Text2SpeechEditor

Release Report

Lumos

Anestis Kyrkenidis 3016

Anastasia-Dimitra Giannoudi 2664

Georgios Grigoriadis-Kotsalis 3209

VERSIONS HISTORY

Date	Version	Description	Author
25/5/2020	**2nd**	**User Stories Made : 1,2,3,4,5,6,7,8,9,10,11,12,13**	Anestis Kyrkenidis Anastastia-Dimitra Giannoudi Georgios Grigoriadis-Kotsalis

1 Introduction

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

1.1 Purpose

The objective of this project is to develop a simple text editor that converts the text given by the user to speech. One of the most prominent feature of this text editor, is that it can encode the document using two different encoding strategies. The first is the AtBash cipher, which reverses the alphabet, i.e the first letter becomes last, the second becomes second to last etc. The second is the Rot13 cipher, which is a special case of the Caesar cipher. In this cipher, each letter is replaced with the letter 13th letter after itself.

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

<For the user stories included in this release specify below corresponding tests using a typical tabular form.>

2.1 Scrum team

Product Owner	Georgios
Scrum Master	Anastasia-Dimitra Giannoudi
Developme nt Team	Anestis Kyrkenidis

2.2 Sprint Backlog

<List below the user stories that have been realized in this Sprint>

User Story	[US1]
Test Class	CreateDocumentTest
Description	In this test we create a new document to see if its contents are empty.
User Story	[US2]
Test Class	EditDocumentTest
Description	In this test we edit a document to see if its contents have changed and match the text that we have given.
User Story	[US3]
Test Class	SaveDocumentTest
Description	In this test we save a document and check if the contents of the document match the contents of the file saved.
User Story	[US4]
Test Class	OpenDocumentTest
Description	In this test we open an existing file (txt) document and check if the contents of the file match the contents of the ddocument.
User Story	[US5]
Test Class	DocumentTest1
Description	In this test we create a FakeTextToSpeechAPI and then check if the contents given of the current Document, match the contents of the string that the fake object is holding when the user selects to transform the document to speech.
User Story	[US6]
Test Class	LineTest1
Description	In this test we create a FakeTextToSpeechAPI and then

	check if the contents of the line of the current Document object, match the contents of the string that the fake object is holding when the user selects to transform a specific line to speech.
User Story	[US7]
Test Class	ReverseDocumentTest
Description	In this test we create a FakeTextToSpeechAPI and then check if the reversed contents given of the currentDocument, match the contents of the string that the fake object is holding when the user selects to reverse transform the document to speech.
User Story	[US8]
Test Class	ReverseLineTest
Description	In this test we create a FakeTextToSpeechAPI and then check if the reversed contents given of the line of the current Document, match the contents of the string that the fake object is holding when the user selects to reverse transform a specific line to speech.
User Story	[US9]
Test Class	AtBashEncodeDocumenTest, Rot13EncodeDocumentTest
Description	In these tests we create a FakeTextToSpeechAPI and then encode the contents of the current Document based on the given strategy to see if they match the contents of the string that the fake object is holding when the user selects to encode the document.
User Story	[US10]
Test Class	AtBashEncodeLineTest, Rot13EncodeLineTest
Description	In these tests we create a FakeTextToSpeechAPI and then encode the contents of the line of the current Document based on the given strategy to see if they match the contents of the string that the fake object is holding when the user selects to encode a specific line.

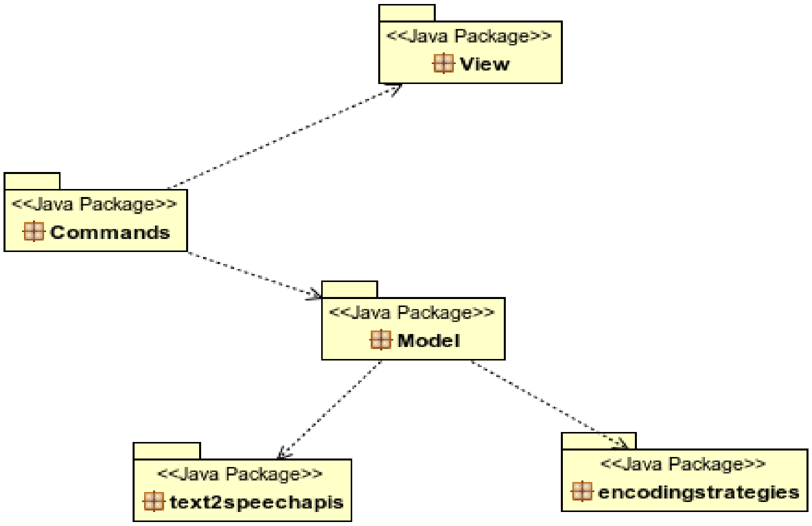
User Story	[US11]
Test Class	TuneAudioTest
Description	In this test we create a FakeTextToSpeechAPI and then call the TuneAudio command in order to change the 3 voice parameters. After that we use thems as inputs for the fake object functions (the 3 set functions) and we check to see if they match.
User Story	[US12]
Test Class	TuneEncodingTest
Description	In this test we create a TuneEncoding command and then call it to change the parameter that defines the encoding stratgey. It is AtBash by default so we change it to Rot13. Finally we call the tuneEncodingStrategy in order to change the encoding and the to see if they are different.

*Note: The User Story [US13 has been implemented and works but we could not created an acceptance test for it.

3 Design

3.1 Architecture

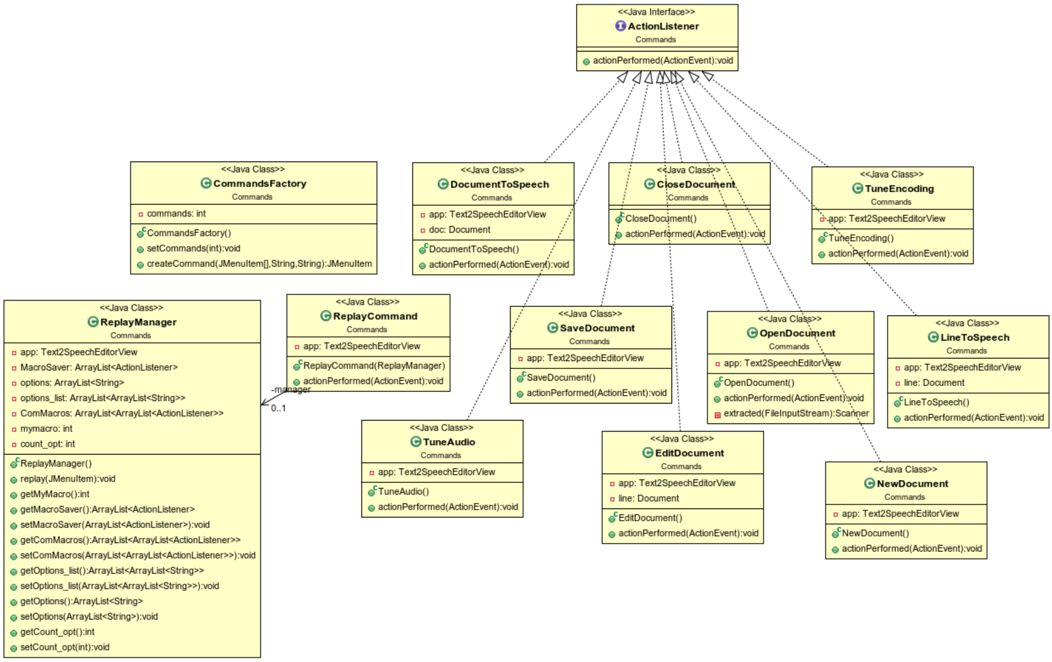
<Specify the overall architecture for this release in terms of a **UML package diagram**.>



3.2 Design

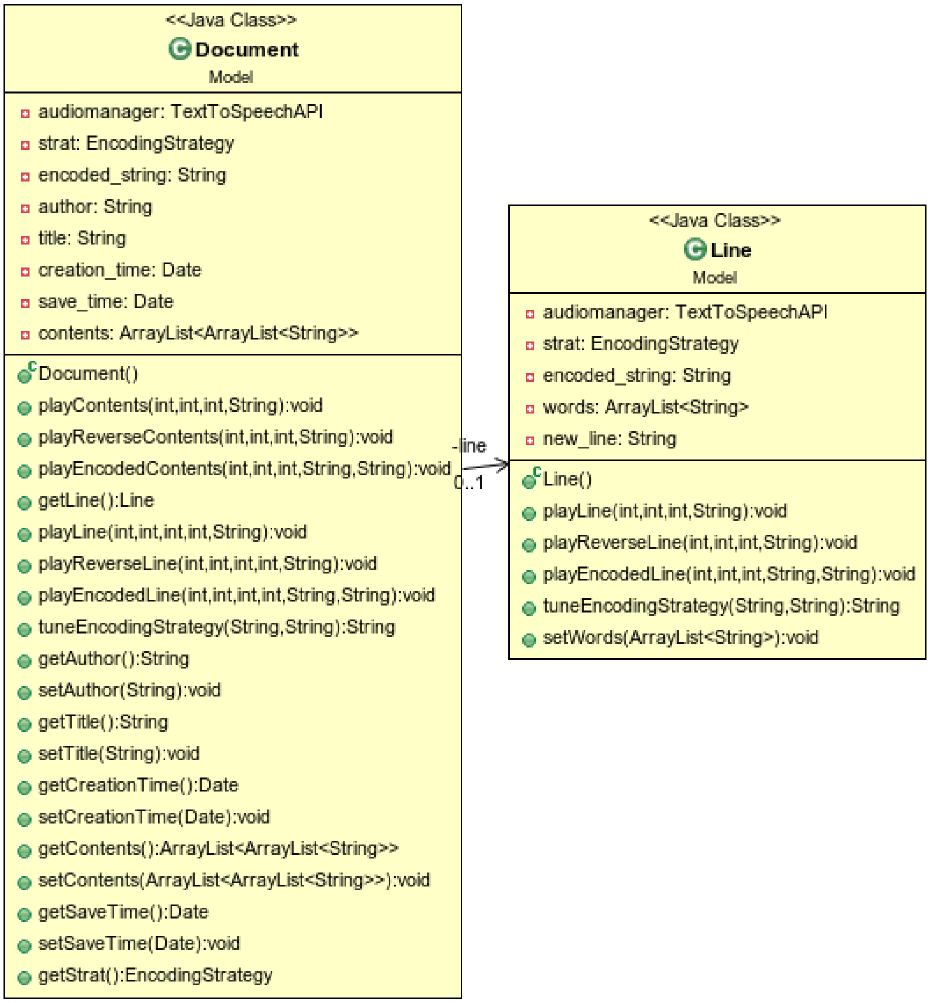
<Specify the detailed design for this release in terms of **UML class diagrams**.>

UML 1: Commands Package

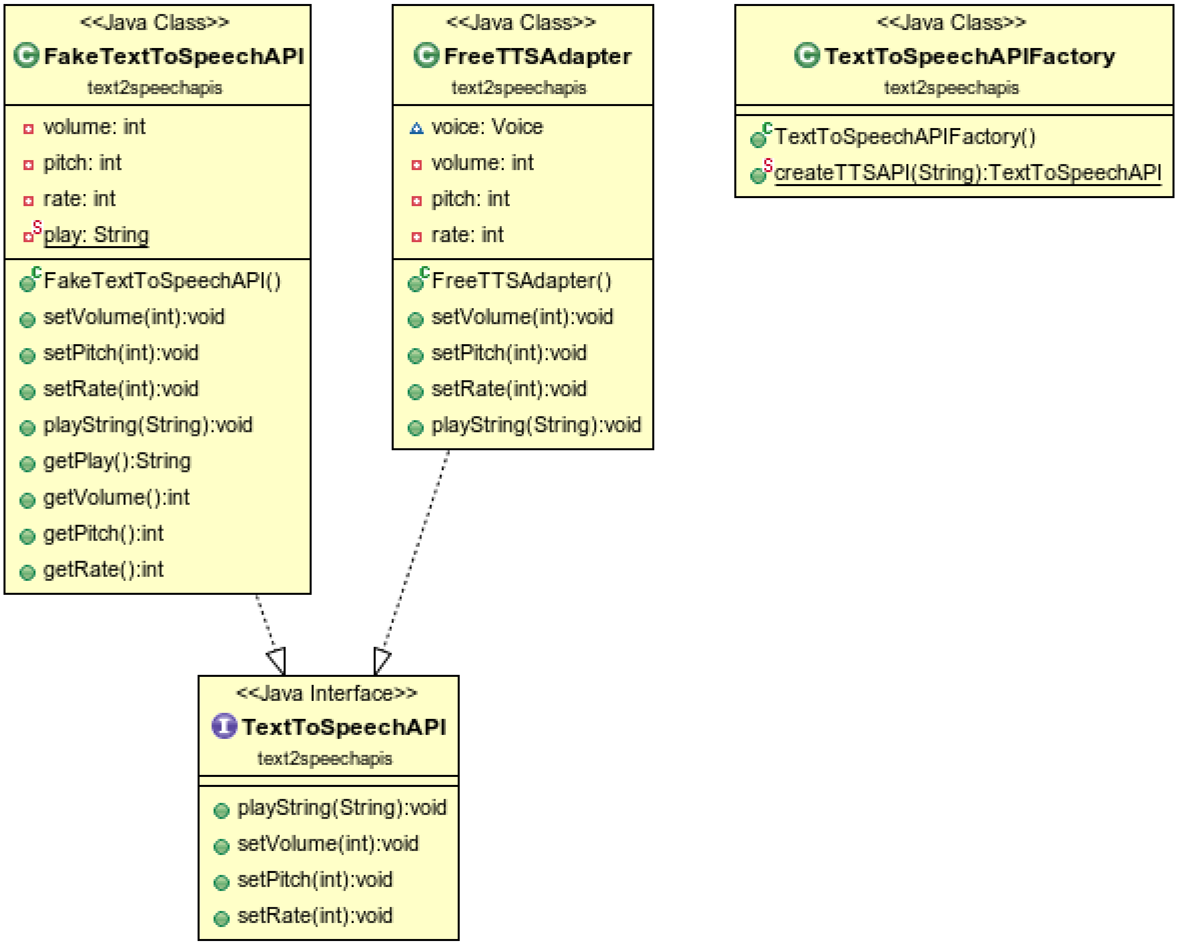


<<Java Interface>>
ActionListener
Commands
actionPerformed(ActionEvent):void
<<Java Class>>
CommandsFactory
Commands
commands: int
CommandsFactory()
setCommands(int):void
createCommand(JMenuItem[],String,String):JMenuItem
<<Java Class>>
DocumentToSpeech
Commands
app: Text2SpeechEditorView
doc: Document
DocumentToSpeech()
actionPerformed(ActionEvent):void
<<Java Class>>
CloseDocument
Commands
CloseDocument()
actionPerformed(ActionEvent):void
<<Java Class>>
TuneEncoding
Commands
app: Text2SpeechEditorView
TuneEncoding()
actionPerformed(ActionEvent):void
<<Java Class>>
ReplayManager
Commands
app: Text2SpeechEditorView
MacroSaver: ArrayList<ActionListener>
options: ArrayList<String>
options_list: ArrayList<ArrayList<String>>
ComMacros: ArrayList<ArrayList<ActionListener>>
mymacro: int
count_opt: int
ReplayManager()
replay(JMenuItem):void
getMyMacro():int
getMacroSaver():ArrayList<ActionListener>
setMacroSaver(ArrayList<ActionListener>):void
getComMacros():ArrayList<ArrayList<ActionListener>>
setComMacros(ArrayList<ArrayList<ActionListener>>):void
getOptions_list():ArrayList<ArrayList<String>>
setOptions_list(ArrayList<ArrayList<String>>):void
getOptions():ArrayList<String>
setOptions(ArrayList<String>):void
getCount_opt():int
setCount_opt(int):void
<<Java Class>>
ReplayCommand
Commands
app: Text2SpeechEditorView
ReplayCommand(ReplayManager)
actionPerformed(ActionEvent):void
-manager
0..1
<<Java Class>>
SaveDocument
Commands
app: Text2SpeechEditorView
SaveDocument()
actionPerformed(ActionEvent):void
<<Java Class>>
OpenDocument
Commands
app: Text2SpeechEditorView
OpenDocument()
actionPerformed(ActionEvent):void
extracted(FileInputStream):Scanner
<<Java Class>>
LineToSpeech
Commands
app: Text2SpeechEditorView
line: Document
LineToSpeech()
actionPerformed(ActionEvent):void
<<Java Class>>
TuneAudio
Commands
app: Text2SpeechEditorView
TuneAudio()
actionPerformed(ActionEvent):void
<<Java Class>>
EditDocument
Commands
app: Text2SpeechEditorView
line: Document
EditDocument()
actionPerformed(ActionEvent):void
<<Java Class>>
NewDocument
Commands
app: Text2SpeechEditorView
NewDocument()
actionPerformed(ActionEvent):void

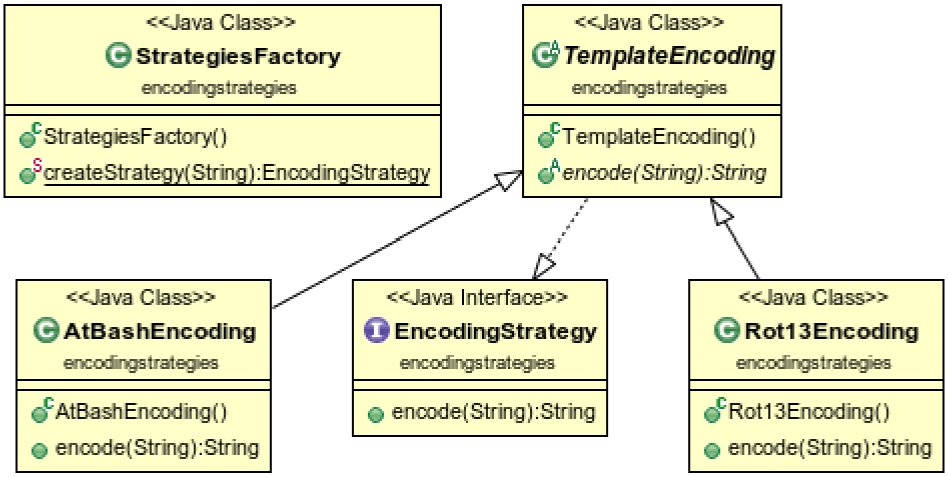
UML 2: Model Package


<<Java Class>>
Document
Model
audiomanager: TextToSpeechAPI
strat: EncodingStrategy
encoded_string: String
author: String
title: String
creation_time: Date
save_time: Date
contents: ArrayList<ArrayList<String>>
Document()
playContents(int,int,int,String):void
playReverseContents(int,int,int,String):void
playEncodedContents(int,int,int,String,String):void
getLine():Line
playLine(int,int,int,int,String):void
playReverseLine(int,int,int,int,String):void
playEncodedLine(int,int,int,int,String,String):void
tuneEncodingStrategy(String,String):String
getAuthor():String
setAuthor(String):void
getTitle():String
setTitle(String):void
getCreationTime():Date
setCreationTime(Date):void
getContents():ArrayList<ArrayList<String>>
setContents(ArrayList<ArrayList<String>>):void
getSaveTime():Date
setSaveTime(Date):void
getStrat():EncodingStrategy
-line
0..1
<<Java Class>>
Line
Model
audiomanager: TextToSpeechAPI
strat: EncodingStrategy
encoded_string: String
words: ArrayList<String>
new_line: String
Line()
playLine(int,int,int,String):void
playReverseLine(int,int,int,String):void
playEncodedLine(int,int,int,String,String):void
tuneEncodingStrategy(String,String):String
setWords(ArrayList<String>):void

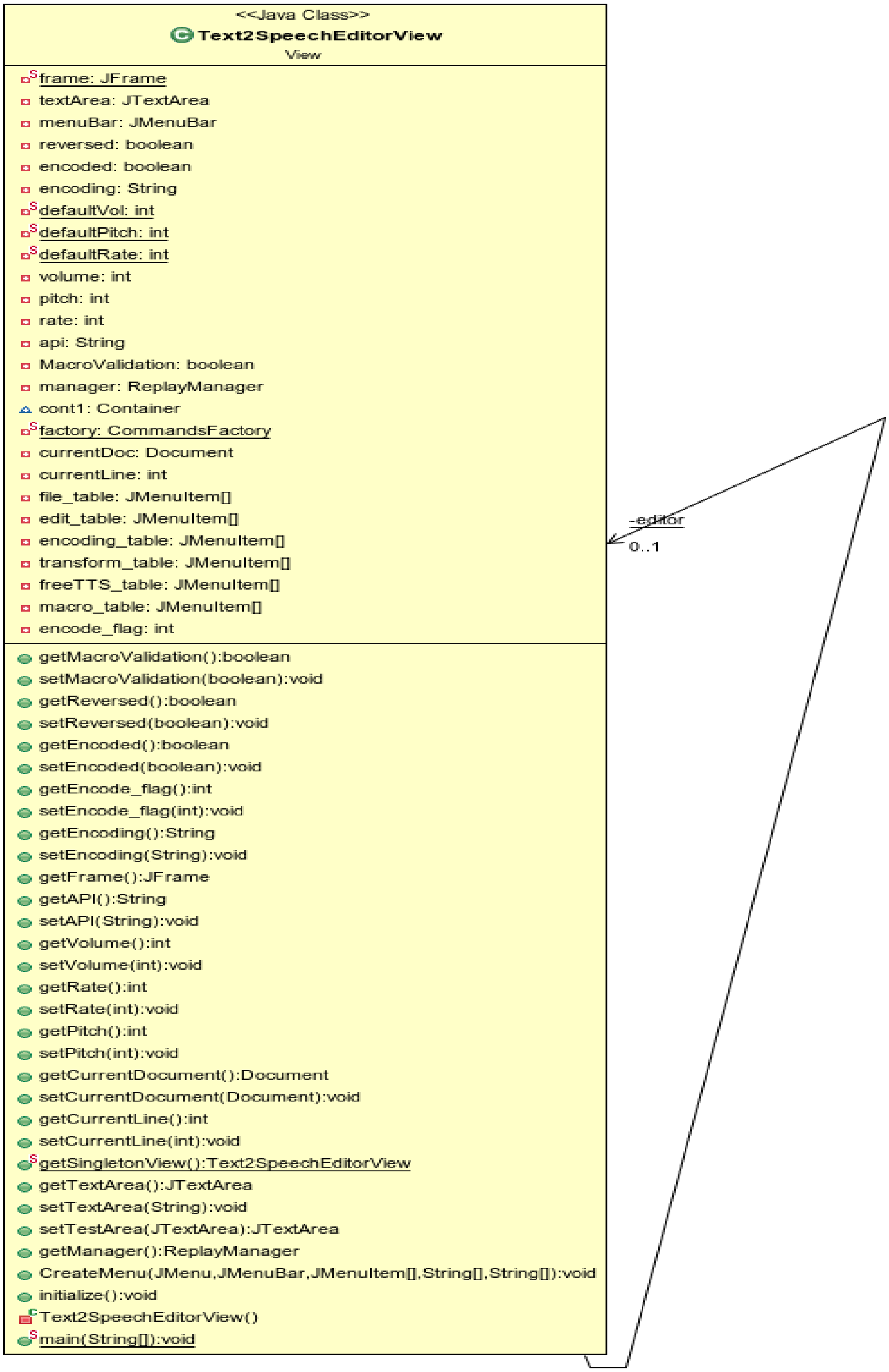
UML 3: text2speechapis Package



UML 4: encodingstrategies Package



UML 5: View Package



<Document the classes that are included in this release in terms of CRC cards according to the template that is given below.>

Class Name: Commands.ActionListener	
Responsibilities:	**Collaborations:**
▪ This class is an interface class that is setting up that each class will include the method actionPerformed.	

Class Name: Commands.CloseDocument	
Responsibilities:	**Collaborations:**
▪ Closes the document and exits the program.	

Class Name: Commands.CommandsFactory	
Responsibilities:	**Collaborations:**
▪ This class is creates the commands on the menu given by the Text2SpeechEditorView class.	

Class Name: Commands.DocumentToSpeech	
Responsibilities:	**Coollaborations:**
▪ This class is responsible for starting the operation of transforming a text to speech, either reverse transform or normal.	▪ Collaborates with the Document class. ▪ Collaborates with ReplayManager if it is part of a macro.

Class Name: Commands.EditDocument	
Responsibilities: ▪ This class is edit the contents of the document, or encodes the text given, based on the encoding strategy parameter of Text2SpeechEditorView class, and then transforms it to speech and then changes the text and the contents of the Document.	**Collaborations:** • Collaborates with the Document class. • Collaborates with the ReplayManager if EditDocument is part of a macro.

Class Name: Commands.LineToSpeech	
Responsibilities: ▪ This class is responsible for starting the operation of transforming a text to speech, either reverse transform or normal.	**Collaborations:** • Collaborates with the Document class (which in turn passes the line to the Line class so the transformation to speech begins). • Collaborates with ReplayManager if it is part of a macro.

Class Name: Commands.NewDocument	
Responsibilities: ▪ This class icreates a new Document object and opens a new window asking the user to give the author and the title of the document. ▪ Saves the author, title, sets the textArea to null and saves the date of creation.	**Collaborations:** • Collaborates with Text2SpeechEditorView class to create a new Document.

Class Name: Commands.OpenDocument	
Responsibilities:	**Collaborations:**
▪ This class is opens an existing txt file and puts its contents in a Document().	• Collaborates with Text2SpeechEditorView class to set the new contents of the Document.

Class Name: Commands.ReplayCommand	
Responsibilities:	**Collaborations:**
▪ This class is responsible for starting the recording of commands used. ▪ It is responsible for stopping the recording of commands used.	• Collaborates with the ReplayManager class to create Macro buttons for the commands.

Class Name: Commands.ReplayManager	
Responsibilities:	**Collaborations:**
▪ This class is responsible for creating the Macro buttons and sets the appropriate tooltip for the start of the Command of a macro. ▪ It stores 4 arraylists required to store the commands and the Tooltips of specific buttons that were used to call those commands.	• Collaborates with ReplayCommand, SaveDocument, EditDocument, DocumentToSpeech, LineToSpeech.

Class Name: Commands.TuneAudio	
Responsibilities:	**Collaborations:**
▪ This class changes the parameters of volume, pitch, rate for the transformation to speech.	• Collaborates with Text2SpeechEditorView class to change these parameters

Class Name: Commands.TuneEncoding	
Responsibilities:	**Collaborations:**
▪ This class changes the parameter of the encoding strategy for the encoding of the contents of the Document.	• Collaborates with Text2SpeechEditorView class to change the parameter of the encoding strategy.

Class Name: Model.Document	
Responsibilities:	**Collaborations:**
▪ This class is responsible for the Document, its contents, author, title, save and creation dates ▪ It is also responsible for calling the classes responsible for the transformation to speech and the encoding.	• Collaborates with StrategiesFactory, Text2SpeechAPIFactory. • Collaborates with the classes created from the above. • Collaborates with Line class in case the user wants to transform a specific line to speech.

Class Name: Model.Line	
Responsibilities:	**Collaborations:**
▪ This class is responsible for storing the line that is to be transformed to speech, reversed transform and encoded.	• Collaborates with Text2SpeechAPIFactory and StrategiesFactory. • Collaborates with the classes created from the above functions.

Class Name: text2speechapis.FakeTextToSpeechAPI	
Responsibilities:	**Collaborations:**
▪ This class exists only for the Junit Testing cases and is responsible for keeping the variables given to it, i.e volume, pitch, rate and the text that is to be transformed to speech.	• Collaborates with the TextToSpeechAPI interface class.

Class Name: text2speechapis.FreeTTSAdapter	
Responsibilities: ▪ This class is responsible for transform the text given to it, to speech. ▪ Changes the volume, pitch and rate of the speech.	**Collaborations:** • Collaborates with the TextToSpeechAPI interface class.

Class Name: text2speechapis.TextToSpeechAPI	
Responsibilities: ▪ This class is an interface class that is setting up that each class will include the methods setVolume, setPitch, setRate, playString.	**Collaborations:** • Collaborates with FreeTTSAdapter class and FakeTextToSpeechAPI class.

Class Name: text2speechapis.TextToSpeechAPIFactory	
Responsibilities: ▪ This class is responsible for creating the appropriate TextToSpeechAPI class based on the input given.	**Collaborations:** • Collaborates with Document, Line, which call it with the api given.

Class Name: encodingstrategies.AtBashEncoding	
Responsibilities: ▪ This class is responsible for encoding the input that it takes using the AtBash cipher.	**Collaborations:** • Collaborates with the TemplateEncoding class.

Class Name: encodingstrategies.EncodingStrategy	
Responsibilities: ▪ This class is an interface class that is setting up that each class will include the method encode.	**Collaborations:** • Collaborates with the TemplateEncoding class.

Class Name: encodingstrategies.StrategiesFactory	
Responsibilities:	**Collaborations:**
▪ This class is responsible for creating the appropriate EncodingStrategy class based on the input given.	• Collaborates with Document, Line classes, which call it with the strategy given.

Class Name: encodingstrategies.Rot13Encoding	
Responsibilities:	**Collaborations:**
▪ This class is responsible for encoding the input that it takes using the Rot13 cipher.	• Collaborates with the TemplateEncoding class.

Class Name: encodingstrategies.TemplateEncoding	
Responsibilities:	**Collaborations:**
▪ This abstract class serves as a template for the classes AtBashEncoding and Rot13Encoding.	• Collaborates with the EncodingStrategy, AtBashEncoding and the Rot13Encoding classes.