import UIKit

class AddSongViewController: UIViewController {

@IBOutlet weak var songInput: UITextField!

@IBOutlet weak var artistInput: UITextField!

@IBOutlet weak var lyricsInput: UITextView!

//add button clicked

@IBAction func addSongButton(\_ sender: UIButton) {

//when button clicked save title and artist into array

if (songInput.text != ""){

songTitle.append(songInput.text!)

artist.append(artistInput.text!)

let defaults = UserDefaults.standard

defaults.set(songTitle, forKey: "SongList")

let defaultsArtist = UserDefaults.standard

defaultsArtist.set(artist, forKey: "Artist")

}

//when button pressed write on file or create file

let tabFile = songInput.text!

let DocumentDirURL = try! FileManager.default.url(for: .documentDirectory, in: .userDomainMask, appropriateFor: nil, create: true)

let fileURL = DocumentDirURL.appendingPathComponent(tabFile).appendingPathExtension("txt")

let writeString = lyricsInput.text!

do{

try writeString.write(to: fileURL, atomically: true, encoding: String.Encoding.utf8)

} catch let error as NSError {

}

//reset fields

songInput.text = ""

artistInput.text = ""

lyricsInput.text = ""

//removing first responder status

self.songInput.resignFirstResponder()

self.artistInput.resignFirstResponder()

self.lyricsInput.resignFirstResponder()

}

//Give up first responder at touch screen

override func touchesBegan(\_ touches: Set<UITouch>, with event: UIEvent?) {

self.view.endEditing(true)

}

override func viewDidLoad() {

super.viewDidLoad()

//setting up text view border temporarily

lyricsInput.layer.borderWidth = 1

self.lyricsInput.layer.borderColor = UIColor(red: 214/255, green: 214/255, blue: 214/255, alpha: 1).cgColor

let gestureRecognizer = UITapGestureRecognizer(target: self, action: #selector(hello))

gestureRecognizer.numberOfTapsRequired = 2

songInput.addGestureRecognizer(gestureRecognizer)

}

func hello(){

artistInput.text = "Hello"

}

//covert lyrics when convert button is pressed

@IBAction func convertButton(\_ sender: UIButton) {

let fileName = "chordsList"

let DocDirURL = try! FileManager.default.url(for: .documentDirectory, in: .userDomainMask, appropriateFor: nil, create: true)

let fileURL = DocDirURL.appendingPathComponent(fileName).appendingPathExtension("txt")

var chordsString = ""

do{

chordsString = try String(contentsOf: fileURL)

let arrayOfChords = chordsString.components(separatedBy: ",")

var StringArray = lyricsInput.text.components(separatedBy: " ")

let arrayCheck = StringArray

print(StringArray)

for currentChord in arrayOfChords{ //for loop to go through every chord

for i in 0..<StringArray.count{ //for loop to go through every word of the lyrics

if currentChord.characters.count <= StringArray[i].characters.count{

//check to see if it is the first loop

if i == 0{

//check to see if cell is empty

if StringArray[i] != ""{

//check to see if the string hasn't been modified already

if arrayCheck[i] == StringArray[i]{

//check to see if the chord is in a cell by itself

if currentChord.characters.count == StringArray[i].characters.count{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">")

}else {

let arrayCell = StringArray[i]

let removable = "\n"

var noChordString = StringArray[i]

let copyOfCurrentString = StringArray[i]

var checkString2 = ""

var checkString = ""

var countOfRepeated = 0

var countForOffSet = 0

var positionOfChordVar = ""

//range to find only chord

let numChar = countForOffSet + currentChord.characters.count

let startPos3 = copyOfCurrentString.index(StringArray[i].startIndex, offsetBy: 0)

let endPos3 = copyOfCurrentString.index(startPos3, offsetBy: numChar)

let rangePos3 = startPos3..<endPos3

let string = copyOfCurrentString.substring(with: rangePos3)

noChordString = noChordString.replacingOccurrences(of: currentChord, with: "", options: .regularExpression)

if countOfRepeated == 0 {

//get the position of the start of the chord

for var i in 0..<StringArray[i].characters.count{

let startPositionOfChord = arrayCell.index(arrayCell.startIndex, offsetBy: i)

let endPositionOfChord = arrayCell.index(startPositionOfChord, offsetBy: 1)

let rangeOfPositionOfChord = startPositionOfChord..<endPositionOfChord

positionOfChordVar = arrayCell.substring(with: rangeOfPositionOfChord)

if positionOfChordVar == String(currentChord.characters.first!) {

countForOffSet = i

break

}

}

//get the position of the character after the chord to see if it's \n if the chord is not at the beggining of string

let startPos1 = copyOfCurrentString.index(StringArray[i].startIndex, offsetBy: countForOffSet)

let endPos1 = copyOfCurrentString.index(startPos1, offsetBy: 1)

let rangeOfPos1 = startPos1..<endPos1

checkString2 = copyOfCurrentString.substring(with: rangeOfPos1)

//get the first letter of the chord

let firstLetterofChord = String(currentChord.characters.first!)

if StringArray[i].characters.last == "\n"{

//get the position of the character after the chord to see if it's \n, if the chord is at the beggining of string

let startPos2 = noChordString.index(StringArray[i].startIndex, offsetBy: countForOffSet)

let endPos2 = noChordString.index(startPos2, offsetBy: 1)

let rangeOfPos2 = startPos2..<endPos2

checkString = noChordString.substring(with: rangeOfPos2)

if countForOffSet + currentChord.characters.count <= StringArray[i].characters.count{

if firstLetterofChord == checkString2 && checkString == removable{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">")

countOfRepeated += 1

}

}

} else{

if StringArray[i].characters.last == currentChord.characters.last{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">")

}

}

}

//check to see if the chord is at the beggining of the string

if StringArray[i].characters.first == currentChord.characters.first{

//get the position of the character after the chord to see if it's \n, if the chord is at the beggining of string

let startPos4 = noChordString.index(StringArray[i].startIndex, offsetBy: countForOffSet)

let endPos4 = noChordString.index(startPos4, offsetBy: 1)

let rangePos4 = startPos4..<endPos4

checkString = noChordString.substring(with: rangePos4)

if checkString == removable{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">", range: rangePos3)

}

}

}

}

}

} else {

//check to see if it is the last cell

if i == StringArray.count - 1{

} else{

//check to see if cell is empty

if StringArray[i] != ""{

//check to see if the string hasn't been modified already

if arrayCheck[i] == StringArray[i]{

//check to see if the chord is in a cell by itself

if currentChord.characters.count == StringArray[i].characters.count{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">")

}else{

let arrayCell = StringArray[i]

let removable = "\n"

var noChordString = StringArray[i]

let copyOfCurrentString = StringArray[i]

var checkString2 = ""

var checkString = ""

var countOfRepeated = 0

var countForOffSet = 0

var positionOfChordVar = ""

//range to find only chord

let numChar = countForOffSet + currentChord.characters.count

let startPos3 = copyOfCurrentString.index(StringArray[i].startIndex, offsetBy: 0)

let endPos3 = copyOfCurrentString.index(startPos3, offsetBy: numChar)

let rangePos3 = startPos3..<endPos3

let string = copyOfCurrentString.substring(with: rangePos3)

noChordString = noChordString.replacingOccurrences(of: currentChord, with: "", options: .regularExpression)

if countOfRepeated == 0 {

//get the position of the start of the chord

for var i in 0..<StringArray[i].characters.count{

let startPositionOfChord = arrayCell.index(arrayCell.startIndex, offsetBy: i)

let endPositionOfChord = arrayCell.index(startPositionOfChord, offsetBy: 1)

let rangeOfPositionOfChord = startPositionOfChord..<endPositionOfChord

positionOfChordVar = arrayCell.substring(with: rangeOfPositionOfChord)

if positionOfChordVar == String(currentChord.characters.first!) {

countForOffSet = i

break

}

}

//get the position of the character after the chord to see if it's \n if the chord is not at the beggining of string

let startPos1 = copyOfCurrentString.index(StringArray[i].startIndex, offsetBy: countForOffSet)

let endPos1 = copyOfCurrentString.index(startPos1, offsetBy: 1)

let rangeOfPos1 = startPos1..<endPos1

checkString2 = copyOfCurrentString.substring(with: rangeOfPos1)

//get the first letter of the chord

let firstLetterofChord = String(currentChord.characters.first!)

if StringArray[i].characters.last == "\n"{

//get the position of the character after the chord to see if it's \n, if the chord is at the beggining of string

let startPos2 = noChordString.index(StringArray[i].startIndex, offsetBy: countForOffSet)

let endPos2 = noChordString.index(startPos2, offsetBy: 1)

let rangeOfPos2 = startPos2..<endPos2

checkString = noChordString.substring(with: rangeOfPos2)

if countForOffSet + currentChord.characters.count <= StringArray[i].characters.count{

if firstLetterofChord == checkString2 && checkString == removable{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">")

countOfRepeated += 1

}

}

} else{

if StringArray[i].characters.last == currentChord.characters.last{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">")

}

}

}

//check to see if the chord is at the beggining of the string

if StringArray[i].characters.first == currentChord.characters.first{

//get the position of the character after the chord to see if it's \n, if the chord is at the beggining of string

let startPos4 = noChordString.index(StringArray[i].startIndex, offsetBy: countForOffSet)

let endPos4 = noChordString.index(startPos4, offsetBy: 1)

let rangePos4 = startPos4..<endPos4

checkString = noChordString.substring(with: rangePos4)

if checkString == removable{

StringArray[i] = StringArray[i].replacingOccurrences(of: currentChord, with: "<" + currentChord + ">", range: rangePos3)

}

}

}

}

}

}

}

}

}

}

print(StringArray)

lyricsInput.text = ""

for temp in StringArray{

lyricsInput.text = lyricsInput.text.appending(temp + " ")

}

}catch let error as NSError{

}

}

}