
encodingstrategies
StrategiesFactory()
createEncodingStrategy(String):EncodingStrategy
getStrategy():EncodingStrategy
<<Java Class>>
FreeTTSAdapter
texttospeechapis
vm: VoiceManager
voice: Voice
FreeTTSAdapter()
play(String):void
setVolume(int):void
setPinch(int):void
setRate(int):void
<<Java Class>>
FakeTextToSpeechAPI
texttospeechapis
contentToPlay: ArrayList<String>
volumeValue: int
pinchValue: int
rateValue: int
FakeTextToSpeechAPI()
play(String):void
setVolume(int):void
setPinch(int):void
setRate(int):void
getContentToPlay():ArrayList<String>
getVolumeValue():int
getPinchValue():int
getRateValue():int
<<Java Class>>
TemplateEncoding
encodingstrategies
TemplateEncoding()
encode(String):String
mapCharacter(char):char
<<Java Class>>
Rot13Encoding
encodingstrategies
Rot13Encoding()
encode(String):String
mapCharacter(char):char
<<Java Class>>
AtBashEncoding
encodingstrategies
AtBashEncoding()
encode(String):String
mapCharacter(char):char

Class Name: CommandsFactory	
Responsibilities: ▪ Creates commands (ActionLIstener)	**Collaborations:** ▪ DocumentToSpeech ▪ EditDocument ▪ EncodedDocumentToSpeech ▪ LineToSpeech ▪ NewDocument ▪ OpenDocument ▪ PlayEncodedLine ▪ PlayReverseLine ▪ ReverseDocumentToSpeech ▪ SaveDocument ▪ TuneAudio ▪ TuneEncodings

Class Name: DocumentToSpeech	
Responsibilities: ▪ Calls method that transforms document to speech.	**Collaborations:** ▪ Document

Class Name: EditDocument	
Responsibilities: ▪ Calls method that edits document's contents.	**Collaborations:** ▪ Document

Class Name: EncodedDocumentToSpeech	
Responsibilities: ▪ Calls method that encodes document and transform it to speech.	**Collaborations:** ▪ Document

Class Name: LineToSpeech	
Responsibilities: ▪ Calls method that transforms line to speech.	**Collaborations:** ▪ Document

Class Name: NewDocument	
Responsibilities: ▪ Calls method that creates a new document.	**Collaborations:** ▪ Document

Class Name: OpenDocument	
Responsibilities: ▪ Calls method that opens a document.	**Collaborations:** ▪ Document

Class Name: PlayEncodedLine	
Responsibilities: ▪ Calls method that plays encoded line.	**Collaborations:** ▪ Document

Class Name: PlayReverseLine	
Responsibilities: ▪ Calls method that plays selected line in reverse.	**Collaborations:** ▪ Document

Class Name:ReverseDocumentToSpeech	
Responsibilities: ▪ Calls a method that play the document in reverse.	**Collaborations:** ▪ Document

Class Name: SaveDocuments	
Responsibilities: ▪ Calls a method that save a document.	**Collaborations:** ▪ Document

Class Name: TuneAudio	
Responsibilities: ▪ Tunes audio parameters.	**Collaborations:** ▪ Document

Class Name: TuneEncodings	
Responsibilities: ▪ Calls a method that tunes encodings.	**Collaborations:** ▪ Document

Class Name: AtBashEncodings	
Responsibilities: ▪ Encodes given string.	**Collaborations:** ▪ Extends TemplateEncoding

Class Name: Rot13Encoding	
Responsibilities: ▪ Encodes given string.	**Collaborations:** ▪ Extends TemplateEncoding

Class Name: EncodingStrategy (Interface)	
Responsibilities: ▪ Encode()	**Collaborations:** ▪ Implemented by TemplateEncoding

Class Name: StrategiesFactory	
Responsibilities: ▪ Create an encoding strategy.	**Collaborations:** ▪ EncodingStrategy

Class Name: TamplateEncoding	
Responsibilities: ▪ Abstract class that implements EncodingStrategy and help with TemplateMethod pattern.	**Collaborations:** ▪ EncodingStrategy

Class Name: Document	
Responsibilities: ▪ Opens document. ▪ Edit document. ▪ Save document. ▪ Plays content (reverse,encoded also). ▪ Plays line (reverse,encoded also).	**Collaborations:** ▪ Line ▪ EncodingStrategy ▪ TextToSpeechAPI

Class Name: Line	
Responsibilities: ▪ Splits words from a given String. ▪ Transforms line to speech (reverse,encoded).	**Collaborations:** ▪ TextToSpeechAPI ▪ EncodingStrategy

Class Name: FakeTextToSpeechAPI	
Responsibilities: ▪ Used for testing. ▪ Stores String and int that we want to play, or tune audio.	**Collaborations:** ▪ Implements TextToSpeechAPI

Class Name: FreeTTSAdapter	
Responsibilities: ▪ Used for adapter pattern. ▪ Transforms given string to speech. ▪ Tunes audio.	**Collaborations:** ▪ Implements TextToSpeechAPI ▪ VoiceManager ▪ Voice

Class Name: TextToSpeechAPI	
Responsibilities: ▪ play ▪ setVolume ▪ setPicth ▪ setRate	**Collaborations:** ▪ Implemented by FreeTTSAdapter and FakeTextToSpeechAPI.

Class Name: TextToSpeechAPIFactory	
Responsibilities: ▪ Create TextToSpeechAPI object.	**Collaborations:** ▪ TextToSpeechAPI

Class Name:NewFileView	
Responsibilities: ▪ Popup dialog that is used when a new document is created.	**Collaborations:** ▪ Extends JDialog

Class Name: TextToSpeechEditorView	
Responsibilities: ▪ Executes commands (ActionListener) ▪ Stores current document. ▪ GUI for the application.	**Collaborations:** ▪ Document ▪ CommandsFactory ▪ StrategiesFactory ▪ ActionListener

……

……