Text2SpeechEditor

Sprint No X Report

<Fill here the Team Name>

<Fill here the Group Members, Name and AM>

Βάμβας Ιωάννης Α.Μ.: 2943

Γεωργούλας Βασίλης Α.Μ.: 2954

Κολιάτος Δημήτρης Α.Μ.: 3252

VERSIONS HISTORY

Date	Version	Description	Author

1 Introduction

This document provides information concerning the **<X>** sprint of the project.

1.1 Purpose

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.>

For the **US1** our test is class **NewDocumentTest** in package Testing in folder Test. The main idea for this test case was to set a text to text area of our editor and then when call the actionListener of the button create empty document with the parameters which are expected then we see if the contents of our editor are empty.

For the **US2** our test is class **EditDocumentTest** in package Testing in folder Test. The main idea for this test case was to set a text to text area of our editor, click the button edit and get the contents of the current document. Also we set a different text to text area of our editor, click the button edit and get the contents of the current document. Finally we see if the contents of our document are same with the text that we set in second time and also we see if the get contents of the first document are different of the get contents of the second document.

For the **US3** our test class is **SaveDocumentTest** in package Testing in folder Test. The main idea for this test case was to set the title, the author, the text, the creation day and the day of the last save of the file of our editor. Then we perform the click of save button and see if the contents of the saved file are the same with the contents of our document.

For the **US4** our test class is **OpenDocumentTest** in package Testing in folder Test. The main idea for this test case was to set a text, then perform the click of open file and finally see if the text of our document are the same with the text of the file that just opened and displayed in text area of our editor.

For the **US5-US10** our test class is **SpeechTest** in package Testing in folder Test. The main idea for this test case was to set a text of 2 lines in our document and depending on the button that was clicked, see if the contents(or the content if we talk about line buttons) of our document(or the reversed or the encoded contents of our document) if are the same with the contents that we have to FakeTextToSpeechAPI for the voice display of the string(depending on the button that was clicked).

For the **US11** our test class is **TuneEncodingTest** in package Testing in folder Test. The main idea for this test case was to click button atbash and make a instance of the strategy atbash and see if the strategy that created via button atbash are the same with the instance that we made it. The same thing we do for the button rot13. Finally we check if the 2 strategies that are created via these 2 buttons and set it to our editor as encoding strategy are different.

For the **US12** our test class is **TuneAudioTest** in package Testing in folder Test. The main idea for this test case was to set the same parameter to VoiceSlider, to RateSlider and to PitchSlider and check if indeed these 3 sliders have the price that we set it.

2.1 Scrum team

Product Owner	Apostolos Zarras
Scrum Master	....
Development Team	....

2.2 Sprint Backlog

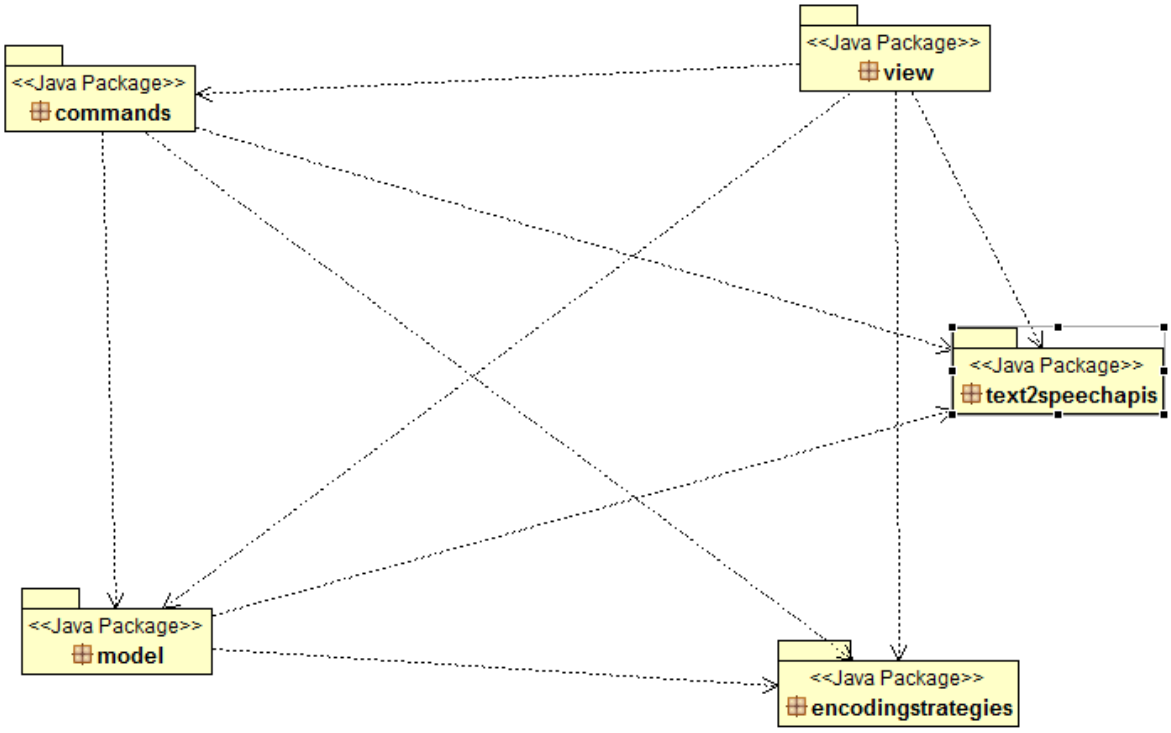
<List below the user stories that have been realized in this Sprint>

The user stories that have been implemented in this Sprint are US1-US13.

3 Design

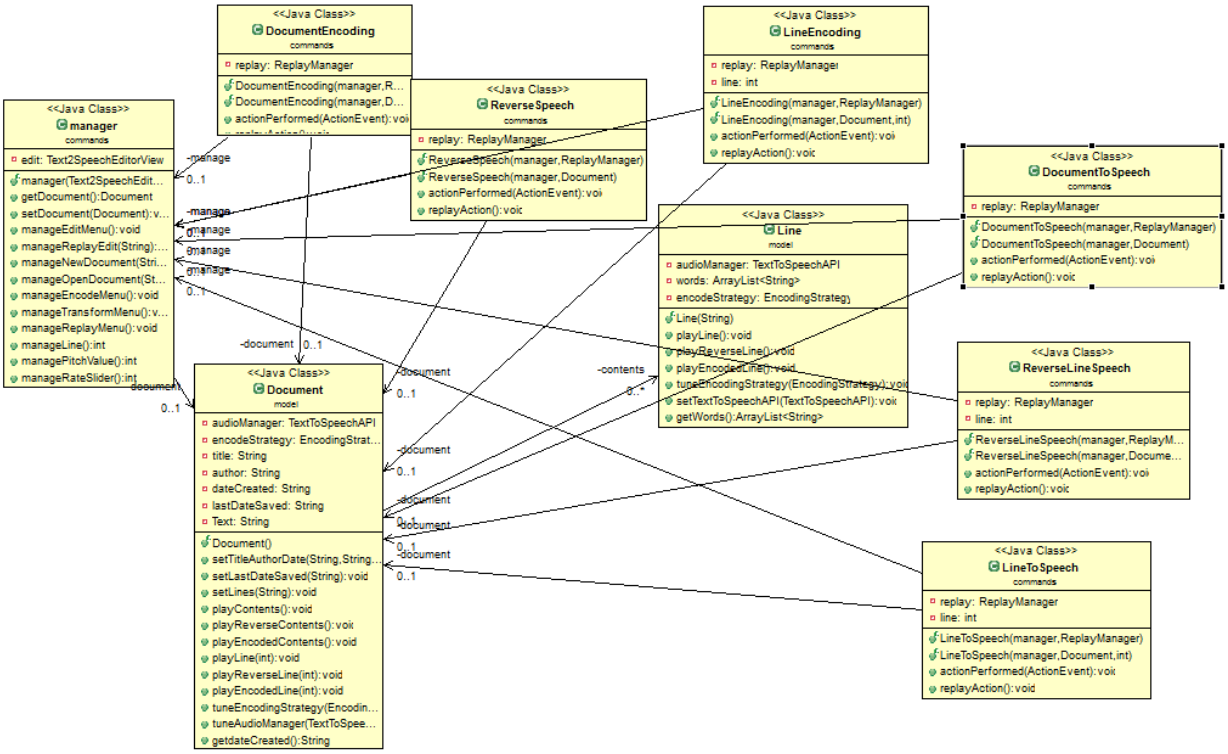
3.1 Architecture

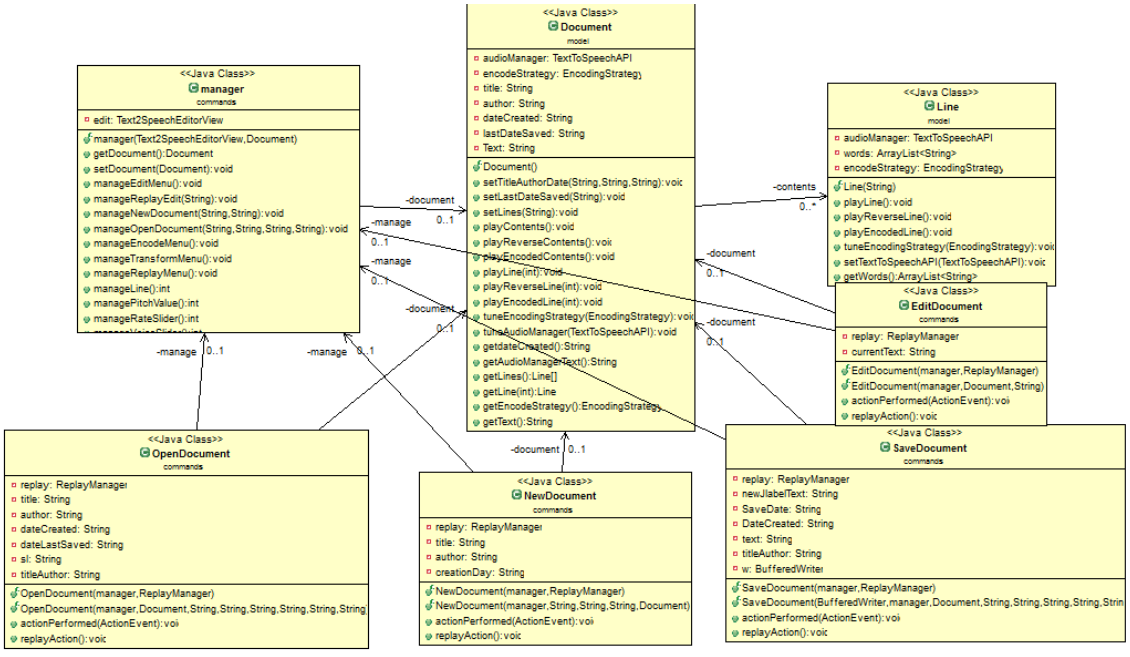
<Specify the overall architecture for this release in terms of a **UML package diagram**.>



3.2 Design

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





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

Class Name: NewDocument	
Responsibilities: ▪ Executes when user clicks the button Create Empty Document ▪ Creates a new window in which user can write the title and the author of the new Empty Document ▪ Takes the title and the author from user ▪ Creates a new empty document ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Create Empty Document and display the previous action	**Collaborations:** ▪ Gives to Document class the title, the author and the creation date of the new document ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage ▪ Calls manageNewDocument() method from manager class to do the things that are related with Text2SpeecEditorView class(enable buttons, text and set current document) ▪ If the user cancels the operation then calls cancelMenu() method from manager class that displays a window that says that user cancelled the operation

Class Name: EditDocument	
Responsibilities: ▪ Executes when user clicks the button Edit ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Edit and display the previous action	**Collaborations:** ▪ Gives to Document class the text, to “refresh” the contents of document class(more specific the Lines class objects) ▪ Manage the edit menu via manageEditMenu() of class manager ▪ Take the text from the JTextArea of the current editor via getDocument() method of manager class and getText() method from Document() class ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage

Class Name: OpenDocument	
Responsibilities: • Executer when user clicks the button open • Open the file that user selects via JFileChooser • Split the text from the file to tile, author, creationDay and the day of the last save of the file that user opens and to the contents that user writed to the file before • Creates a new Document class object • Create a new instance of this object(via copy constructor) with some parameters for later usage • Executes when user hits button replay via replayAction() method if user earlier clicks the button Open and display the previous action	**Collaborations:** ▪ Gives to Document class the author, the title, the creation day and the day of last save of the file ▪ Calls the manageOpenDocument method via manager class that set the title, the autor, the creation day, the day of tha last save of the file, the text to the right positions of our editor and gives the new instance of Document class to our editor ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage ▪ If the user cancels the operation then calls cancelMenu() method from manager class that displays a window that says that user cancelled the operation

Class name: SaveDocument	
Responsibilities: ▪ Executes when user click the button Save ▪ Save the file to the location that user selects via JFileChooser ▪ Save also to the same file the creation day, author, title and day of last save of the file ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Save and display the previous action	**Collaborations:** ▪ Get from the current editor the content of JLabel for the display of the new day of last save of the file via manageLabelText() method from manager class ▪ Get the current document that user works via getDocument() method of manager class ▪ Get the day that the file created via getDateCreated method of the Document class ▪ Get the text that was in our editor via manageTextArea from manager class that gives us the JTextArea of our editor ▪ Gives to Document class the day that user saves the file ▪ Gives an instance of the current object(via

	copy constructor) to ReplayManager class for later usage ▪ If the user cancels the operation then calls cancelMenu() method from manager class that displays a window that says that user cancelled the operation

Class name: DocumentToSpeech	
Responsibilities: ▪ Executes when user clicks the button Speech ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Speech and display the previous action	**Collaborations:** ▪ Get the current document that users works via getDocument() method of the manager class ▪ Call the method playContents() from the Document class for the voice display of the contents ▪ Call the method manageTransformMenu() of manager class for the management of the menu of button Speech ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage

Class name: LineToSpeech	
Responsibilities: ▪ Executes when user clicks the button Line Speech ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Line Speech and display the previous action	**Collaborations:** ▪ Get the current document that users works via getDocument() method of the manager class ▪ Get the current Line that user is via manageLine() method of manager class ▪ Call the method manageTransformMenu() of manager class for the management of the menu of button Line Speech ▪ Call the method playLine() of the class Document for the voice display of the line that user is ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage

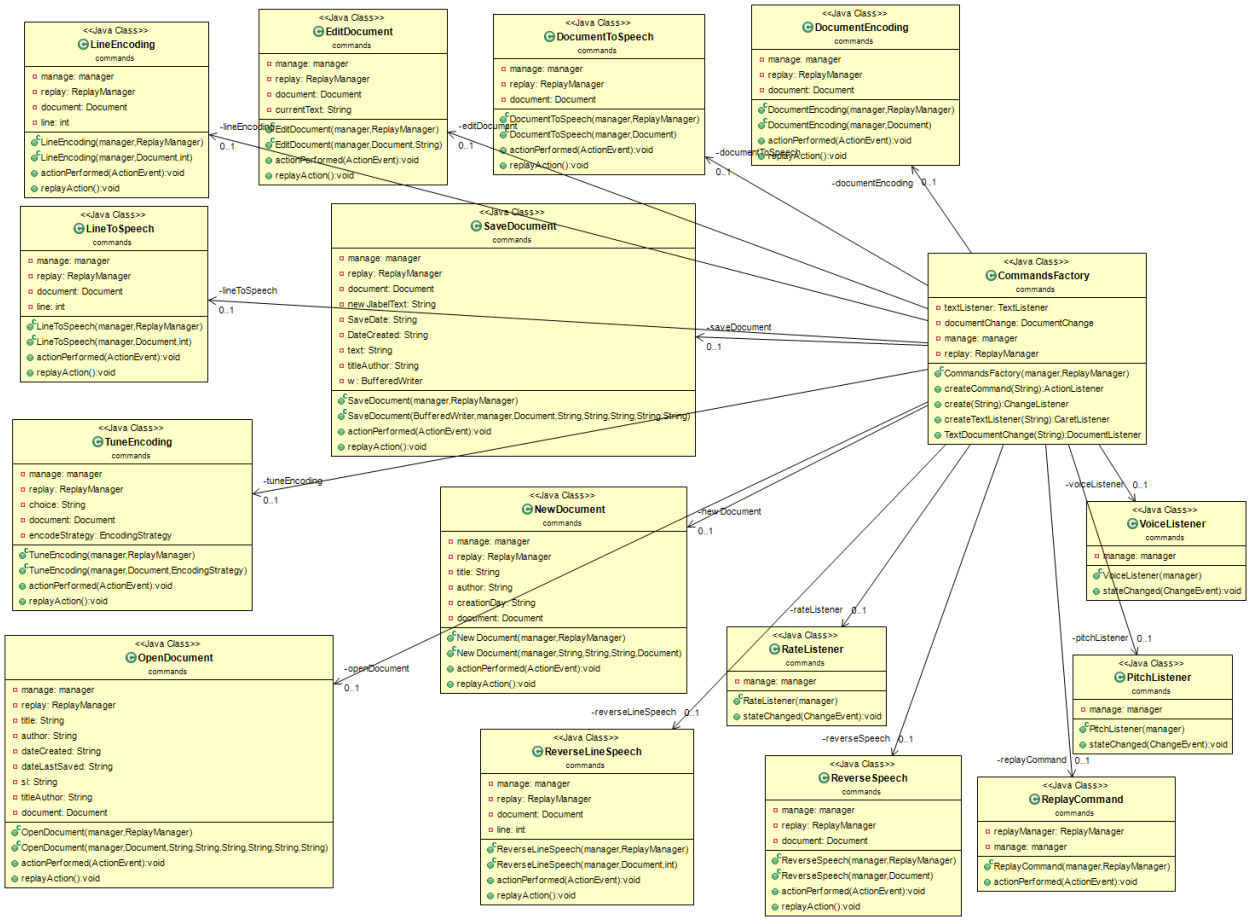
Class name: ReverseSpeech	
Responsibilities: ▪ Executes when user clicks the button Reverse Speech ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Reverse Speech and display the previous action	**Collaborations:** ▪ Get the current document that users works via getDocument() method of the manager class ▪ Call the method manageTransformMenu() of manager class for the management of the menu of button Reverse Speech ▪ Call the method playReverseContents() from the Document class for the voice display of the contents ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage

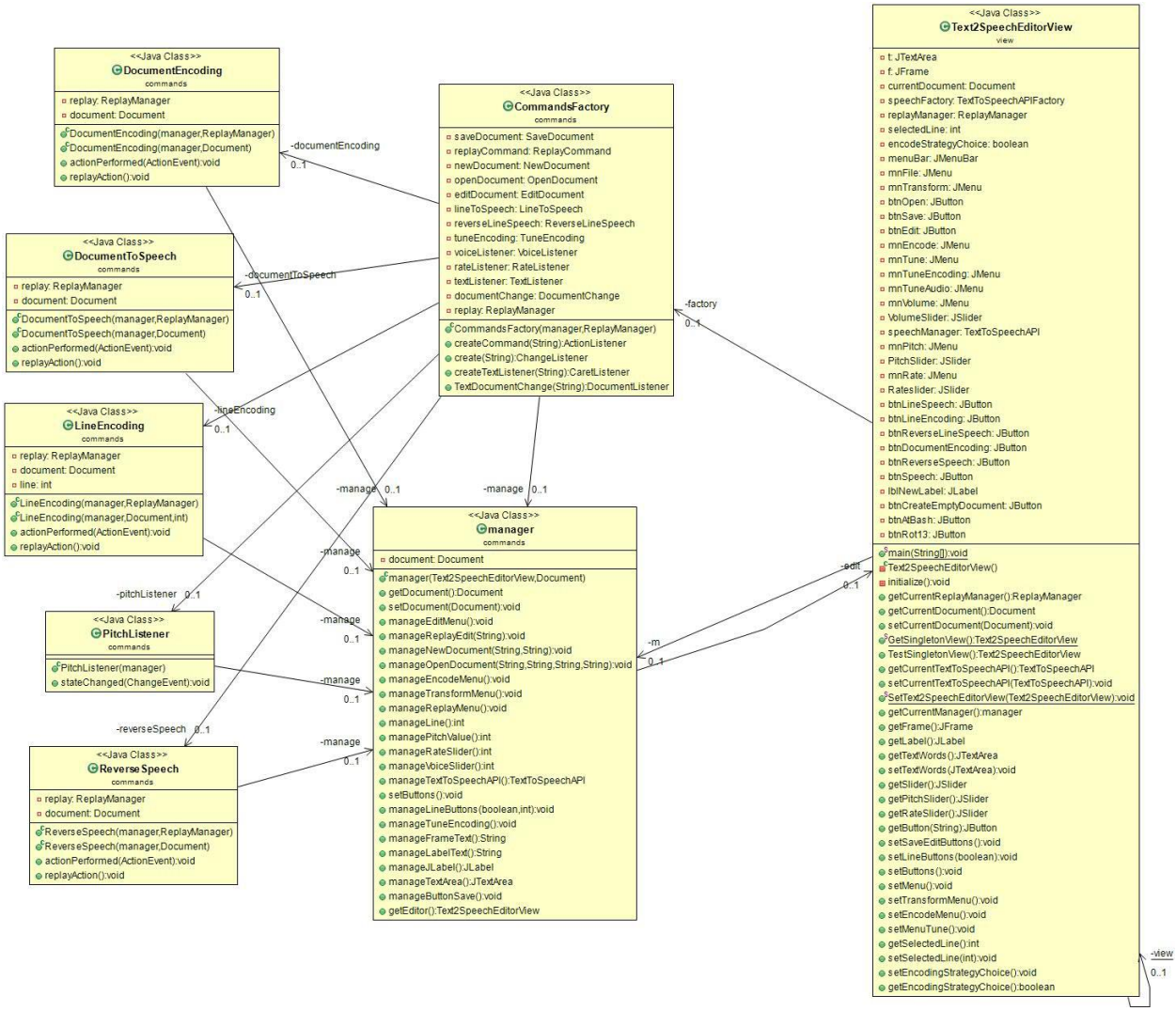
Class name: ReverseLineSpeech	
Responsibilities: ▪ Executes when user clicks the button Reverse Line Speech ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Reverse Line Speech and display the previous action	**Collaborations:** ▪ Get the current document that users works via getDocument() method of the manager class ▪ Get the current Line that user is via manageLine() method of manager class ▪ Call the method manageTransformMenu() of manager class for the management of the menu of button Reverse Line Speech ▪ Call the method playReverseLine() from the Document class for the voice display of the line that user is ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage

Class name: DocumentEncoding	
Responsibilities: ▪ Executes when user clicks the button Document Encoding ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Document Speech and display the previous action	**Collaborations:** ▪ Get the current document that users works via getDocument() method of the manager class ▪ Call the method manageEncodeMenu() of manager class for the management of the menu of button Document Speech ▪ Call the method playEncodedContents() from the Document class for the voice display of the contents ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage

Class name: LineEncoding	
Responsibilities: ▪ Executes when user clicks the button Line Encoding ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Line Encoding and display the previous action	**Collaborations:** ▪ Get the current document that users works via getDocument() method of the manager class ▪ Call the method manageEncodeMenu() of manager class for the management of the menu of button Document Speech ▪ Get the current Line that user is via manageLine() method of manager class ▪ Call the method playEncodedContents() from the Document class for the voice display of the contents ▪ Gives an instance of the current object(via copy constructor) to ReplayManager class for later usage

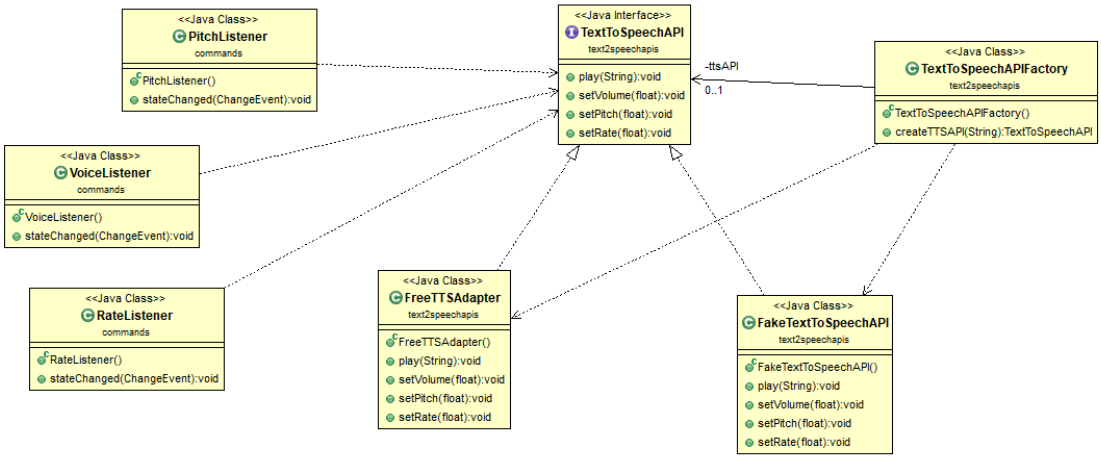
Class name: manager	
Responsibilities: ▪ Connect the buttons Action Listeners classes with the Text2SpeechEditorView class	**Collaborations:** ▪ Call the appropriate methods of the Text2SpeechEditor class depending on the button that user clicks ▪ Call methods of Text2SpeechEditorView class for the enable or disable of one or more buttons ▪ Call methods of Text2SpeechEditorView class for set the current document that user works, set the text of our editor and similar tasks ▪ Return to the ActionListeners of the buttons some fields from the Text2SpeechEditorView class like JFrame, JEditor, the line that user is and similar information that is important for the implementation of the action of the buttons


<<Java Class>>
LineEncoding
commands
manage: manager
replay: ReplayManager
document: Document
line: int
LineEncoding(manager,ReplayManager)
LineEncoding(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
EditDocument
commands
manage: manager
replay: ReplayManager
document: Document
currentText: String
EditDocument(manager,ReplayManager)
EditDocument(manager,Document,String)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
DocumentToSpeech
commands
manage: manager
replay: ReplayManager
document: Document
DocumentToSpeech(manager,ReplayManager)
DocumentToSpeech(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
DocumentEncoding
commands
manage: manager
replay: ReplayManager
document: Document
DocumentEncoding(manager,ReplayManager)
DocumentEncoding(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
LineToSpeech
commands
manage: manager
replay: ReplayManager
document: Document
line: int
LineToSpeech(manager,ReplayManager)
LineToSpeech(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
SaveDocument
commands
manage: manager
replay: ReplayManager
document: Document
newJlabelText: String
SaveDate: String
DateCreated: String
text: String
titleAuthor: String
w: BufferedWriter
SaveDocument(manager,ReplayManager)
SaveDocument(BufferedWriter,manager,Document,String,String,String,String,String)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
CommandsFactory
commands
textListener: TextListener
documentChange: DocumentChange
manage: manager
replay: ReplayManager
CommandsFactory(manager,ReplayManager)
createCommand(String):ActionListener
create(String):ChangeListener
createTextListener(String):CaretListener
TextDocumentChange(String):DocumentListener
<<Java Class>>
TuneEncoding
commands
manage: manager
replay: ReplayManager
choice: String
document: Document
encodeStrategy: EncodingStrategy
TuneEncoding(manager,ReplayManager)
TuneEncoding(manager,Document,EncodingStrategy)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
NewDocument
commands
manage: manager
replay: ReplayManager
title: String
author: String
creationDay: String
document: Document
New Document(manager,ReplayManager)
New Document(manager,String,String,String,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
VoiceListener
commands
manage: manager
VoiceListener(manager)
stateChanged(ChangeEvent):void
<<Java Class>>
OpenDocument
commands
manage: manager
replay: ReplayManager
title: String
author: String
dateCreated: String
dateLastSaved: String
sl: String
titleAuthor: String
document: Document
OpenDocument(manager,ReplayManager)
OpenDocument(manager,Document,String,String,String,String,String,String)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
RateListener
commands
manage: manager
RateListener(manager)
stateChanged(ChangeEvent):void
<<Java Class>>
PitchListener
commands
manage: manager
PitchListener(manager)
stateChanged(ChangeEvent):void
<<Java Class>>
ReverseLineSpeech
commands
manage: manager
replay: ReplayManager
document: Document
line: int
ReverseLineSpeech(manager,ReplayManager)
ReverseLineSpeech(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReverseSpeech
commands
manage: manager
replay: ReplayManager
document: Document
ReverseSpeech(manager,ReplayManager)
ReverseSpeech(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReplayCommand
commands
replayManager: ReplayManager
manage: manager
ReplayCommand(manager,ReplayManager)
actionPerformed(ActionEvent):void
-lineEncoding
0..1
-editDocument
0..1
-documentToSpeech
0..1
-documentEncoding
0..1
-lineToSpeech
0..1
-saveDocument
0..1
-tuneEncoding
0..1
-newDocument
0..1
-voiceListener
0..1
-openDocument
0..1
-rateListener
0..1
-pitchListener
0..1
-reverseLineSpeech
0..1
-reverseSpeech
0..1
-replayCommand
0..1


<<Java Class>>
DocumentEncoding
commands
replay: ReplayManager
document: Document
DocumentEncoding(manager,ReplayManager)
DocumentEncoding(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
-documentEncoding
0..1
<<Java Class>>
CommandsFactory
commands
saveDocument: SaveDocument
replayCommand: ReplayCommand
newDocument: NewDocument
openDocument: OpenDocument
editDocument: EditDocument
lineToSpeech: LineToSpeech
reverseLineSpeech: ReverseLineSpeech
tuneEncoding: TuneEncoding
voiceListener: VoiceListener
rateListener: RateListener
textListener: TextListener
documentChange: DocumentChange
replay: ReplayManager
CommandsFactory(manager,ReplayManager)
createCommand(String):ActionListener
create(String):ChangeListener
createTextListener(String):CaretListener
TextDocumentChange(String):DocumentListener
-factory
0..1
<<Java Class>>
Text2SpeechEditorView
view
t: JTextArea
f: JFrame
currentDocument: Document
speechFactory: TextToSpeechAPIFactory
replayManager: ReplayManager
selectedLine: int
encodeStrategyChoice: boolean
menuBar: JMenuBar
mnFile: JMenu
mnTransform: JMenu
btnOpen: JButton
btnSave: JButton
btnEdit: JButton
mnEncode: JMenu
mnTune: JMenu
mnTuneEncoding: JMenu
mnTuneAudio: JMenu
mnVolume: JMenu
VolumeSlider: JSlider
speechManager: TextToSpeechAPI
mnPitch: JMenu
PitchSlider: JSlider
mnRate: JMenu
Rateslider: JSlider
btnLineSpeech: JButton
btnLineEncoding: JButton
btnReverseLineSpeech: JButton
btnDocumentEncoding: JButton
btnReverseSpeech: JButton
btnSpeech: JButton
lblNewLabel: JLabel
btnCreateEmptyDocument: JButton
btnAtBash: JButton
btnRot13: JButton
main(String[]):void
Text2SpeechEditorView()
initialize():void
getCurrentReplayManager():ReplayManager
getCurrentDocument():Document
setCurrentDocument(Document):void
GetSingletonView():Text2SpeechEditorView
TestSingletonView():Text2SpeechEditorView
getCurrentTextToSpeechAPI():TextToSpeechAPI
setCurrentTextToSpeechAPI(TextToSpeechAPI):void
SetText2SpeechEditorView(Text2SpeechEditorView):void
getCurrentManager():manager
getFrame():JFrame
getLabel():JLabel
getTextWords():JTextArea
setTextWords(JTextArea):void
getSlider():JSlider
getPitchSlider():JSlider
getRateSlider():JSlider
getButton(String):JButton
setSaveEditButtons():void
setLineButtons(boolean):void
setButtons():void
setMenu():void
setTransformMenu():void
setEncodeMenu():void
setMenuTune():void
getSelectedLine():int
setSelectedLine(int):void
setEncodingStrategyChoice():void
getEncodingStrategyChoice():boolean
-view
0..1
<<Java Class>>
DocumentToSpeech
commands
replay: ReplayManager
document: Document
DocumentToSpeech(manager,ReplayManager)
DocumentToSpeech(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
-documentToSpeech
0..1
<<Java Class>>
LineEncoding
commands
replay: ReplayManager
document: Document
line: int
LineEncoding(manager,ReplayManager)
LineEncoding(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
-lineEncoding
0..1
<<Java Class>>
manager
commands
document: Document
manager(Text2SpeechEditorView,Document)
getDocument():Document
setDocument(Document):void
manageEditMenu():void
manageReplayEdit(String):void
manageNewDocument(String,String):void
manageOpenDocument(String,String,String,String):void
manageEncodeMenu():void
manageTransformMenu():void
manageReplayMenu():void
manageLine():int
managePitchValue():int
manageRateSlider():int
manageVoiceSlider():int
manageTextToSpeechAPI():TextToSpeechAPI
setButtons():void
manageLineButtons(boolean,int):void
manageTuneEncoding():void
manageFrameText():String
manageLabelText():String
manageJLabel():JLabel
manageTextArea():JTextArea
manageButtonSave():void
getEditor():Text2SpeechEditorView
-manage
0..1
-manage
0..1
-manage
0..1
-manage
0..1
-manage
0..1
-manage
0..1
-edit
0..1
-m
0..1
-pitchListener
0..1
<<Java Class>>
PitchListener
commands
PitchListener(manager)
stateChanged(ChangeEvent):void
-reverseSpeech
0..1
<<Java Class>>
ReverseSpeech
commands
replay: ReplayManager
document: Document
ReverseSpeech(manager,ReplayManager)
ReverseSpeech(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void

Class name: Text2SpeechEditorView	
Responsibilities: ▪ Initialize our editor ▪ Initialize the text area of our editor ▪ Initialize the JLabel and the JFrame of our editor ▪ Initialize all the buttons and the menu of our editor ▪ Add the action listeners of the buttons ▪ Initialize a new instance of class CommandFactory for the creation of the actionListeners ▪ Initialize a new instance of manager class for the management of the actions of the buttons for our editor ▪ Initialize a new instance of replayManager for the management of replay button ▪ Initialize a new instance of document class ▪ Initialize a new instance of TextToSpeechAPIFactory class ▪ Initialize a new instance of StrategiesFactory class for the set of the default encoding choice ▪ Sets the document that user works now ▪ Sets the text that users works on our editor ▪ Sets the line that user is in our editor ▪ Disable or enable some buttons of our editor via setSaveButton(), setLineButton(), setButtons(), setMenu(), setTransformMenu(), setEncodeMenu() and setMenuTune() method	**Collaborations:** ▪ Gives the current editor to manager class and to document class ▪ Gives the current document that user works via getCurrentDocument() to manager class ▪ Gives the current TextToSpeechAPI via getCurrentTextToSpeechAPI method to RateListener, VoiceListener, PitchListener(via manager class) and Document class ▪ Gives the current buttons, the JFrame, the current text area and the JLabel to ActionListener classes via manager class ▪ Gives the line that user is(from TextListener class) via getSelectedLine() to manager class

Class name: CommandFactory	
Responsibilities: ▪ Creates the ActionListeners of the buttons of our editor via createCommand() method ▪ Create the Text Listener via createTextListener() method ▪ Create the Document Listener via TextDocumentChange() method	**Collaboratios:** ▪ Returns the ActionListeners to current text editor ▪ Returns the Text Listener to current text editor ▪ Returns the Document Listener to current text editor



Class name: TuneVoice	
Responsibilities: ▪ Convert the voice from range of 0-100 to range 0-1 because the value of voice of freeTTS are 0-1	**Collaborations:** ▪ Get the current voice choice from the user(via manager class, manageVoiceSlider() method) ▪ Get the current TextToSpeechAPI from manager class(via manageTextToSpeechAPI() method ▪ Call the method setVolume() of FreeTTSAdapter class to set the voice that user selects

Class name: TunePitch	
Responsibilities: ▪ Convert the pitch choice of user to float	**Collaborations:** ▪ Get the current pitch choice from the user(via manager class, managePitchSlider() method) ▪ Get the current TextToSpeechAPI from manager class(via manageTextToSpeechAPI() method ▪ Call the method setPitch() of FreeTTSAdapter class to set the pitch that user selects

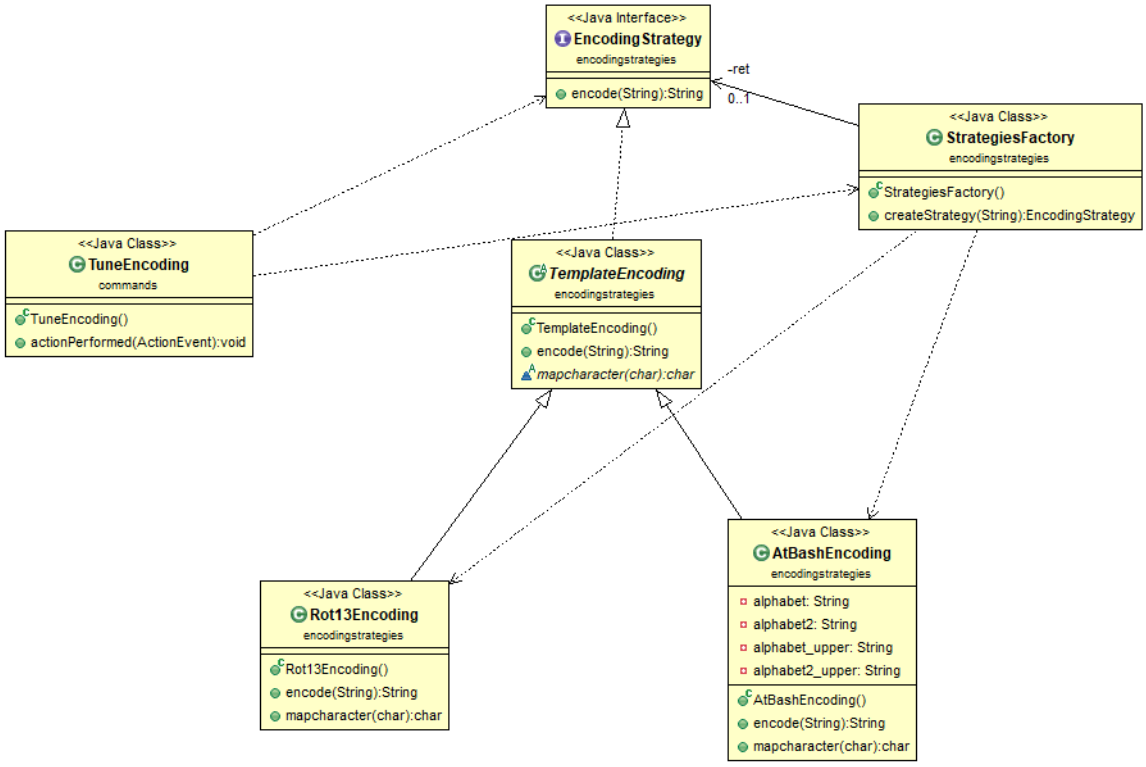
Class name: TuneRate	
Responsibilities: ▪ Convert the rate choice of user to float	**Collaborations:** ▪ Get the current rate choice from the user(via manager class, manageRateSlider() method Get the current TextToSpeechAPI ▪ Get the current TextToSpeechAPI from manager class(via manageTextToSpeechAPI() method ▪ Call the method setPitch() of FreeTTSAdapter class to set the rate that user selects

Class name: TextToSpeechAPI	
Responsibilities: ▪ Create the prototypes of the methods play(), setVolume(), setRate() and setPitch() ▪ Create the prototypes of the methods of getPlayText(), getVolume(), getRate() and getPitch() for test reasons	**Collaborations:**

Class name: TextToSpeechAPIFactory	
Responsibilities: ▪ Create a new instance of FreeTTSAdapter class or a new instance of FakeTextToSpeechAPI	**Collaborations:** ▪ Return the new instance of FreeTTSAdapter class or FakeTextToSpeechAPI to current editor(Text2SpeechEditorView class)

Class name: FreeTTSAdapter	
Responsibilities: ▪ Set the voice volume, the voice rate and the voice pitch of FreeTTS ▪ Play as volume a string	**Collaborations:** ▪ Implements the interface class TextToSpeechAPI

Class name: FakeTextToSpeechAPI	
Responsibilities: ▪ Holds the voice volume, the rate, the pitch and the text that supposed to play with voice of freeTTS	**Collaborations:** ▪ Implements the interface class TextToSpeechAPI



Class name: TuneEncoding	
Responsibilities: ▪ Get the button that user clicks(Atbash or Rot13) ▪ Creates a new instance of StrategiesFactory class for the set of the tune choice of the user ▪ Create a new instance of this object(via copy constructor) with some parameters for later usage ▪ Executes when user hits button replay via replayAction() method if user earlier clicks the button Line Encoding and display the previous action	**Collaborations:** ▪ Manage the tune menu via manageTuneEncoding() method of manager class ▪ Get the document that user works via getDocument() method of class manager ▪ Calls the createStrategy() method of StrategiesFactory class for the set of the strategy choice that user selects ▪ Calls the tuneEncodingStrategy() method of document class for the set of the strategy choice of the user to document class ▪ Calls the manageEncodeStrategy() method of manager class for the set of the strategy choice of user to editor(Text2SpeechEditorView class) via manager class

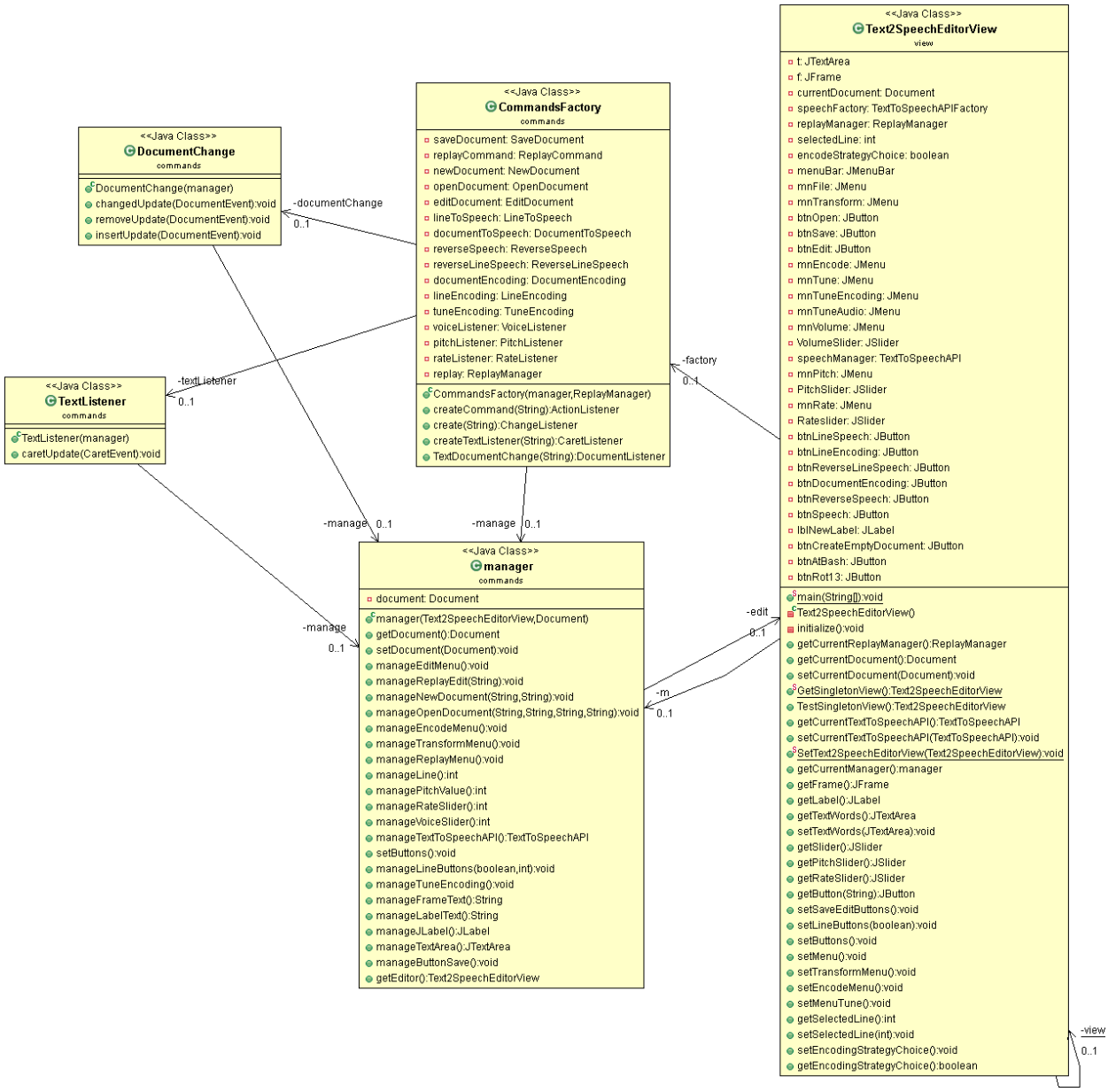
Class name: EncodingStrategy	
Responsibilities: ▪ Create the prototype of encode method	**Collaborations:**

Class name: StrategiesFactory	
Responsibilities: ▪ Create a new instance of AtBashEncoding class or a new instance of Rot13Encoding class	**Collaborations:** ▪ Return the new instance of AtBashEncoding class or Rot13Encoding class to TuneEncoding and Document class

Class name: TemplateEncoding	
Responsibilities: ▪ Create the prototypes of encode() and mapcharacter() methods	**Collaborations:**

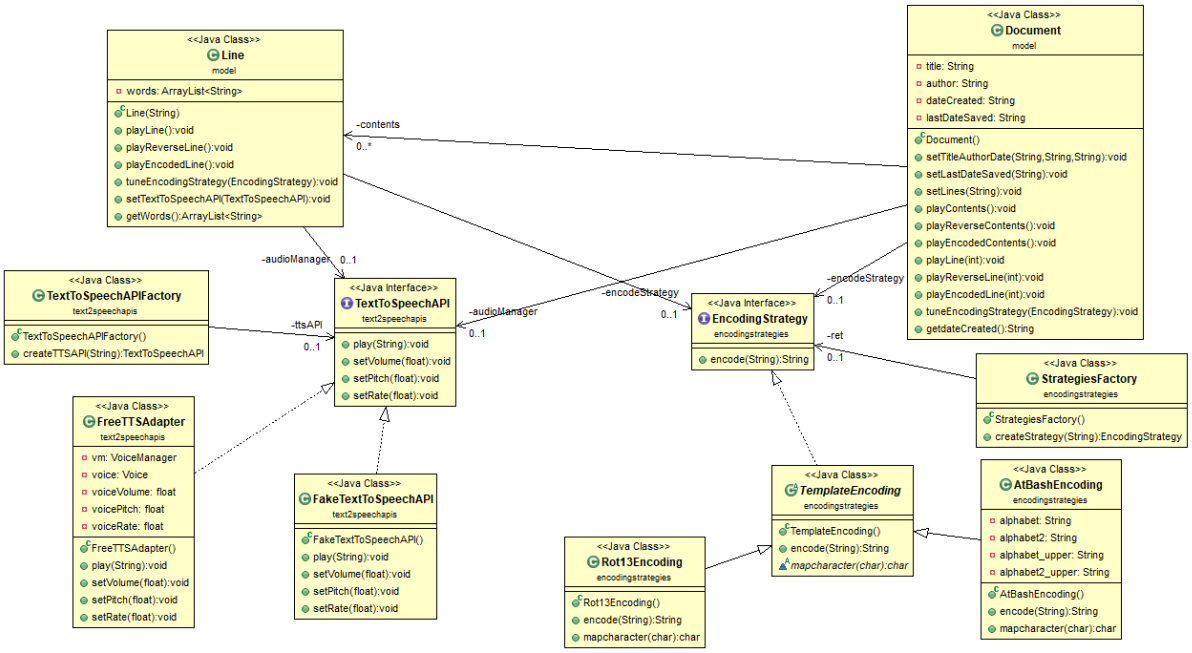
Class name: AtBashEncoding	
Responsibilities: ▪ Implements the AtBash encoding of a string ▪ Implements the encode() method and the mapcharacter() method of TemplateEncoding class	**Collaborations:** ▪ Returns the encoded string to Document or Line class(depends on who called the class)

Class name: Rot13Encoding	
Responsibilities: ▪ Implements the Rot13 encoding of a string ▪ Implements the encode() method and the mapcharacter() method of TemplateEncoding class	**Collaborations:** ▪ Returns the encoded string to Document or Line class(depends on who called the class)


<<Java Class>>
DocumentChange
commands
DocumentChange(manager)
changedUpdate(DocumentEvent):void
removeUpdate(DocumentEvent):void
insertUpdate(DocumentEvent):void
-documentChange
0..1
<<Java Class>>
CommandsFactory
commands
saveDocument: SaveDocument
replayCommand: ReplayCommand
newDocument: NewDocument
openDocument: OpenDocument
editDocument: EditDocument
lineToSpeech: LineToSpeech
documentToSpeech: DocumentToSpeech
reverseSpeech: ReverseSpeech
reverseLineSpeech: ReverseLineSpeech
documentEncoding: DocumentEncoding
lineEncoding: LineEncoding
tuneEncoding: TuneEncoding
voiceListener: VoiceListener
pitchListener: PitchListener
rateListener: RateListener
replay: ReplayManager
CommandsFactory(manager,ReplayManager)
createCommand(String):ActionListener
create(String):ChangeListener
createTextListener(String):CaretListener
TextDocumentChange(String):DocumentListener
<<Java Class>>
Text2SpeechEditorView
view
t: JTextArea
f: JFrame
currentDocument: Document
speechFactory: TextToSpeechAPIFactory
replayManager: ReplayManager
selectedLine: int
encodeStrategyChoice: boolean
menuBar: JMenuBar
mnFile: JMenu
mnTransform: JMenu
btnOpen: JButton
btnSave: JButton
btnEdit: JButton
mnEncode: JMenu
mnTune: JMenu
mnTuneEncoding: JMenu
mnTuneAudio: JMenu
mnVolume: JMenu
VolumeSlider: JSlider
speechManager: TextToSpeechAPI
mnPitch: JMenu
PitchSlider: JSlider
mnRate: JMenu
Rateslider: JSlider
btnLineSpeech: JButton
btnLineEncoding: JButton
btnReverseLineSpeech: JButton
btnDocumentEncoding: JButton
btnReverseSpeech: JButton
btnSpeech: JButton
lblNewLabel: JLabel
btnCreateEmptyDocument: JButton
btnAtBash: JButton
btnRot13: JButton
main(String[]):void
Text2SpeechEditorView()
initialize():void
getCurrentReplayManager():ReplayManager
getCurrentDocument():Document
setCurrentDocument(Document):void
GetSingletonView():Text2SpeechEditorView
TestSingletonView():Text2SpeechEditorView
getCurrentTextToSpeechAPI():TextToSpeechAPI
setCurrentTextToSpeechAPI(TextToSpeechAPI):void
SetText2SpeechEditorView(Text2SpeechEditorView):void
getCurrentManager():manager
getFrame():JFrame
getLabel():JLabel
getTextWords():JTextArea
setTextWords(JTextArea):void
getSlider():JSlider
getPitchSlider():JSlider
getRateSlider():JSlider
getButton(String):JButton
setSaveEditButtons():void
setLineButtons(boolean):void
setButtons():void
setMenu():void
setTransformMenu():void
setEncodeMenu():void
setMenuTune():void
getSelectedLine():int
setSelectedLine(int):void
setEncodingStrategyChoice():void
getEncodingStrategyChoice():boolean
-factory
0..1
<<Java Class>>
TextListener
commands
TextListener(manager)
caretUpdate(CaretEvent):void
-textListener
0..1
-manage
0..1
-manage
0..1
-manage
0..1
<<Java Class>>
manager
commands
document: Document
manager(Text2SpeechEditorView,Document)
getDocument():Document
setDocument(Document):void
manageEditMenu():void
manageReplayEdit(String):void
manageNewDocument(String,String):void
manageOpenDocument(String,String,String,String):void
manageEncodeMenu():void
manageTransformMenu():void
manageReplayMenu():void
manageLine():int
managePitchValue():int
manageRateSlider():int
manageVoiceSlider():int
manageTextToSpeechAPI():TextToSpeechAPI
setButtons():void
manageLineButtons(boolean,int):void
manageTuneEncoding():void
manageFrameText():String
manageLabelText():String
manageJLabel():JLabel
manageTextArea():JTextArea
manageButtonSave():void
getEditor():Text2SpeechEditorView
-edit
0..1
-m
0..1
-view
0..1

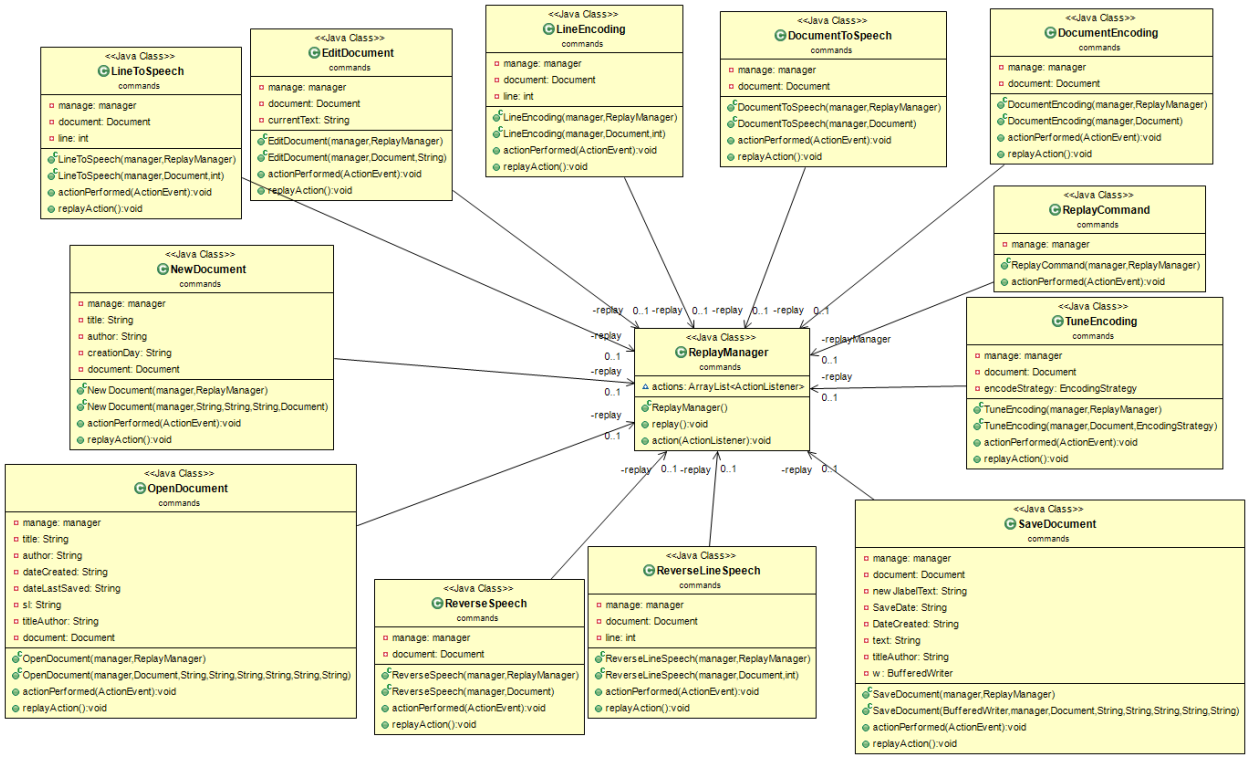
Class name: DocumentChange	
Responsibilities: ▪ This class activates when user write or remove text(or press enter or press backspasce) from our editor via removeUpdate() and insertUpdate() methods	**Collaborations:** ▪ Call the method setButtons() from manager class which is responsible for the set of document and line buttons

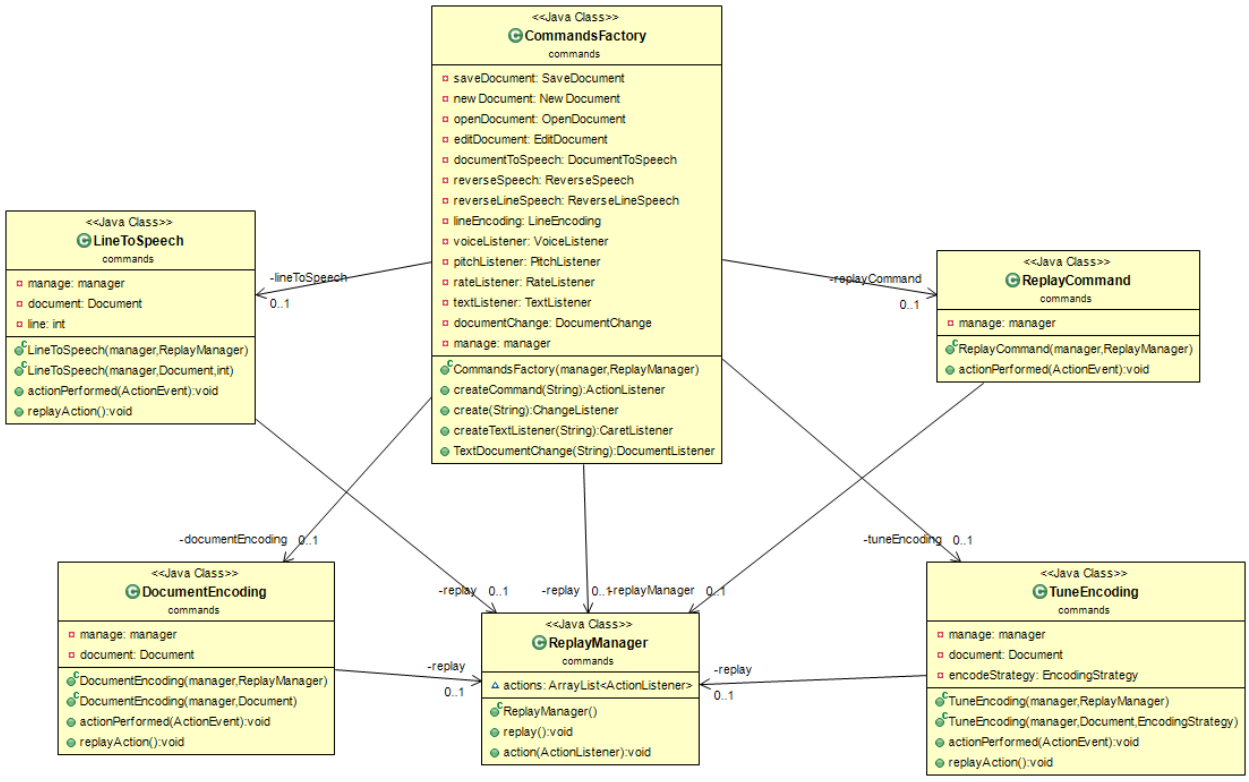
Class name: TextListener	
Responsibilities: ▪ Finds the line that user is right now and if this line have text(words)	**Collaborations:** ▪ Call the manageLineButtons() method from manager class which is responsible for the set of line buttons and for the update of the line that user is right now



Class name: Line	
Responsibilities: ▪ Separate the lines of the text to words ▪ Play(with voice) a line of the current text via playLine() method ▪ Play(with voice) a reverse line of the current text via playReverseLine() method ▪ Encode a specific line and play(with voice) the result via playEncodedLine() method ▪ Set the encodeStrategy of a specific line via tuneEncodingStrategy() method ▪ Set the current TextToSpeechAPI of a specific line via setTextToSpeechAPI() method	**Collaborations:** ▪ Returns a specific line of the text to the Document class via getWords() method ▪ Call the method play() of FreeTTSAdapter class for voice display

Class name: Document	
Responsibilities: ▪ Separate all the text to lines and make the respective objects of the Line class(via setLines method) ▪ Initialize the title, the author, the date that the file created and the the the date of the last save of the current document ▪ Play the contents of the textArea via playContents() method ▪ Play the contents in reversed way via playReverseContents() method ▪ Encode the contents of the textArea and play the result via playEncodedContents() method ▪ Returns the text of our document via getText() method ▪ Returns the date that the file was created via getdateCreated() method ▪ Set the encoding strategy of the document via tuneEncodingStrategy method ▪ Set the current TextToSpeechAPI of the document via tuneAudioManager method	**Collaborations:** ▪ Set the encoding strategy of all the Lines object of the document via tuneEncodingStrategy of the Line object ▪ Set the current TextToSpeechAPI of all the Lines object of the document via tuneEncodingStrategy of the Line object ▪ Call the method playLine() of the object Line for the voice display of a specific Line of our document ▪ Call the method playReverseLine() of the object Line for the voice display(of a reversed line) of a specific Line of our document ▪ Call the method playEncodedLine() of the object Line for the voice display(of a encoded line) of a specific Line of our document ▪ Get a specific Line of our text with the call of the getWords() method of Line class and turns the line to a strint text for the voice display of all the text(via playContents(), playReverseContents() and playEncodedContents()) ▪ Call the method play() of FreeTTSAdapter class for voice display ▪ Returns a specific Line object or all the Lines objects of the document class for test reasons


<<Java Class>>
LineToSpeech
commands
manage: manager
document: Document
line: int
LineToSpeech(manager,ReplayManager)
LineToSpeech(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
EditDocument
commands
manage: manager
document: Document
currentText: String
EditDocument(manager,ReplayManager)
EditDocument(manager,Document,String)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
LineEncoding
commands
manage: manager
document: Document
line: int
LineEncoding(manager,ReplayManager)
LineEncoding(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
DocumentToSpeech
commands
manage: manager
document: Document
DocumentToSpeech(manager,ReplayManager)
DocumentToSpeech(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
DocumentEncoding
commands
manage: manager
document: Document
DocumentEncoding(manager,ReplayManager)
DocumentEncoding(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReplayCommand
commands
manage: manager
ReplayCommand(manager,ReplayManager)
actionPerformed(ActionEvent):void
<<Java Class>>
NewDocument
commands
manage: manager
title: String
author: String
creationDay: String
document: Document
New Document(manager,ReplayManager)
New Document(manager,String,String,String,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReplayManager
commands
actions: ArrayList<ActionListener>
ReplayManager()
replay():void
action(ActionListener):void
<<Java Class>>
TuneEncoding
commands
manage: manager
document: Document
encodeStrategy: EncodingStrategy
TuneEncoding(manager,ReplayManager)
TuneEncoding(manager,Document,EncodingStrategy)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
OpenDocument
commands
manage: manager
title: String
author: String
dateCreated: String
dateLastSaved: String
st: String
titleAuthor: String
document: Document
OpenDocument(manager,ReplayManager)
OpenDocument(manager,Document,String,String,String,String,String,String)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReverseSpeech
commands
manage: manager
document: Document
ReverseSpeech(manager,ReplayManager)
ReverseSpeech(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReverseLineSpeech
commands
manage: manager
document: Document
line: int
ReverseLineSpeech(manager,ReplayManager)
ReverseLineSpeech(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
SaveDocument
commands
manage: manager
document: Document
new JlabelText: String
SaveDate: String
DateCreated: String
text: String
titleAuthor: String
w : BufferedWriter
SaveDocument(manager,ReplayManager)
SaveDocument(BufferedWriter,manager,Document,String,String,String,String,String)
actionPerformed(ActionEvent):void
replayAction():void
-replay
0..1
-replayManager
0..1


<<Java Class>>
CommandsFactory
commands
saveDocument: SaveDocument
new Document: New Document
openDocument: OpenDocument
editDocument: EditDocument
documentToSpeech: DocumentToSpeech
reverseSpeech: ReverseSpeech
reverseLineSpeech: ReverseLineSpeech
lineEncoding: LineEncoding
voiceListener: VoiceListener
pitchListener: PitchListener
rateListener: RateListener
textListener: TextListener
documentChange: DocumentChange
manage: manager
CommandsFactory(manager,ReplayManager)
createCommand(String):ActionListener
create(String):ChangeListener
createTextListener(String):CaretListener
TextDocumentChange(String):DocumentListener
-lineToSpeech
0..1
-replayCommand
0..1
-documentEncoding
0..1
-tuneEncoding
0..1
-replay
0..1
-replayManager
0..1
<<Java Class>>
LineToSpeech
commands
manage: manager
document: Document
line: int
LineToSpeech(manager,ReplayManager)
LineToSpeech(manager,Document,int)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReplayCommand
commands
manage: manager
ReplayCommand(manager,ReplayManager)
actionPerformed(ActionEvent):void
<<Java Class>>
DocumentEncoding
commands
manage: manager
document: Document
DocumentEncoding(manager,ReplayManager)
DocumentEncoding(manager,Document)
actionPerformed(ActionEvent):void
replayAction():void
<<Java Class>>
ReplayManager
commands
actions: ArrayList<ActionListener>
ReplayManager()
replay():void
action(ActionListener):void
<<Java Class>>
TuneEncoding
commands
manage: manager
document: Document
encodeStrategy: EncodingStrategy
TuneEncoding(manager,ReplayManager)
TuneEncoding(manager,Document,EncodingStrategy)
actionPerformed(ActionEvent):void
replayAction():void

Class name: ReplayCommand	
Responsibilities: ▪ Executes when user clicks the button replay	**Collaborations:** ▪ Call the method replay() of replayManager method for the execution of the buttons that was clicked before ▪ Call the method manageReplayMenu() of the class manager for the management of the menu of the replay button

Class name: ReplayManager	
Responsibilities: ▪ Add the actionsListeners of the buttons that has executed until now to an arrayList via method action	**Collaborations:** ▪ Get the actionListeners of the button that has executed until the user clicks the button replay and call the appropriate actionPerformed of these actionListeners