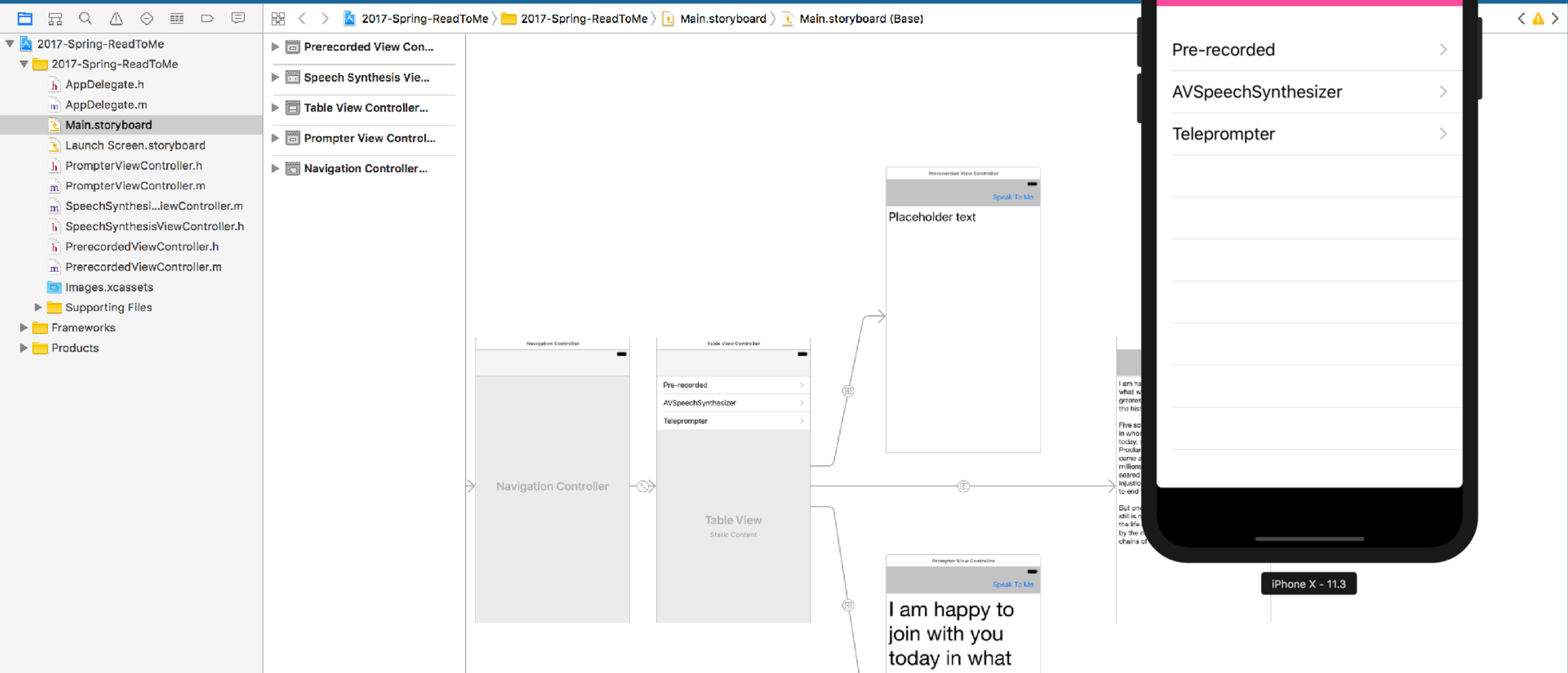
THE UNIVERSITY OF CHICAGO

ADVANCED iOS APPLICATION DEVELOPMENT

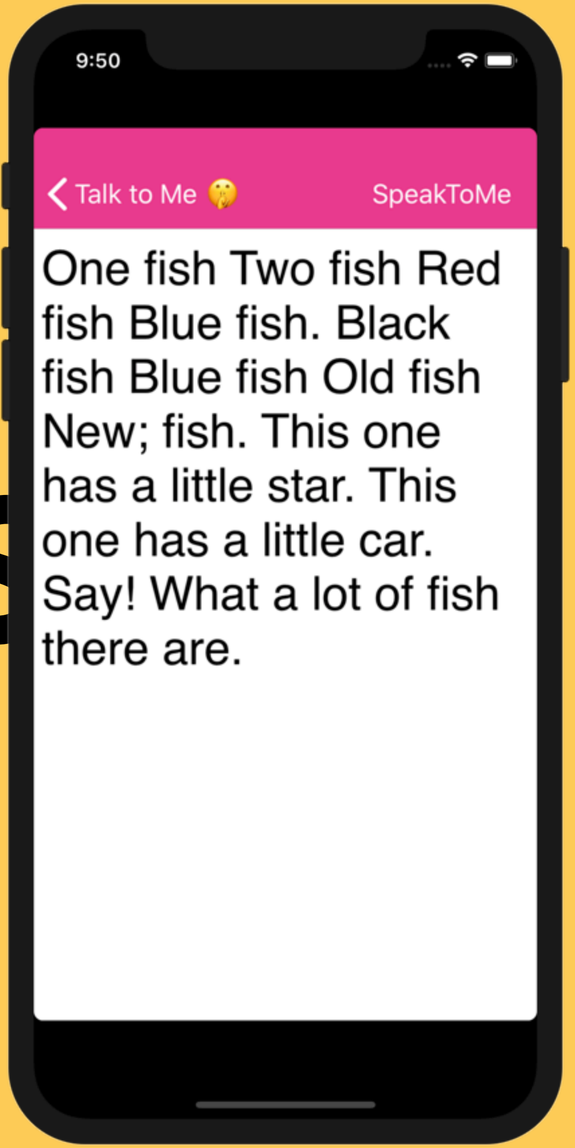
MPCS 51032 • SPRING 2020 • SESSION 1E

READ-TO-ME DEMO

READ-TO-ME

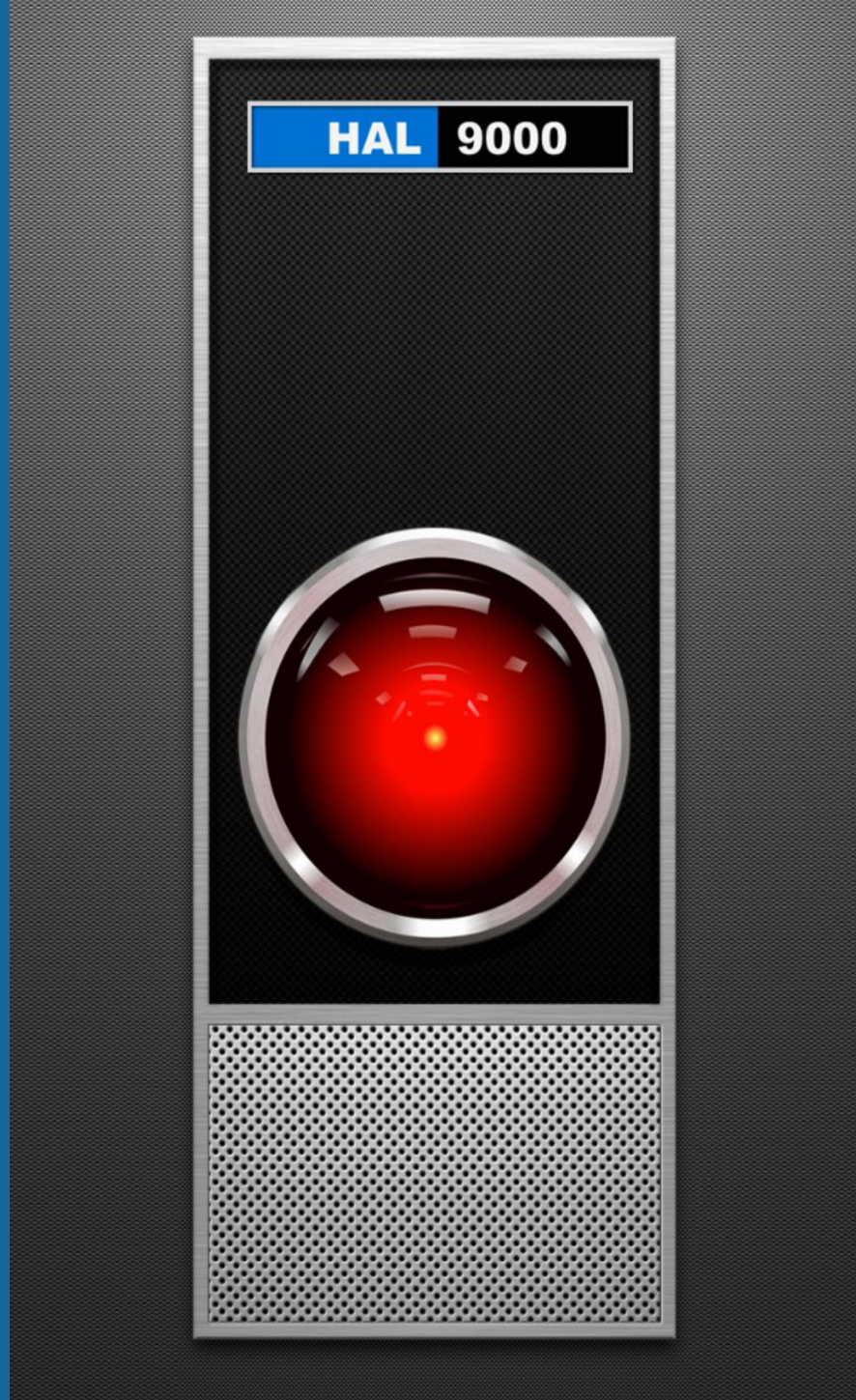

2017-Spring-ReadToMe
2017-Spring-ReadToMe
AppDelegate.h
AppDelegate.m
Main.storyboard
Launch Screen.storyboard
PrompterViewController.h
PrompterViewController.m
SpeechSynthesi...iewController.m
SpeechSynthesisViewController.h
PrerecordedViewController.h
PrerecordedViewController.m
Images.xcassets
Supporting Files
Frameworks
Products
2017-Spring-ReadToMe
2017-Spring-ReadToMe
Main.storyboard
Main.storyboard (Base)
Prerecorded View Con...
Speech Synthesis Vie...
Table View Controller...
Prompter View Control...
Navigation Controller...
Navigation Controller
Table View Controller
Pre-recorded
AVSpeechSynthesizer
Teleprompter
Table View
Static Content
Prerecorded View Controller
Speak To Me
Placeholder text
Prompter View Controller
Speak To Me
I am happy to join with you today in what
9:51
Talk to Me
Pre-recorded
AVSpeechSynthesizer
Teleprompter
iPhone X - 11.3

SPEECH SYNTHES



SPEECH SYNTHESIS

- AVSpeechSynthesizer
  - Introduced in iOS7
- Offline speech synthesis capability



SPEECH SYNTHESIS

AVSpeechSynthesizer

The `AVSpeechSynthesizer` class produces synthesized speech from text on an iOS device, and provides methods for controlling or monitoring the progress of ongoing speech.

Language

Swift | Objective-C

SDKs

iOS 7.0+

tvOS 7.0+

watchOS 2.0+

On This Page

Overview
Symbols
Relationships

Overview

To speak some amount of text, you must first create an `AVSpeechUtterance` instance containing the text. (Optionally, you may also use the utterance object to control parameters affecting its speech, such as voice, pitch, and rate.) Then, pass it to the `speak(_:)` method on a speech synthesizer instance to speak that utterance.

The speech synthesizer maintains a queue of utterances to be spoken. If the synthesizer is

SPEECH SYNTHESIS

- Support for 30 voices (specific country variations)
- Adjust rate of speed
- Adjust pitch
- Adjust volume
- Delegate protocol to allow interaction during speech

Class

AVSpeechSynthesizer

The `AVSpeechSynthesizer` class produces synthesized speech from text on an iOS device, and provides methods for controlling or monitoring the progress of ongoing speech.

Language

Swift | Objective-C

SDKs

iOS 7.0+

tvOS 7.0+

watchOS 2.0+

On This Page

Overview
Symbols
Relationships

Overview

To speak some amount of text, you must first create an `AVSpeechUtterance` instance containing the text. (Optionally, you may also use the utterance object to control parameters affecting its speech, such as voice, pitch, and rate.) Then, pass it to the `speak(_:)` method on a speech synthesizer instance to speak that utterance.

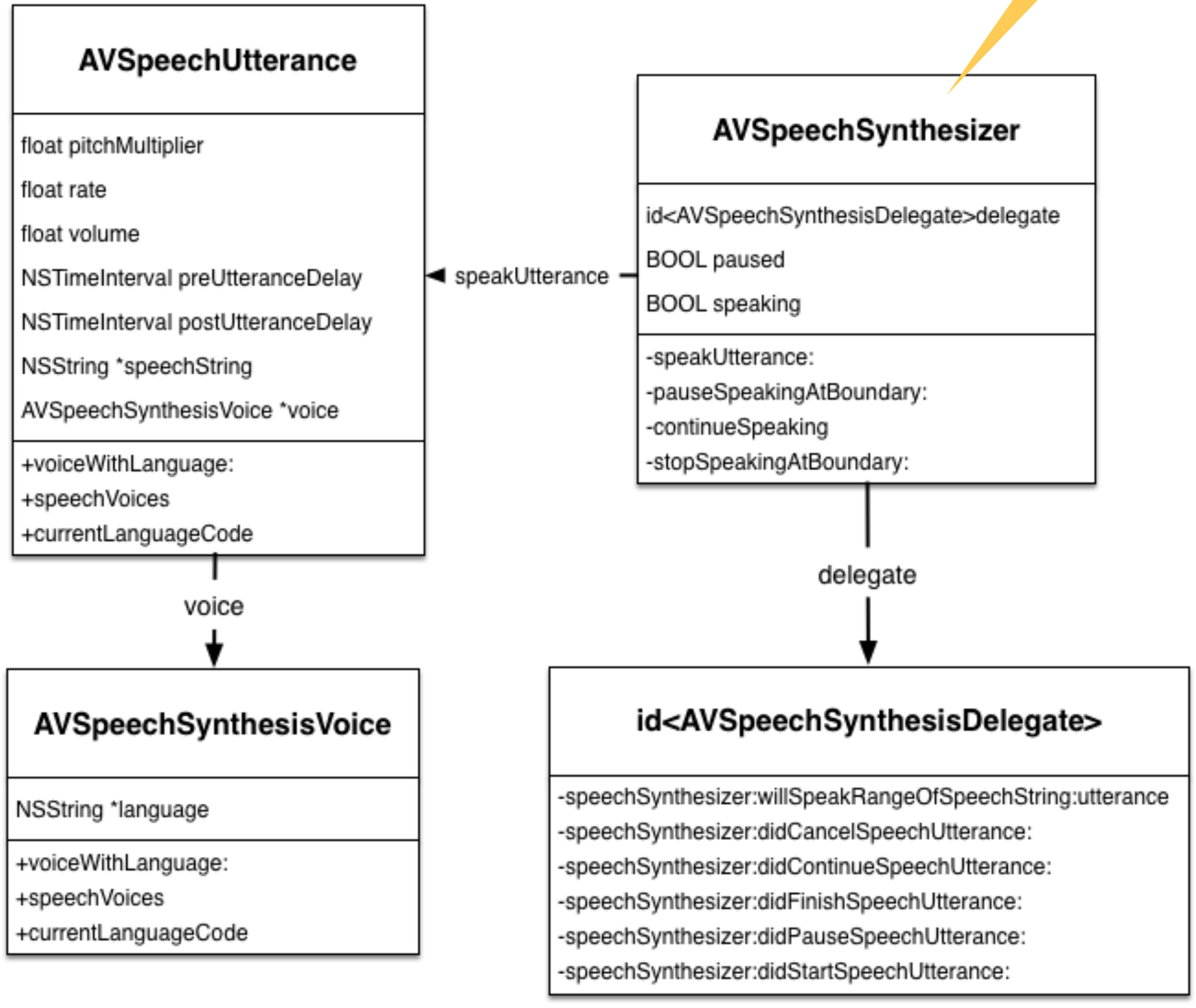
The speech synthesizer maintains a queue of utterances to be spoken. If the synthesizer is not currently speaking, calling `speak(_:)` begins speaking that utterance immediately (or begin waiting through its `preUtteranceDelay` if one is set). If the synthesizer is speaking, utterances are added to a queue and spoken in the order they are received.

After speech has begun, you can use the synthesizer object to pause or stop speech. After speech is paused, it may be continued from the point at which it left off; stopping ends speech entirely, removing any utterances yet to be spoken from the synthesizer's queue.

You may monitor the speech synthesizer by examining its speaking and paused properties, or by setting a delegate. Messages in the `AVSpeechSynthesizerDelegate` protocol are sent as significant events occur during speech synthesis.

Symbols

SPEECH SYNTHESIS



SPEECH SYNTHESIS

SUBTITLE

- Create instance of `AVSpeechSynthesis`
- Create `AVSpeechUtterance` for each section of text to be spoken

Wait until word is spoken before pausing

```
let synthesizer = AVSpeechSynthesizer()
synthesizer.stopSpeaking(at: .word)
let voice = AVSpeechSynthesisVoice(language: "en_US")
let utterance = AVSpeechUtterance(string: string)
utterance.voice = voice
synthesizer.speak(utterance)
```

SPEECH SYNTHESIS

```
let string = "One fish Two fish Red fish Blue fish. Black fish Blue fish Old fish New; fish.
    This one has a little star. This one has a little car. Say! What a lot of fish there are;"

let synthesizer = AVSpeechSynthesizer()
synthesizer.stopSpeaking(at: .word)
let voice = AVSpeechSynthesisVoice(language: "en_US")
let utterance = AVSpeechUtterance(string: string)
utterance.voice = voice
utterance.pitchMultiplier = 1.0
utterance.rate = 1.0
synthesizer.speak(utterance)
```

Variations to the spoken text

SPEECH SYNTHESIS

SUBTITLE

- AVSpeech Utterance properties
  - Properties are meant to better emulate and customize the spoken text
  - Example:
    - Different utterances can be varied for ‘emphasis’ in reading
    - Pause for dramatic effect

var pitchMultiplier: Float

The baseline pitch at which the utterance will be spoken.

var postUtteranceDelay: TimeInterval

The amount of time a speech synthesizer will wait after the utte
handling the next queued utterance.

var preUtteranceDelay: TimeInterval

The amount of time a speech synthesizer will wait before actual
utt

var ra

Th

var vo

Th

var vo

The volume used when speaking the utterance.

Sometimes useful to add punctuation, capitalize or misspell to get desired effect

SPEECH SYNTHESIS

SUBTITLE

- Available voices
  - List them using [AVSpeechSynthesisVoice speechVoices]

```
Arabic (Saudi Arabia) - ar-SA
Chinese (China) - zh-CN
Chinese (Hong Kong SAR China) - zh-HK
Chinese (Taiwan) - zh-TW
Czech (Czech Republic) - cs-CZ
Danish (Denmark) - da-DK
Dutch (Belgium) - nl-BE
Dutch (Netherlands) - nl-NL
English (Australia) - en-AU
English (Ireland) - en-IE
English (South Africa) - en-ZA
English (United Kingdom) - en-GB
English (United States) - en-US
Finnish (Finland) - fi-FI
French (Canada) - fr-CA
French (France) - fr-FR
German (Germany) - de-DE
Greek (Greece) - el-GR
Hindi (India) - hi-IN
Hungarian (Hungary) - hu-HU
Indonesian (Indonesia) - id-ID
Italian (Italy) - it-IT
Japanese (Japan) - ja-JP
Korean (South Korea) - ko-KR
Norwegian (Norway) - no-NO
Polish (Poland) - pl-PL
Portuguese (Brazil) - pt-BR
Portuguese (Portugal) - pt-PT
```

SPEECH SYNTHESIS

```
let string = "あのイーハトーヴォのすきとおった風、夏でも底に冷たさをもつ青いそら、うつくしい森で
    飾られたモーリオ市、郊外のぎらぎらひかる草の波。またそのなかでいっしょになったたくさんのひとた
    ち、ファゼーロとロザーロ、羊飼のミーロや、顔の赤いこどもたち、地主のテーモ、山猫博士のボーガン
    ト・テストゥパーゴなど、いまこの暗い巨きな石の建物のなかで考えていると、みんなむかし風のなつか
    しい青い幻燈のように思われます。"

let synthesizer = AVSpeechSynthesizer()
let voice = AVSpeechSynthesisVoice(language: "ja_JP")
let utterance = AVSpeechUtterance(string: string)
utterance.voice = voice
synthesizer.speak(utterance)
```

SPEECH SYNTHESIS

- Does not translate into languages
  - You need to provide the correctly written text in a language
  - Sometimes different languages sound better
    - Australian English

```
Arabic (Saudi Arabia) - ar-SA
Chinese (China) - zh-CN
Chinese (Hong Kong SAR China) - zh-HK
Chinese (Taiwan) - zh-TW
Czech (Czech Republic) - cs-CZ
Danish (Denmark) - da-DK
Dutch (Belgium) - nl-BE
Dutch (Netherlands) - nl-NL
English (Australia) - en-AU
English (Ireland) - en-IE
English (South Africa) - en-ZA
English (United Kingdom) - en-GB
English (United States) - en-US
Finnish (Finland) - fi-FI
French (Canada) - fr-CA
French (France) - fr-FR
German (Germany) - de-DE
Greek (Greece) - el-GR
Hindi (India) - hi-IN
Hungarian (Hungary) - hu-HU
Indonesian (Indonesia) - id-ID
Italian (Italy) - it-IT
Japanese (Japan) - ja-JP
Korean (South Korea) - ko-KR
Norwegian (Norway) - no-NO
Polish (Poland) - pl-PL
Portuguese (Brazil) - pt-BR
Portuguese (Portugal) - pt-PT
Romanian (Romania) - ro-RO
Russian (Russia) - ru-RU
Slovak (Slovakia) - sk-SK
Spanish (Mexico) - es-MX
Spanish (Spain) - es-ES
Swedish (Sweden) - sv-SE
Thai (Thailand) - th-TH
Turkish (Turkey) - tr-TR
```

SPEECH SYNTHESIS

- AVSpeechSynthesisDelegate Protocol
  - Defines methods that the delegate of an AVSpeechSynthesizer object may implement
  - All methods in this protocol are optional
  - Respond to events that occur during speech synthesis

Responding to Speech Synthesis Events

Useful for kids with short attention spans

`func speechSynthesizer(AVSpeechSynthesizer, didCancel: AVSpeechUtterance)`

Tells the delegate when the synthesizer has canceled speaking an utterance.

`func speechSynthesizer(AVSpeechSynthesizer, didContinue: AVSpeechUtterance)`

Tells the delegate when the synthesizer has resumed speaking an utterance after being paused.

`func speechSynthesizer(AVSpeechSynthesizer, didFinish: AVSpeechUtterance)`

Tells the delegate when the synthesizer has finished speaking an utterance.

`func speechSynthesizer(AVSpeechSynthesizer, didPause: AVSpeechUtterance)`

Tells the delegate when the synthesizer has paused while speaking an utterance.

`func speechSynthesizer(AVSpeechSynthesizer, didStart: AVSpeechUtterance)`

Tells the delegate when the synthesizer has begun speaking an utterance.

`func speechSynthesizer(AVSpeechSynthesizer, willSpeakRangeOfSpeechString: NSRange, utterance: AVSpeechUtterance)`

Tells the delegate when the synthesizer is about to speak a portion of an utterance's text.

Identify what is being spoken

SPEECH SYNTHESIS

- Identify start of speaking
- Identify range of spoken characters

```
let string = "One fish Two fish Red fish Blue fish. Black fish Blue fish Old fish New; fish.
    This one has a little star. This one has a little car. Say! What a lot of fish there are."

class PlaygroundViewController: UIViewController {
  let synthesizer = AVSpeechSynthesizer()

  override func viewDidLoad() {

    super.viewDidLoad()

    synthesizer.delegate = self
    let voice = AVSpeechSynthesisVoice(language: "en_US")
    let utterance = AVSpeechUtterance(string: string)
    utterance.voice = voice
    utterance.pitchMultiplier = 1.0
    utterance.rate = 0.25
    synthesizer.speak(utterance)
  }
}


extension PlaygroundViewController: AVSpeechSynthesizerDelegate {

  func speechSynthesizer(_ synthesizer: AVSpeechSynthesizer,
                         didStart utterance: AVSpeechUtterance) {
    print("Started")
  }

  func speechSynthesizer(_ synthesizer: AVSpeechSynthesizer,
                         willSpeakRangeOfSpeechString characterRange: NSRange,
                         utterance: AVSpeechUtterance) {
    print("📣: \(characterRange.location) - \(characterRange.length+characterRange.location)")
  }
}

// Create an instance of the view controller
let viewController = PlaygroundViewController()
```

SPEECH SYNTHESIS

- Can get errors when running on simulator (not on device)
- Does not effect anything
- Open ra

```
AVAudioSessionUtilities.h:88: GetProperty_DefaultToZero: AudioSessionGetProperty
('disa') failed with error: '?ytp'
2014-04-07 10:29:20.932 2014-Spring-ReadToMe[98703:3f03] 10:29:20.932 ERROR:
AVAudioSessionUtilities.h:88: GetProperty_DefaultToZero: AudioSessionGetProperty
('disa') failed with error: '?ytp'
```

?

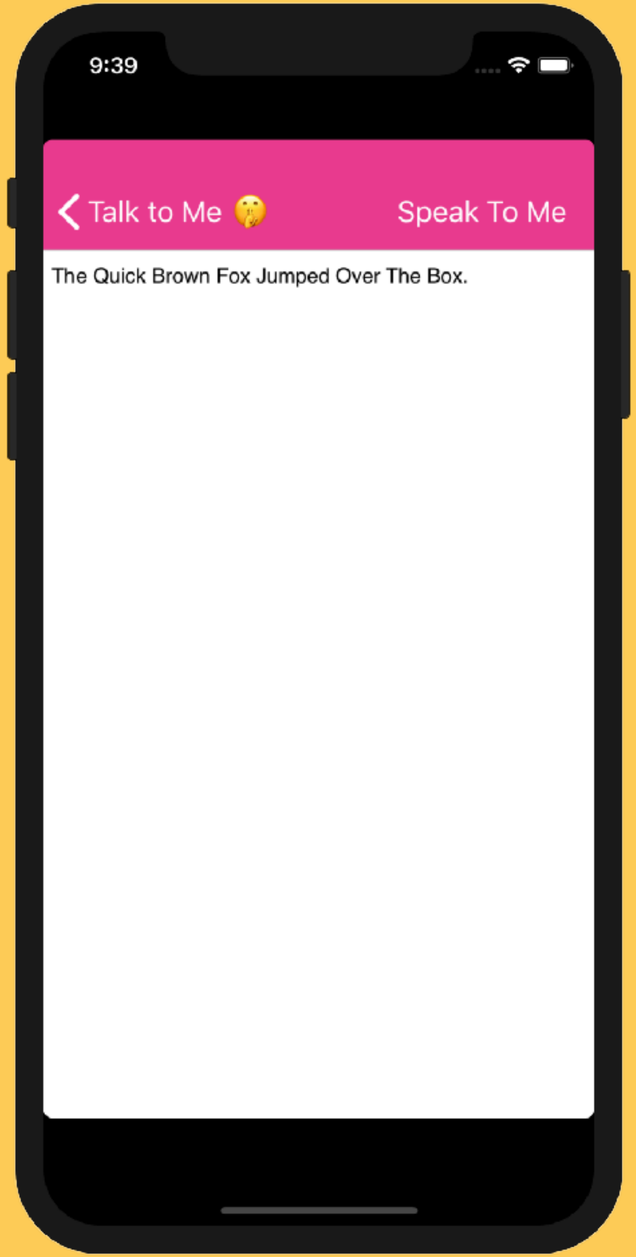
SPEECH SYNTHESIS

```
-(NSString*)detectLanguage:(NSString*)string
{
    NSArray *tagschemes = [NSArray arrayWithObjects:NSLinguisticTagSchemeLanguage, nil];
    NSLinguisticTagger *tagger = [[NSLinguisticTagger alloc] initWithTagSchemes:tagschemes options:0];
    [tagger setString:string];
    NSString *language = [tagger tagAtIndex:0 scheme:NSLinguisticTagSchemeLanguage
                          tokenRange:NULL sentenceRange:NULL];
    return language;

}
```

- Detect text language from text

PRE-RECORDED AUDIO



AUDACITY

Audacity®

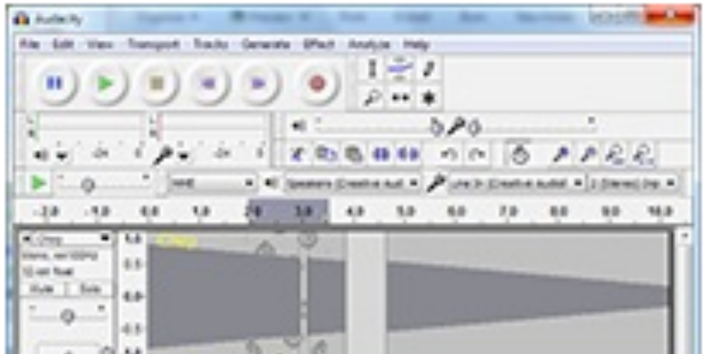
Google™ Custom Search Search

Search Audacity sites and other selected sites.

Home | About | Download | Help | Contact Us | Get Involved | Donate

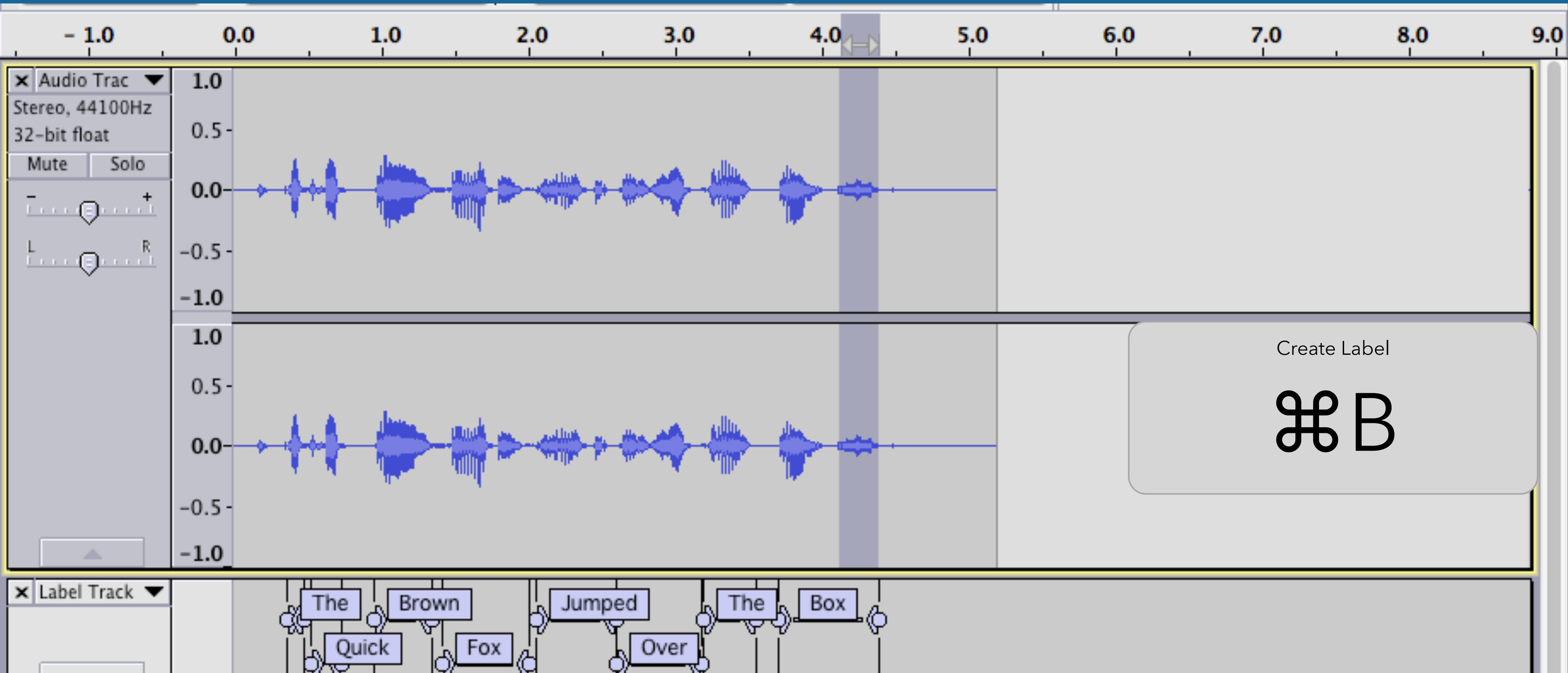
Audacity® is free, open source, cross-platform software for recording and editing sounds.

Audacity is available for Windows®, Mac®, GNU/Linux® and other operating systems. Check our feature list, Wiki and Forum for more information.

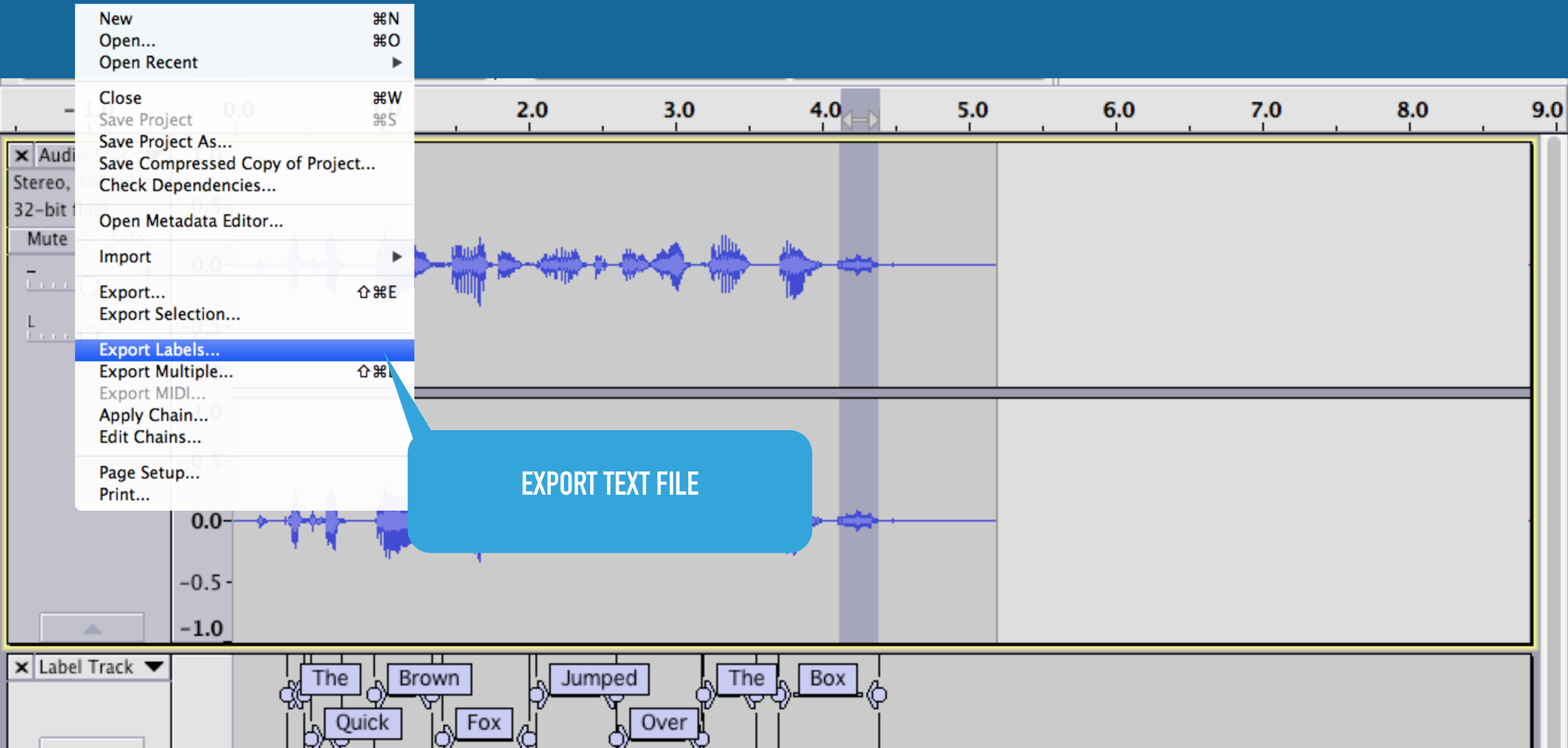


Download Audacity 2.0.3

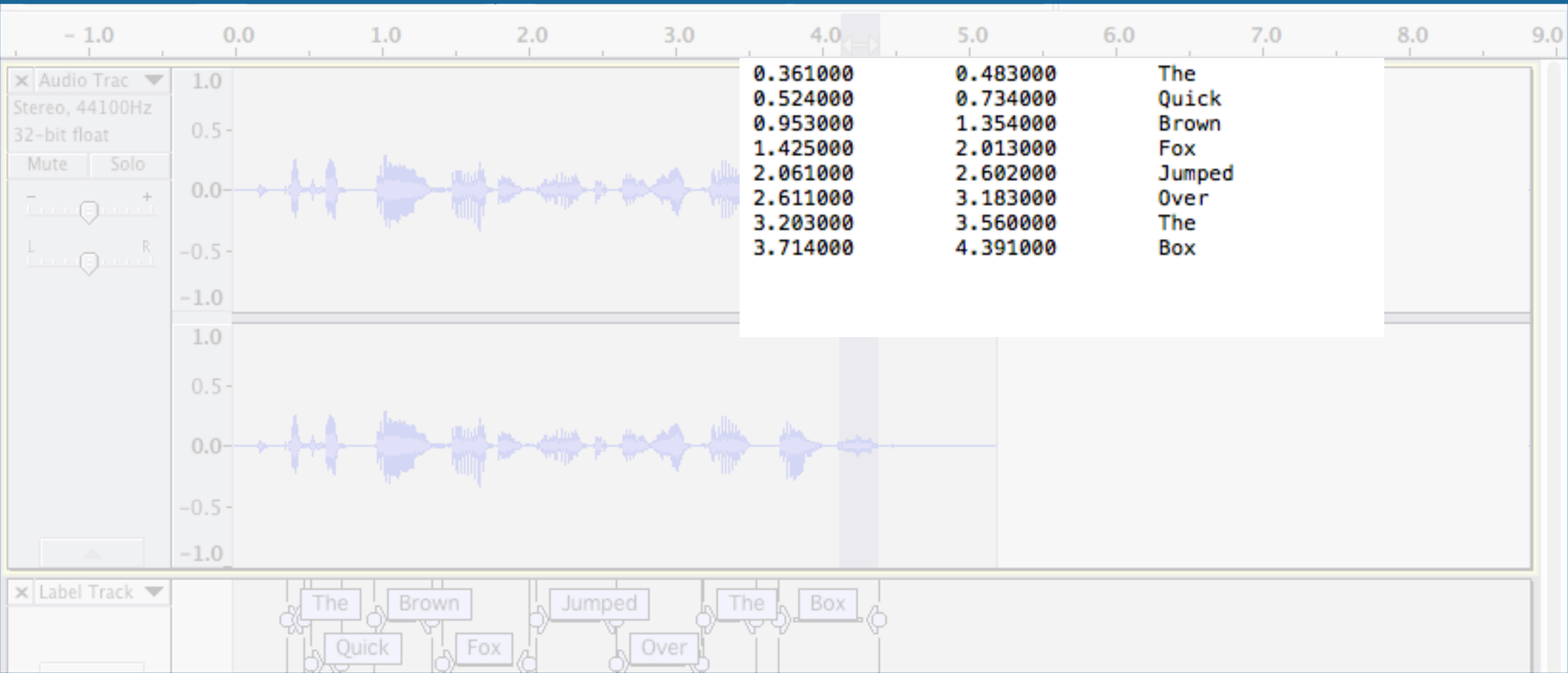
AUDACITY


- 1.0
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
Audio Trac
Stereo, 44100Hz
32-bit float
Mute
Solo
L
R
1.0
0.5
0.0
-0.5
-1.0
Create Label
⌘B
Label Track
The
Brown
Jumped
The
Box
Quick
Fox
Over

AUDACITY


New ⌘N
Open... ⌘O
Open Recent
Close ⌘W
Save Project ⌘S
Save Project As...
Save Compressed Copy of Project...
Check Dependencies...
Open Metadata Editor...
Import
Export... ⇧⌘E
Export Selection...
Export Labels...
Export Multiple... ⇧⌘
Export MIDI...
Apply Chain...
Edit Chains...
Page Setup...
Print...
EXPORT TEXT FILE
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
Stereo,
32-bit
Mute
0.0
-0.5
-1.0
Label Track
The
Brown
Jumped
The
Box
Quick
Fox
Over

AUDACITY



0.361000	0.483000	The
0.524000	0.734000	Quick
0.953000	1.354000	Brown
1.425000	2.013000	Fox
2.061000	2.602000	Jumped
2.611000	3.183000	Over
3.203000	3.560000	The
3.714000	4.391000	Box

APPROACHES

- Strategy #1
  - Get start and stop time for each word.
  - Monitor that with a timer and update based on the time
  - Gets out of sync
- Strategy #2
  - Use AVAudioPlayer timer to update the strings
  - Gets out of sync (especially on short words)

- Strategy #3
  - Set a timer for each word
  - Clear all attributes and update based on the timer

APPROACHES

- NSTimer for each word

```
var startIndex = 0
for i in 0..<timer.count{
    let word = timer[i]["word"] as! String
    let time = timer[i]["start"] as! Double
    let length = word.count
    let wordTimer = Timer.scheduledTimer(timeInterval: time, target: self, selector:
        #selector(updateAttributedString(_:)), userInfo: [startIndex, length], repeats: false)
    timerList.append(wordTimer)
    startIndex += length + 1
}
```

- AVFoundation to play audio

```
let audioFileURL = URL(fileURLWithPath: Bundle.main.path(forResource: name, ofType: "mp3")!)

do {
    try audioFile = AVAudioPlayer(contentsOf: audioFileURL as URL, fileTypeHint: nil)
    audioFile.prepareToPlay()
} catch let err as NSError {
    print(err.debugDescription)
}
audioFile.play()
```

APPROACHES

```
let audioFileURL = URL(fileURLWithPath: Bundle.main.path(forResource: name, ofType: "mp3")!)

do {
    try audioFile = AVAudioPlayer(contentsOf: audioFileURL as URL, fileTypeHint: nil)
    audioFile.prepareToPlay()
} catch let err as NSError {
    print(err.debugDescription)
}
audioFile.play()
```

Create the audio player

APPROACHES

Create an individual timer for each word.

```
var startIndex = 0
for i in 0..<timer.count{
    let word = timer[i]["word"] as! String
    let time = timer[i]["start"] as! Double
    let length = word.count
    let wordTimer = Timer.scheduledTimer(timeInterval: time, target: self, selector:
        #selector(updateAttributedString(_:)), userInfo: [startIndex, length], repeats: false)
    timerList.append(wordTimer)
    startIndex += length + 1
}
```

APPROACHES

```
@objc func updateAttributedString(_ timer: Timer) {
    let range = timer.userInfo as! [Int]

    attributedString.addAttribute(NSBackgroundColorAttributeName,
                                  value: UIColor.clear,
                                  range: NSRange(location: 0, length: attributedString.length))

    attributedString.addAttribute(NSForegroundColorAttributeName,
                                  value: blue,
                                  range: NSRange(location: 0, length: attributedString.length))

    attributedString.addAttribute(NSBackgroundColorAttributeName,
                                  value: blue,
                                  range: NSRange(location: range[0], length: range[1]))

    attributedString.addAttribute(NSForegroundColorAttributeName,
                                  value: UIColor.white,
                                  range: NSRange(location: range[0], length: range[1]))

    textLabel.attributedText = attributedString
}
```

When the timer fires, update the string

Clear the color, update the color

THE UNIVERSITY OF CHICAGO

ADVANCED iOS APPLICATION DEVELOPMENT

MPCS 51032 • SPRING 2020 • SESSION 1E