#WWDC19

Font Management and Text Scaling

Julio González, Type Engineering Manager
Eric Dudiak, UIKit Engineer
Donna Tom, TextKit Engineer

Agenda

New system fonts

Installing and accessing user fonts

Font selection

Text scaling

Agenda

New system fonts

Installing and accessing user fonts

Font selection

Text scaling

Agenda

New system fonts

Installing and accessing user fonts

Font selection

Text scaling

Agenda

New system fonts

Installing and accessing user fonts

Font selection

Text scaling

Agenda

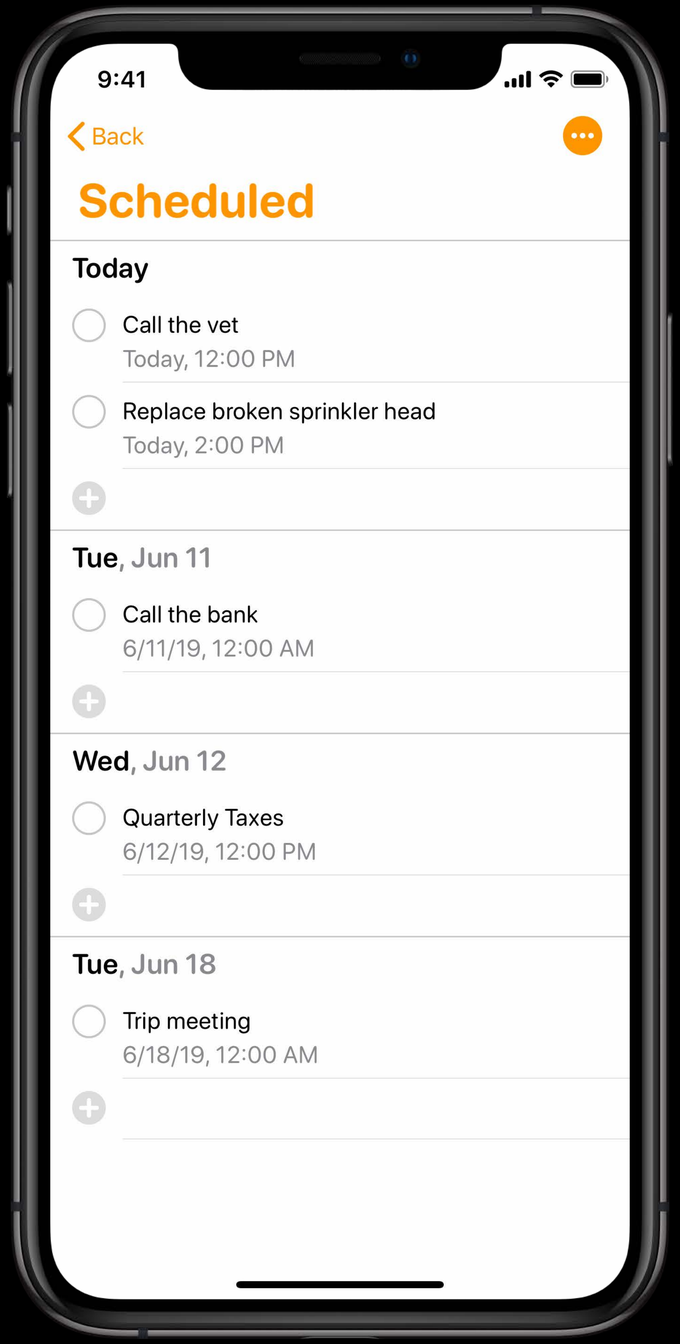
New system fonts

Installing and accessing user fonts

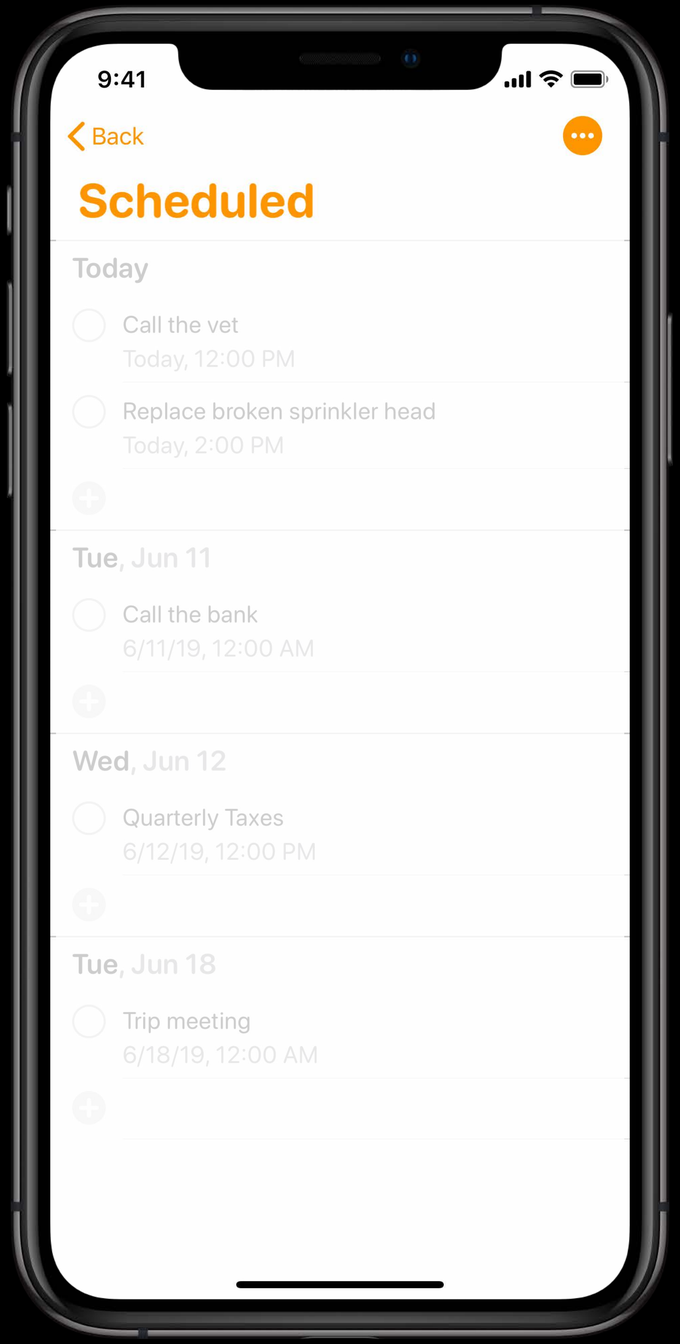
Font selection

Text scaling

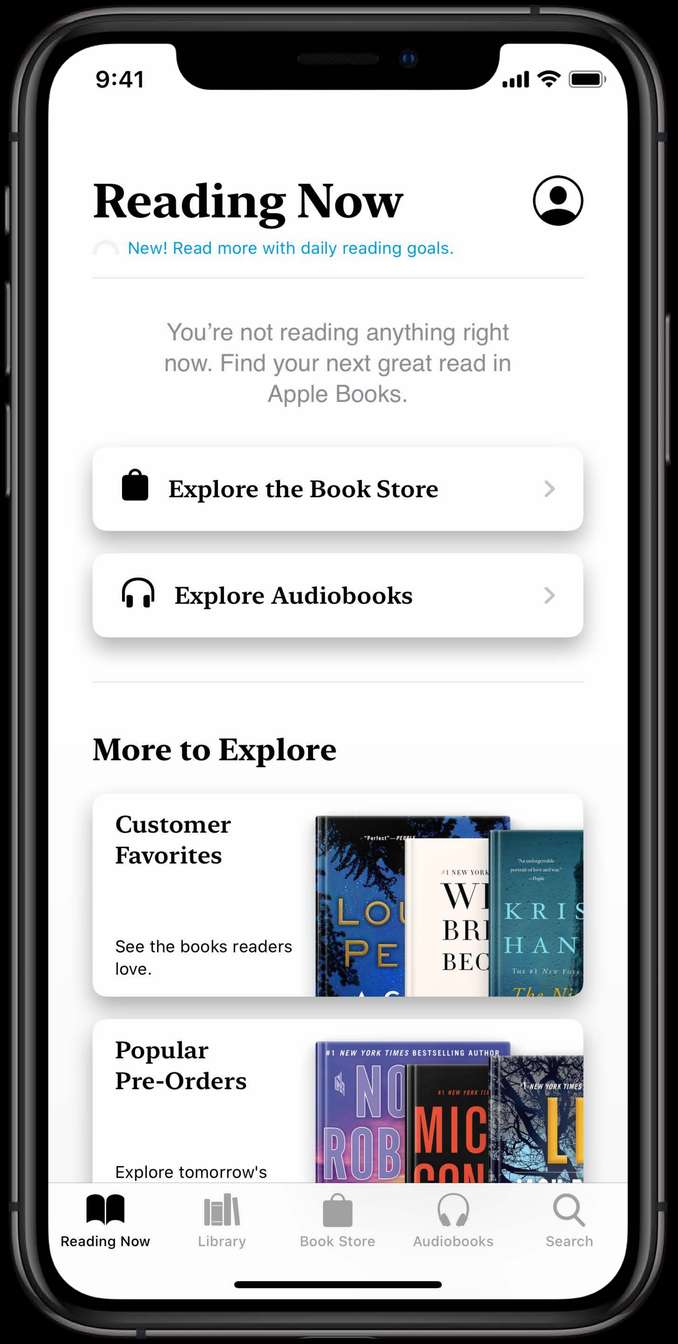
Rounded


9:41
Back
Scheduled
Today
Call the vet
Today, 12:00 PM
Replace broken sprinkler head
Today, 2:00 PM
Tue, Jun 11
Call the bank
6/11/19, 12:00 AM
Wed, Jun 12
Quarterly Taxes
6/12/19, 12:00 PM
Tue, Jun 18
Trip meeting
6/18/19, 12:00 AM

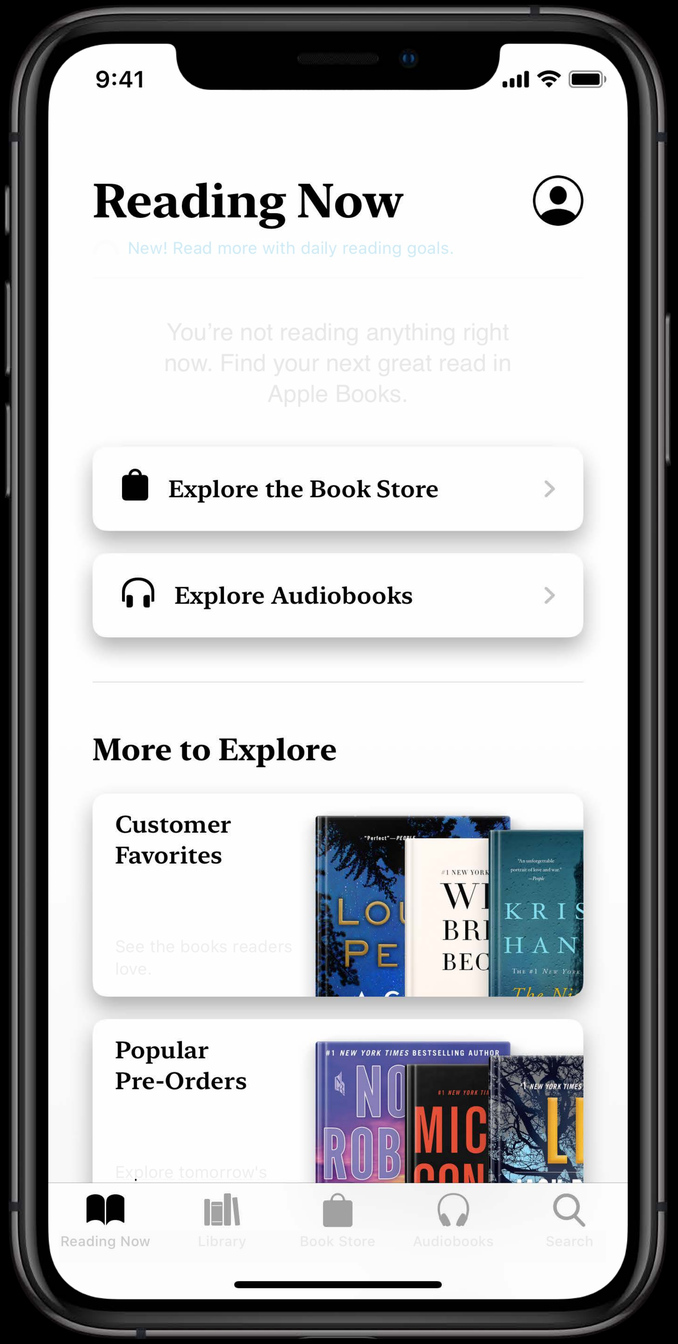
Rounded


9:41
Back
Scheduled
Today
Call the vet
Today, 12:00 PM
Replace broken sprinkler head
Today, 2:00 PM
Tue, Jun 11
Call the bank
6/11/19, 12:00 AM
Wed, Jun 12
Quarterly Taxes
6/12/19, 12:00 PM
Tue, Jun 18
Trip meeting
6/18/19, 12:00 AM

Serif


9:41
Reading Now
New! Read more with daily reading goals.
You're not reading anything right now. Find your next great read in Apple Books.
Explore the Book Store
Explore Audiobooks
More to Explore
Customer Favorites
See the books readers love.
Popular Pre-Orders
Explore tomorrow's
Reading Now
Library
Book Store
Audiobooks
Search

Serif


9:41
Reading Now
New! Read more with daily reading goals.
You're not reading anything right now. Find your next great read in Apple Books.
Explore the Book Store
Explore Audiobooks
More to Explore
Customer Favorites
See the books readers love.
Popular Pre-Orders
Explore tomorrow's
Reading Now
Library
Book Store
Audiobooks
Search

Mono


To create a new variable, use **var** followed by a name for the variable and a value.
New Contact
C
var name = "Chris"
var age = 19
The **assignment operator** (the equals sign) sets the value of the variable.
name stores a **String** value (text in quote marks).
age stores an **Int** value (an integer, or whole number).
4 of 7

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?
```

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}
```

NEW

```
open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?

let descriptor = UIFont.systemFont(ofSize: 17.0, weight: .bold).fontDescriptor
if let roundedDescriptor = descriptor.withDesign(.rounded) {
    let roundedBoldFont = UIFont(descriptor: roundedDescriptor, size: 0.0)
    …
}
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?

let descriptor = UIFont.systemFont(ofSize: 17.0, weight: .bold).fontDescriptor
if let roundedDescriptor = descriptor.withDesign(.rounded) {
    let roundedBoldFont = UIFont(descriptor: roundedDescriptor, size: 0.0)
    …
}
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?

let descriptor = UIFont.systemFont(ofSize: 17.0, weight: .bold).fontDescriptor
if let roundedDescriptor = descriptor.withDesign(.rounded) {
    let roundedBoldFont = UIFont(descriptor: roundedDescriptor, size: 0.0)
    …
}
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?

let descriptor = UIFont.systemFont(ofSize: 17.0, weight: .bold).fontDescriptor
if let roundedDescriptor = descriptor.withDesign(.rounded) {
    let roundedBoldFont = UIFont(descriptor: roundedDescriptor, size: 0.0)
    …
}
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?

let descriptor = UIFont.systemFont(ofSize: 17.0, weight: .bold).fontDescriptor
if let roundedDescriptor = descriptor.withDesign(.rounded) {
    let roundedBoldFont = UIFont(descriptor: roundedDescriptor, size: 0.0)
    …
}
```

NEW

```
extension UIFontDescriptor.SystemDesign {
    public static let `default`: UIFontDescriptor.SystemDesign

    public static let rounded: UIFontDescriptor.SystemDesign

    public static let serif: UIFontDescriptor.SystemDesign

    public static let monospaced: UIFontDescriptor.SystemDesign
}

open func withDesign(_ design: UIFontDescriptor.SystemDesign) -> UIFontDescriptor?

let descriptor = UIFont.systemFont(ofSize: 17.0, weight: .bold).fontDescriptor
if let roundedDescriptor = descriptor.withDesign(.rounded) {
    let roundedBoldFont = UIFont(descriptor: roundedDescriptor, size: 0.0)
    …
}
```

Font Instantiation

By name

Do not instantiate system fonts by name

- Never use font names with a dot prefix

Font instantiation by name is not guaranteed

- Fonts may be replaced
- User installed font may go away

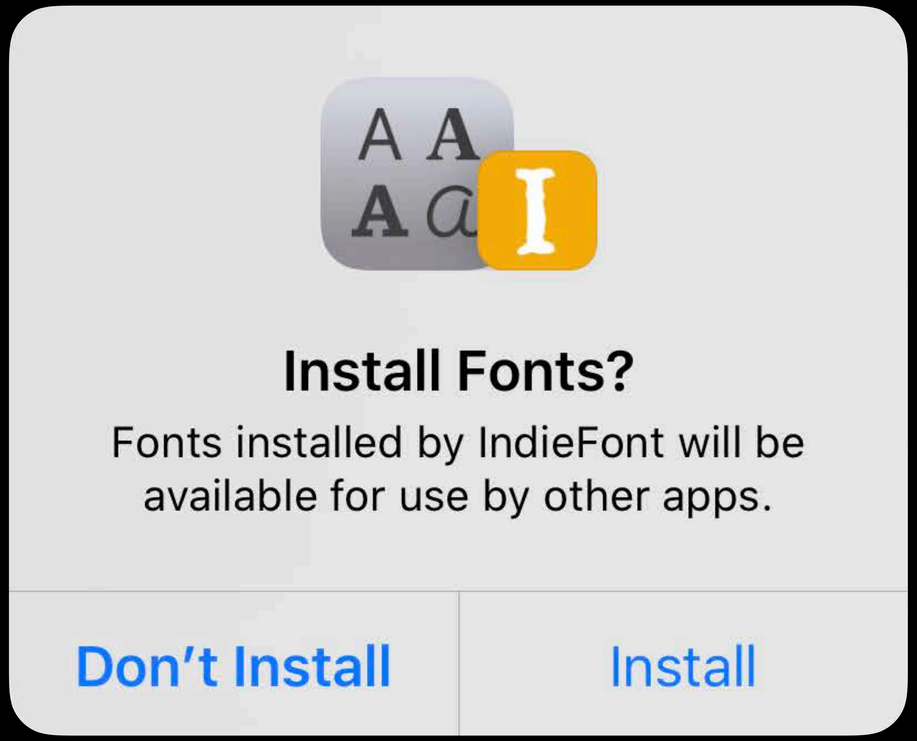
Installing and Accessing User Fonts

Installing Fonts

NEW

Font Provider applications

- Ability to register fonts system wide
- Settings app to manage fonts



Installing Fonts

Font Provider application requirements

NEW

Entitlements

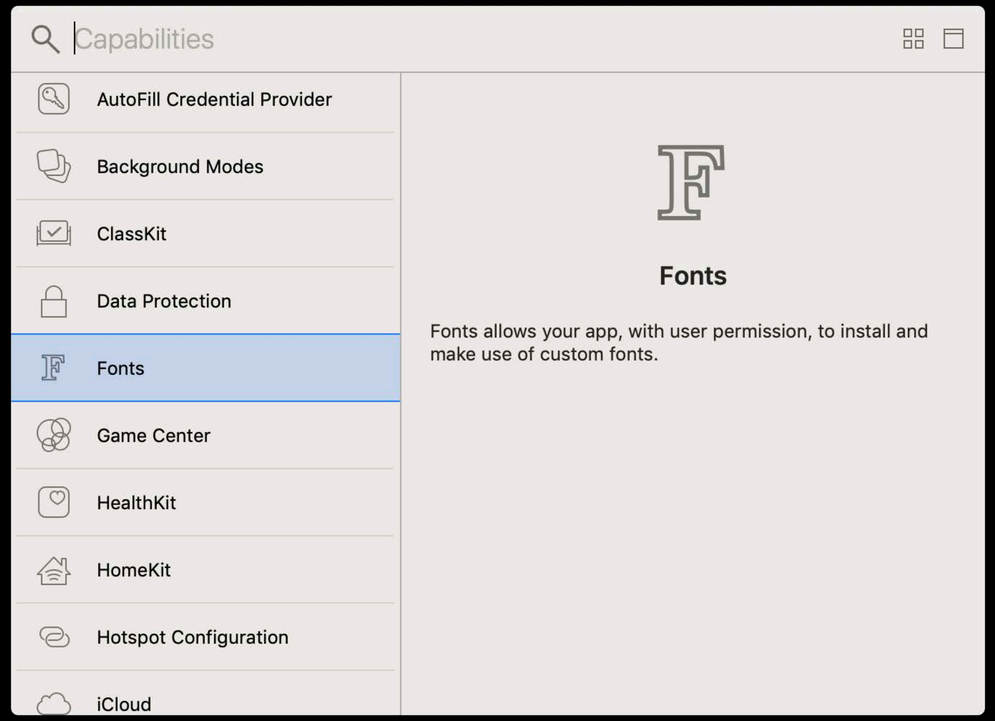
App Store review

- Fonts are bundled or On-Demand Resources
- Valid file formats: ttf, otf, ttc

Installing Fonts

NEW

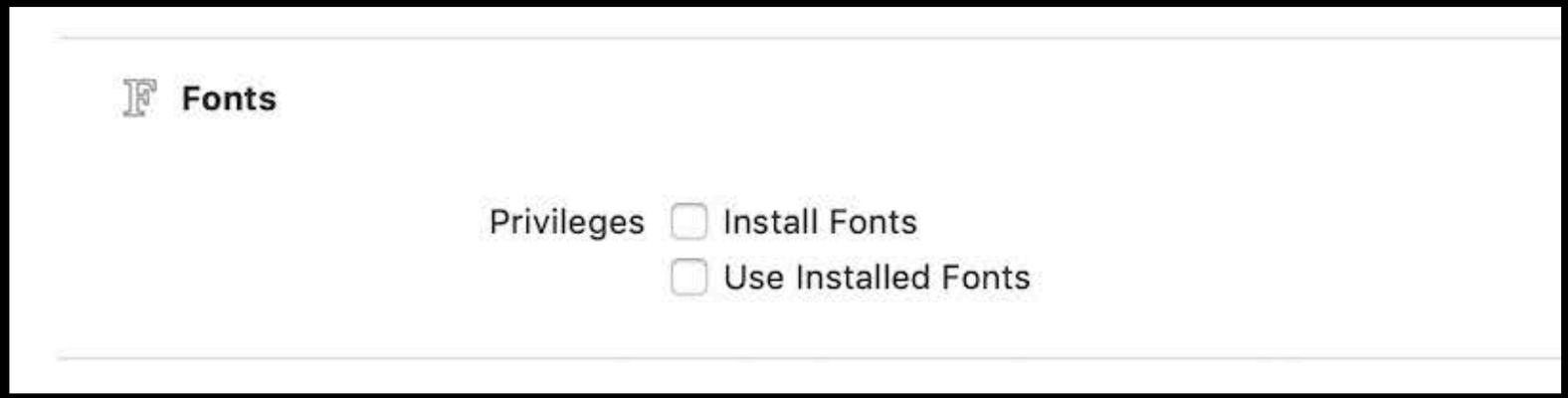
Entitlements



Installing Fonts

NEW

Entitlements



Installing Fonts

Font Provider application expectations

UI to browse application’s font library

Respond to font change notifications

Use On-Demand Resources for large libraries

- Asset Catalogs for very large libraries

Demo

Registration and Access APIs

CoreText/CTFontManager.h

```
CTFontManagerRegisterFontURLs
CTFontManagerRegisterFontDescriptors
CTFontManagerRegisterFontsWithAssetNames
CTFontManagerCopyRegisteredDescriptors
CTFontManagerRequestFonts
CTFontManagerUnregisterFontURLs
CTFontManagerUnegisterFontDescriptors
```

Registration and Access APIs

CoreText/CTFontManager.h

```
CTFontManagerRegisterFontURLs
CTFontManagerRegisterFontDescriptors
CTFontManagerRegisterFontsWithAssetNames
CTFontManagerCopyRegisteredDescriptors
CTFontManagerRequestFonts
CTFontManagerUnregisterFontURLs
CTFontManagerUnegisterFontDescriptors
```

Registration and Access APIs

CoreText/CTFontManager.h

```
CTFontManagerRegisterFontURLs                          CTFontManagerUnregisterFontURLs

CTFontManagerRegisterFontDescriptors                   CTFontManagerUnegisterFontDescriptors

CTFontManagerRegisterFontsWithAssetNames

CTFontManagerCopyRegisteredDescriptors

CTFontManagerRequestFonts
```

Registration and Access APIs

CoreText/CTFontManager.h

```
CTFontManagerRegisterFontURLs
CTFontManagerRegisterFontDescriptors
CTFontManagerRegisterFontsWithAssetNames
CTFontManagerCopyRegisteredDescriptors
CTFontManagerRequestFonts
CTFontManagerUnregisterFontURLs
CTFontManagerUnegisterFontDescriptors
```

Registration and Access APIs

CoreText/CTFontManager.h

```
CTFontManagerRegisterFontURLs                    CTFontManagerUnregisterFontURLs

CTFontManagerRegisterFontDescriptors             CTFontManagerUnegisterFontDescriptors

CTFontManagerRegisterFontsWithAssetNames

CTFontManagerCopyRegisteredDescriptors

CTFontManagerRequestFonts
```

Registration and Access APIs

CoreText/CTFontManager.h

```
CTFontManagerRegisterFontURLs
CTFontManagerRegisterFontDescriptors
CTFontManagerRegisterFontsWithAssetNames
CTFontManagerCopyRegisteredDescriptors
CTFontManagerRequestFonts
CTFontManagerUnregisterFontURLs
CTFontManagerUnegisterFontDescriptors
```

Registration and Access APIs

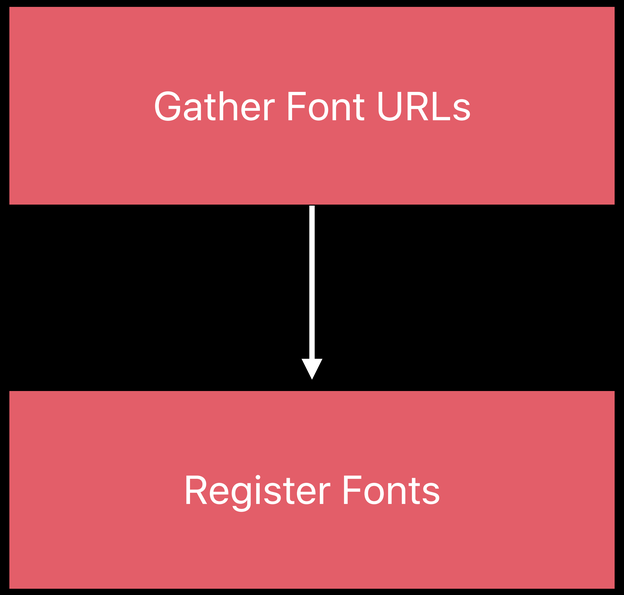
CoreText/CTFontManager.h

```
CTFontManagerRegisterFontURLs                    CTFontManagerUnregisterFontURLs

CTFontManagerRegisterFontDescriptors             CTFontManagerUnegisterFontDescriptors

CTFontManagerRegisterFontsWithAssetNames

CTFontManagerCopyRegisteredDescriptors

CTFontManagerRequestFonts
```

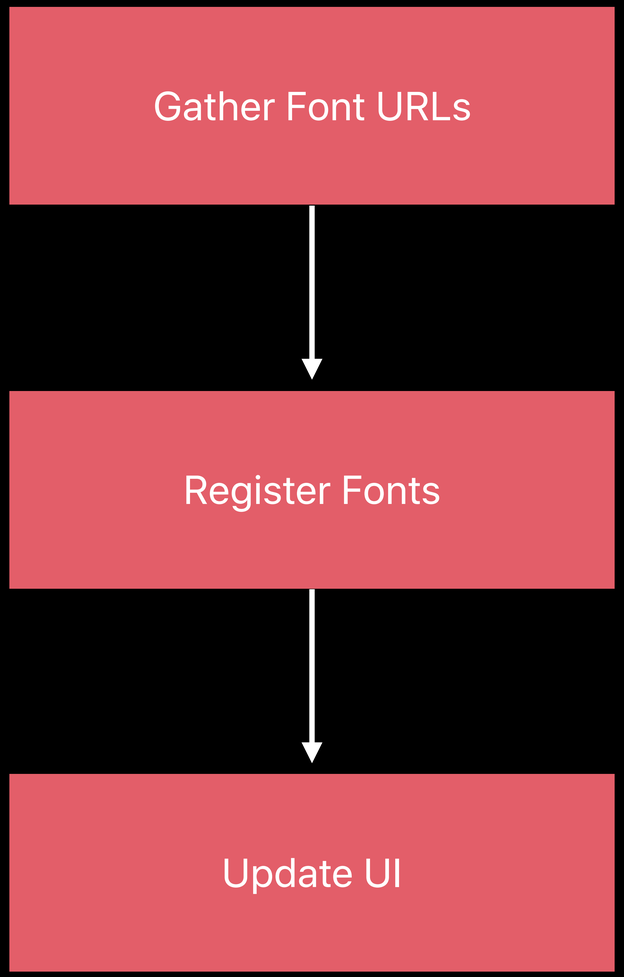
Font Provider — Registration

Font Provider — Registration

Font Provider — Registration



Font Provider — Registration



```
// FontProvider App – Registering for font system notifications

override func viewDidLoad() {
    NotificationCenter.default.addObserver(
            self,
            selector: #selector(fontsChangedNotification(_:)),
            name: kCTFontManagerRegisteredFontsChangedNotification as NSNotification.Name,
            object: nil)
}

func fontsChangedNotification(_ sender: Any) {
    FontProvider.updateRegisteredFonts()
    DispatchQueue.main.async {
        self.detailViewController?.configureView()
        self.tableView.reloadData()
    }
}
```

```
// FontProvider App – Registering for font system notifications

override func viewDidLoad() {
    NotificationCenter.default.addObserver(
            self,
            selector: #selector(fontsChangedNotification(_:)),
            name: kCTFontManagerRegisteredFontsChangedNotification as NSNotification.Name,
            object: nil)
}

func fontsChangedNotification(_ sender: Any) {
    FontProvider.updateRegisteredFonts()
    DispatchQueue.main.async {
        self.detailViewController?.configureView()
        self.tableView.reloadData()
    }
}
```

```
// FontProvider App – Registering for font system notifications

override func viewDidLoad() {
    NotificationCenter.default.addObserver(
            self,
            selector: #selector(fontsChangedNotification(_:)),
            name: kCTFontManagerRegisteredFontsChangedNotification as NSNotification.Name,
            object: nil)
}

func fontsChangedNotification(_ sender: Any) {
    FontProvider.updateRegisteredFonts()
    DispatchQueue.main.async {
        self.detailViewController?.configureView()
        self.tableView.reloadData()
    }
}
```

```
// FontProvider App – Registering for font system notifications

override func viewDidLoad() {
    NotificationCenter.default.addObserver(
            self,
            selector: #selector(fontsChangedNotification(_:)),
            name: kCTFontManagerRegisteredFontsChangedNotification as NSNotification.Name,
            object: nil)
}

func fontsChangedNotification(_ sender: Any) {
    FontProvider.updateRegisteredFonts()
    DispatchQueue.main.async {
        self.detailViewController?.configureView()
        self.tableView.reloadData()
    }
}
```

```swift
// FontProvider App – Registering for font system notifications

override func viewDidLoad() {
    NotificationCenter.default.addObserver(
            self,
            selector: #selector(fontsChangedNotification(_:)),
            name: kCTFontManagerRegisteredFontsChangedNotification as NSNotification.Name,
            object: nil)
}

func fontsChangedNotification(_ sender: Any) {
    FontProvider.updateRegisteredFonts()
    DispatchQueue.main.async {
        self.detailViewController?.configureView()
        self.tableView.reloadData()
    }
}
```

```
// FontProvider App – Registering for font system notifications

override func viewDidLoad() {
    NotificationCenter.default.addObserver(
            self,
            selector: #selector(fontsChangedNotification(_:)),
            name: kCTFontManagerRegisteredFontsChangedNotification as NSNotification.Name,
            object: nil)
}

func fontsChangedNotification(_ sender: Any) {
    FontProvider.updateRegisteredFonts()
    DispatchQueue.main.async {
        self.detailViewController?.configureView()
        self.tableView.reloadData()
    }
}
```

```swift
// FontProvider App — Registering for font system notifications

override func viewDidLoad() {
    NotificationCenter.default.addObserver(
            self,
            selector: #selector(fontsChangedNotification(_:)),
            name: kCTFontManagerRegisteredFontsChangedNotification as NSNotification.Name,
            object: nil)
}

func fontsChangedNotification(_ sender: Any) {
    FontProvider.updateRegisteredFonts()
    DispatchQueue.main.async {
        self.detailViewController?.configureView()
        self.tableView.reloadData()
    }
}
```

```
// FontProvider App — Registering All Fonts

static func registerFamilies(_ names: [FontFamily], _ registrationHandler:
    (([Error], Bool) -> Bool)?) {
    let fontURLs = names.flatMap({ $0.urls })
    CTFontManagerRegisterFontURLs(fontURLs as CFArray, .persistent, true) {
        (errors: CFArray, done: Bool) -> Bool in
        return processHandler(errors: errors, done: done, registrationHandler)
    }
}
```

```
// FontProvider App — Registering All Fonts

static func registerFamilies(_ names: [FontFamily], _ registrationHandler:
    (([Error], Bool) -> Bool)?) {
    let fontURLs = names.flatMap({ $0.urls })
    CTFontManagerRegisterFontURLs(fontURLs as CFArray, .persistent, true) {
        (errors: CFArray, done: Bool) -> Bool in
        return processHandler(errors: errors, done: done, registrationHandler)
    }
}
```

```
// FontProvider App — Registering All Fonts

static func registerFamilies(_ names: [FontFamily], _ registrationHandler:
    (([Error], Bool) -> Bool)?) {
    let fontURLs = names.flatMap({ $0.urls })
    CTFontManagerRegisterFontURLs(fontURLs as CFArray, .persistent, true) {
        (errors: CFArray, done: Bool) -> Bool in
        return processHandler(errors: errors, done: done, registrationHandler)
    }
}
```

```
// FontProvider App — Registering All Fonts

static func registerFamilies(_ names: [FontFamily], _ registrationHandler:
    (([Error], Bool) -> Bool)?) {
    let fontURLs = names.flatMap({ $0.urls })
    CTFontManagerRegisterFontURLs(fontURLs as CFArray, .persistent, true) {
        (errors: CFArray, done: Bool) -> Bool in
        return processHandler(errors: errors, done: done, registrationHandler)
    }
}
```

```
// FontProvider App — Registering All Fonts

static func registerFamilies(_ names: [FontFamily], _ registrationHandler:
    (([Error], Bool) -> Bool)?) {
    let fontURLs = names.flatMap({ $0.urls })
    CTFontManagerRegisterFontURLs(fontURLs as CFArray, .persistent, true) {
        (errors: CFArray, done: Bool) -> Bool in
        return processHandler(errors: errors, done: done, registrationHandler)
    }
}
```

```
// FontProvider App – Updating UI for Registered Fonts

static func updateRegisteredFonts() {
    guard let registeredDescriptors =
        CTFontManagerCopyRegisteredFontDescriptors( .persistent, true) as? [CTFontDescriptor]
            else {  return  }
    . . .

    for registeredDescriptor in registeredDescriptors {
        if let postname =
        CTFontDescriptorCopyAttribute(registeredDescriptor, kCTFontNameAttribute) as? String {
            if let font = UIFont(name: postname, size: 12.0) {
                registeredFamilies.insert(font.providerLocalizedFamilyName)
            }
        }
    }
    . . .
}
```

```
// FontProvider App – Updating UI for Registered Fonts

static func updateRegisteredFonts() {
    guard let registeredDescriptors =
        CTFontManagerCopyRegisteredFontDescriptors( .persistent, true) as? [CTFontDescriptor]
            else {  return  }
    . . .

    for registeredDescriptor in registeredDescriptors {
        if let postname =
        CTFontDescriptorCopyAttribute(registeredDescriptor, kCTFontNameAttribute) as? String {
            if let font = UIFont(name: postname, size: 12.0) {
                registeredFamilies.insert(font.providerLocalizedFamilyName)
            }
        }
    }
    . . .
}
```

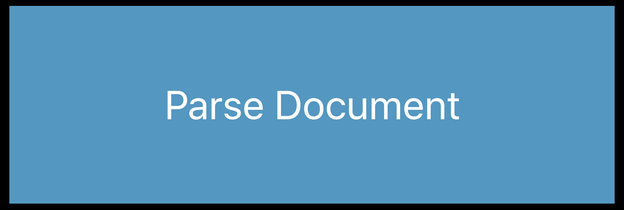
```
// FontProvider App – Updating UI for Registered Fonts

static func updateRegisteredFonts() {
    guard let registeredDescriptors =
        CTFontManagerCopyRegisteredFontDescriptors( .persistent, true) as? [CTFontDescriptor]
            else {  return  }
    . . .

    for registeredDescriptor in registeredDescriptors {
        if let postname =
        CTFontDescriptorCopyAttribute(registeredDescriptor, kCTFontNameAttribute) as? String {
            if let font = UIFont(name: postname, size: 12.0) {
                registeredFamilies.insert(font.providerLocalizedFamilyName)
            }
        }
    }
    . . .
}
```

```
// FontProvider App – Updating UI for Registered Fonts

static func updateRegisteredFonts() {
    guard let registeredDescriptors =
        CTFontManagerCopyRegisteredFontDescriptors( .persistent, true) as? [CTFontDescriptor]
            else {  return  }
    . . .

    for registeredDescriptor in registeredDescriptors {
        if let postname =
        CTFontDescriptorCopyAttribute(registeredDescriptor, kCTFontNameAttribute) as? String {
            if let font = UIFont(name: postname, size: 12.0) {
                registeredFamilies.insert(font.providerLocalizedFamilyName)
            }
        }
    }
    . . .
}
```

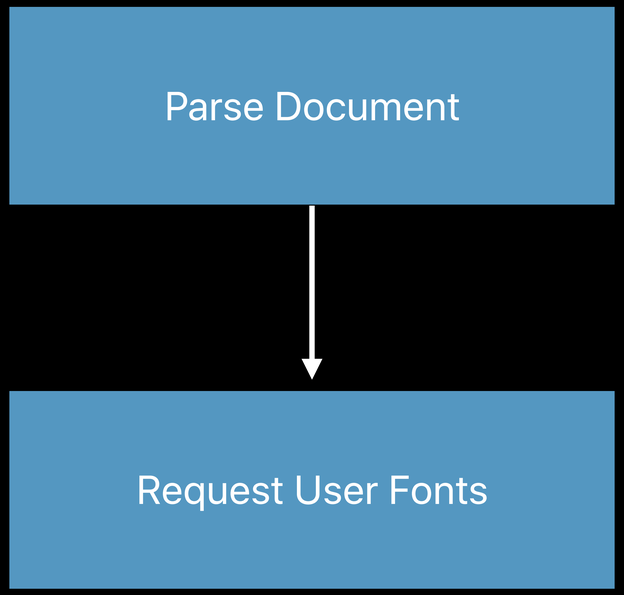
```
// FontProvider App – Updating UI for Registered Fonts

static func updateRegisteredFonts() {
    guard let registeredDescriptors =
        CTFontManagerCopyRegisteredFontDescriptors( .persistent, true) as? [CTFontDescriptor]
            else {  return  }
    . . .

    for registeredDescriptor in registeredDescriptors {
        if let postname =
        CTFontDescriptorCopyAttribute(registeredDescriptor, kCTFontNameAttribute) as? String {
            if let font = UIFont(name: postname, size: 12.0) {
                registeredFamilies.insert(font.providerLocalizedFamilyName)
            }
        }
    }
    . . .
}
```

Font Consumer — Loading Fonts

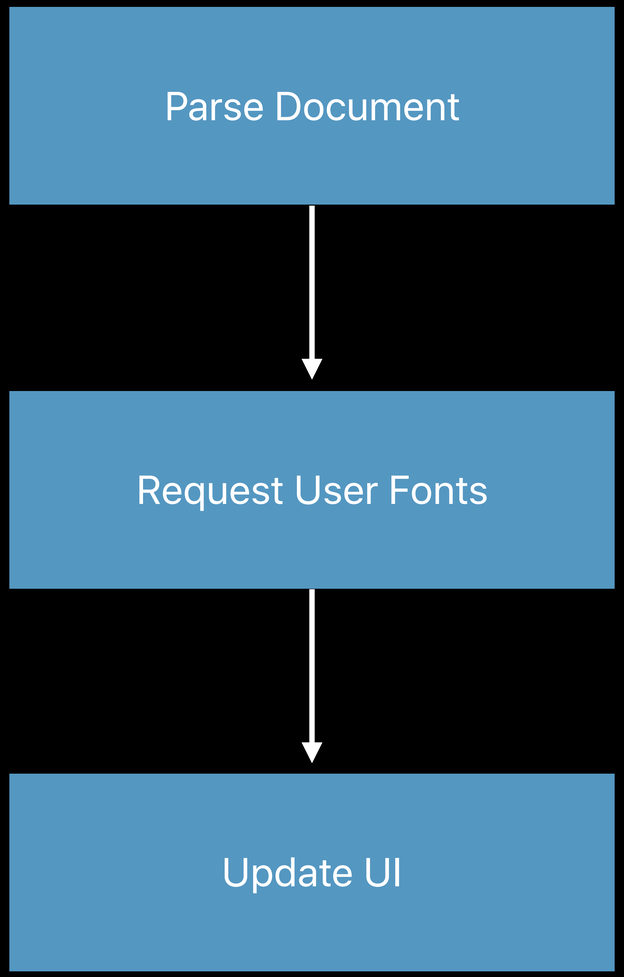
Font Consumer — Loading Fonts



Font Consumer — Loading Fonts



Font Consumer — Loading Fonts



```
// FontConsumer App – Parsing Document

let json = try parseJSON(url: url)

. . .
. . .

for textRun in json {
    let (runString, fontSubstituted) = attributedStringFromRun(textRun)
    if fontSubstituted {
        missingStyleFonts(curStyle)
    }
    attributedString.append(runString)
}
```

```
// FontConsumer App – Parsing Document

let json = try parseJSON(url: url)

. . .
. . .

for textRun in json {
    let (runString, fontSubstituted) = attributedStringFromRun(textRun)
    if fontSubstituted {
        missingStyleFonts(curStyle)
    }
    attributedString.append(runString)
}
```

```
// FontConsumer App – Parsing Document

let json = try parseJSON(url: url)

. . .
. . .

for textRun in json {
    let (runString, fontSubstituted) = attributedStringFromRun(textRun)
    if fontSubstituted {
        missingStyleFonts(curStyle)
    }
    attributedString.append(runString)
}
```

```
// FontConsumer App – Parsing Document

let json = try parseJSON(url: url)

. . .
. . .

for textRun in json {
    let (runString, fontSubstituted) = attributedStringFromRun(textRun)
    if fontSubstituted {
        missingStyleFonts(curStyle)
    }
    attributedString.append(runString)
}
```

```
// FontConsumer App – Parsing Document

let json = try parseJSON(url: url)

. . .
. . .

for textRun in json {
    let (runString, fontSubstituted) = attributedStringFromRun(textRun)
    if fontSubstituted {
        missingStyleFonts(curStyle)
    }
    attributedString.append(runString)
}
```

```
// FontConsumer App — Request User Fonts
    . . .

    let fontsToAsk = Array(theMissingFonts).map { fontName -> UIFontDescriptor in
         return UIFontDescriptor(fontAttributes:
                                 [UIFontDescriptor.AttributeName.name: fontName])
     }

     CTFontManagerRequestFonts(fontsToAsk as CFArray) { (unresolved: CFArray) in
         DispatchQueue.main.async {
             if checkMissingFontsAndUpdateStringAttributes() || attributesUpdated {
                 self.delegate?.textChanged(self.attributedString)
             }
        }
     }

    . . .
```

```
// FontConsumer App — Request User Fonts
    . . .

    let fontsToAsk = Array(theMissingFonts).map { fontName -> UIFontDescriptor in
         return UIFontDescriptor(fontAttributes:
                                 [UIFontDescriptor.AttributeName.name: fontName])
     }

     CTFontManagerRequestFonts(fontsToAsk as CFArray) { (unresolved: CFArray) in
         DispatchQueue.main.async {
             if checkMissingFontsAndUpdateStringAttributes() || attributesUpdated {
                 self.delegate?.textChanged(self.attributedString)
             }
        }
     }

    . . .
```

```
// FontConsumer App — Request User Fonts
    . . .

    let fontsToAsk = Array(theMissingFonts).map { fontName -> UIFontDescriptor in
         return UIFontDescriptor(fontAttributes:
                                 [UIFontDescriptor.AttributeName.name: fontName])
     }

     CTFontManagerRequestFonts(fontsToAsk as CFArray) { (unresolved: CFArray) in
         DispatchQueue.main.async {
             if checkMissingFontsAndUpdateStringAttributes() || attributesUpdated {
                 self.delegate?.textChanged(self.attributedString)
             }
        }
     }

    . . .
```

```
// FontConsumer App — Request User Fonts
    . . .

    let fontsToAsk = Array(theMissingFonts).map { fontName -> UIFontDescriptor in
         return UIFontDescriptor(fontAttributes:
                                 [UIFontDescriptor.AttributeName.name: fontName])
     }

     CTFontManagerRequestFonts(fontsToAsk as CFArray) { (unresolved: CFArray) in
         DispatchQueue.main.async {
             if checkMissingFontsAndUpdateStringAttributes() || attributesUpdated {
                 self.delegate?.textChanged(self.attributedString)
             }
        }
     }

    . . .
```

```
// FontConsumer App — Request User Fonts
    . . .

    let fontsToAsk = Array(theMissingFonts).map { fontName -> UIFontDescriptor in
         return UIFontDescriptor(fontAttributes:
                                 [UIFontDescriptor.AttributeName.name: fontName])
     }

     CTFontManagerRequestFonts(fontsToAsk as CFArray) { (unresolved: CFArray) in
         DispatchQueue.main.async {
             if checkMissingFontsAndUpdateStringAttributes() || attributesUpdated {
                 self.delegate?.textChanged(self.attributedString)
             }
        }
     }

    . . .
```

```
// FontConsumer App — Request User Fonts
    . . .

    let fontsToAsk = Array(theMissingFonts).map { fontName -> UIFontDescriptor in
         return UIFontDescriptor(fontAttributes:
                                 [UIFontDescriptor.AttributeName.name: fontName])
     }

     CTFontManagerRequestFonts(fontsToAsk as CFArray) { (unresolved: CFArray) in
         DispatchQueue.main.async {
             if checkMissingFontsAndUpdateStringAttributes() || attributesUpdated {
                 self.delegate?.textChanged(self.attributedString)
             }
        }
     }

    . . .
```

```
// FontConsumer App — Request User Fonts
    . . .

    let fontsToAsk = Array(theMissingFonts).map { fontName -> UIFontDescriptor in
         return UIFontDescriptor(fontAttributes:
                                 [UIFontDescriptor.AttributeName.name: fontName])
     }

     CTFontManagerRequestFonts(fontsToAsk as CFArray) { (unresolved: CFArray) in
         DispatchQueue.main.async {
             if checkMissingFontsAndUpdateStringAttributes() || attributesUpdated {
                 self.delegate?.textChanged(self.attributedString)
             }
         }
     }

    . . .
```

Installing Fonts

Considerations

Installing Fonts

Considerations

Registration operations limited to Application’s fonts

Installing Fonts

Considerations

Registration operations limited to Application’s fonts

Cannot override fonts already registered

Installing Fonts

Considerations

Registration operations limited to Application’s fonts

Cannot override fonts already registered

Number of registered fonts is limited by the OS

Installing Fonts

Considerations

Registration operations limited to Application’s fonts

Cannot override fonts already registered

Number of registered fonts is limited by the OS

User fonts do not participate in font fallback

Installing Fonts

Considerations

Registration operations limited to Application’s fonts

Cannot override fonts already registered

Number of registered fonts is limited by the OS

User fonts do not participate in font fallback

Fonts are removed when the application is deleted

Font Selection

Eric Dudiak, UIKit Engineer

Font Selection

Enumerating fonts

Privacy concern

Only provide system fonts

Font Selection

NEW

UIFontPickerViewController

- User font selection

Present

- New modal style

Embed

UIFontPickerViewController

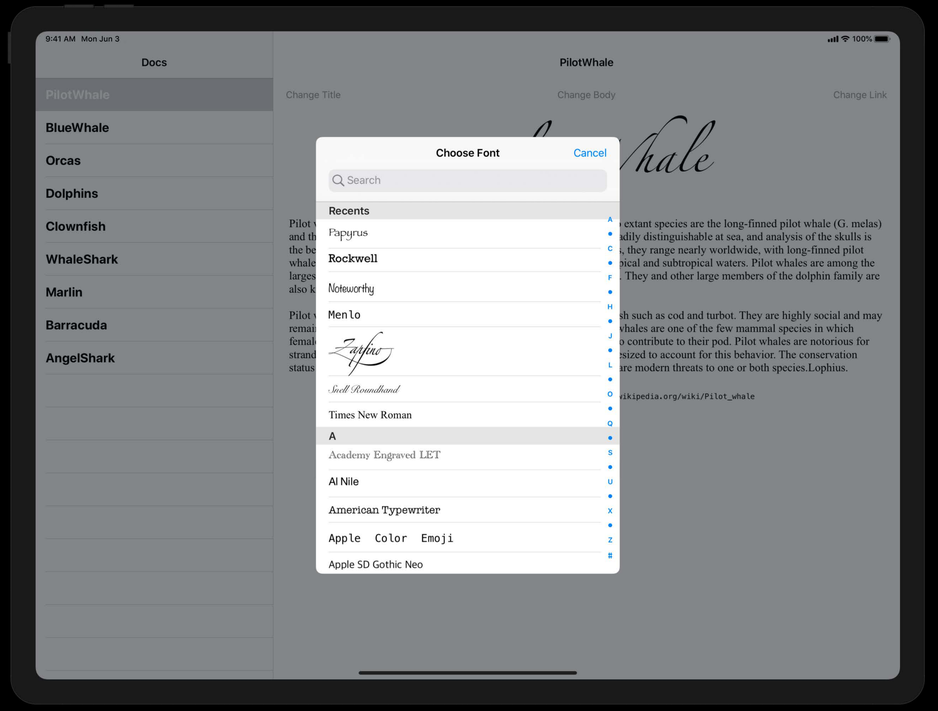
Out of process

Entitlement

- Access user-installed fonts

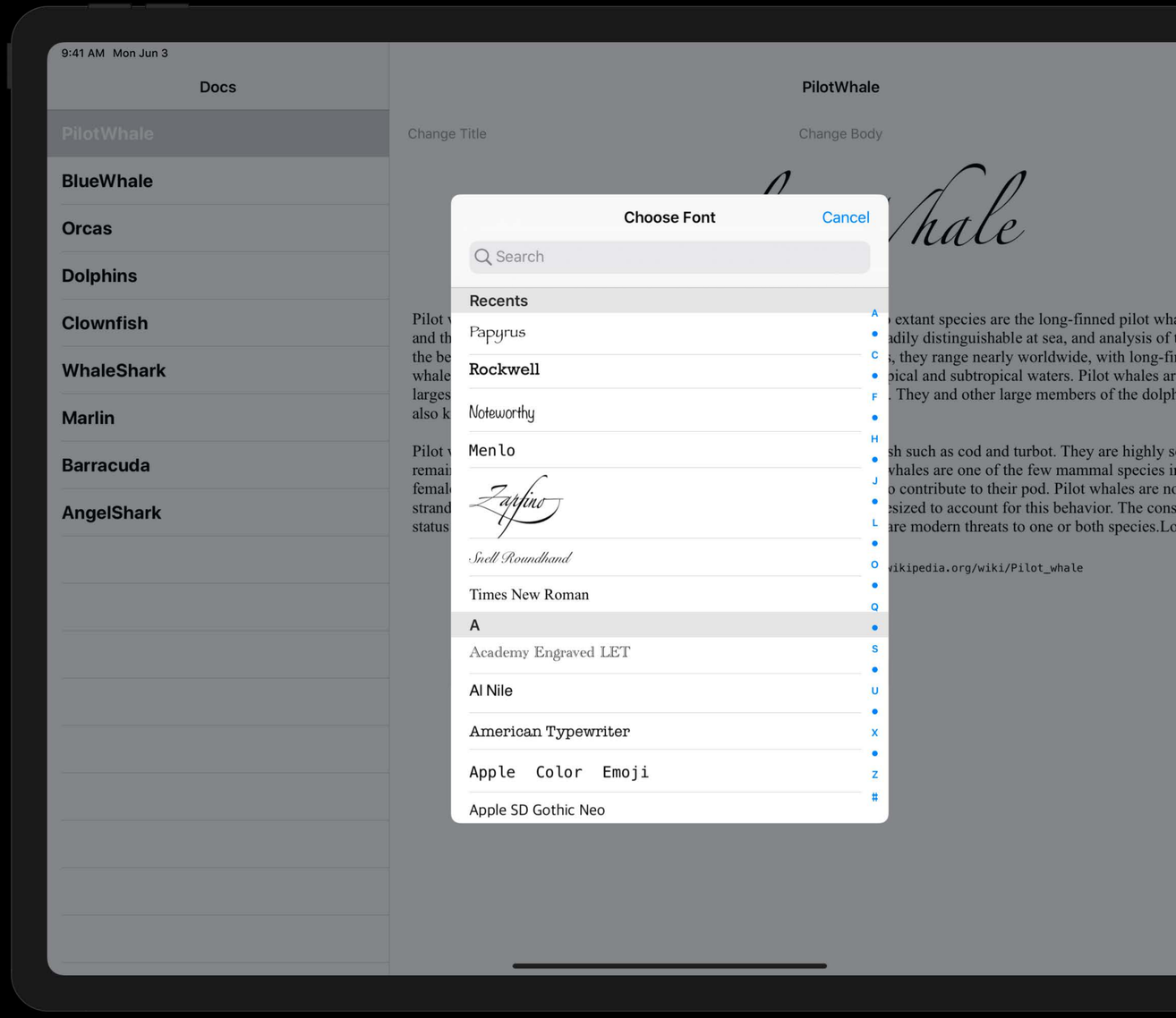
Font available after selection

- No additional API needed after selected



UIFontPickerViewController

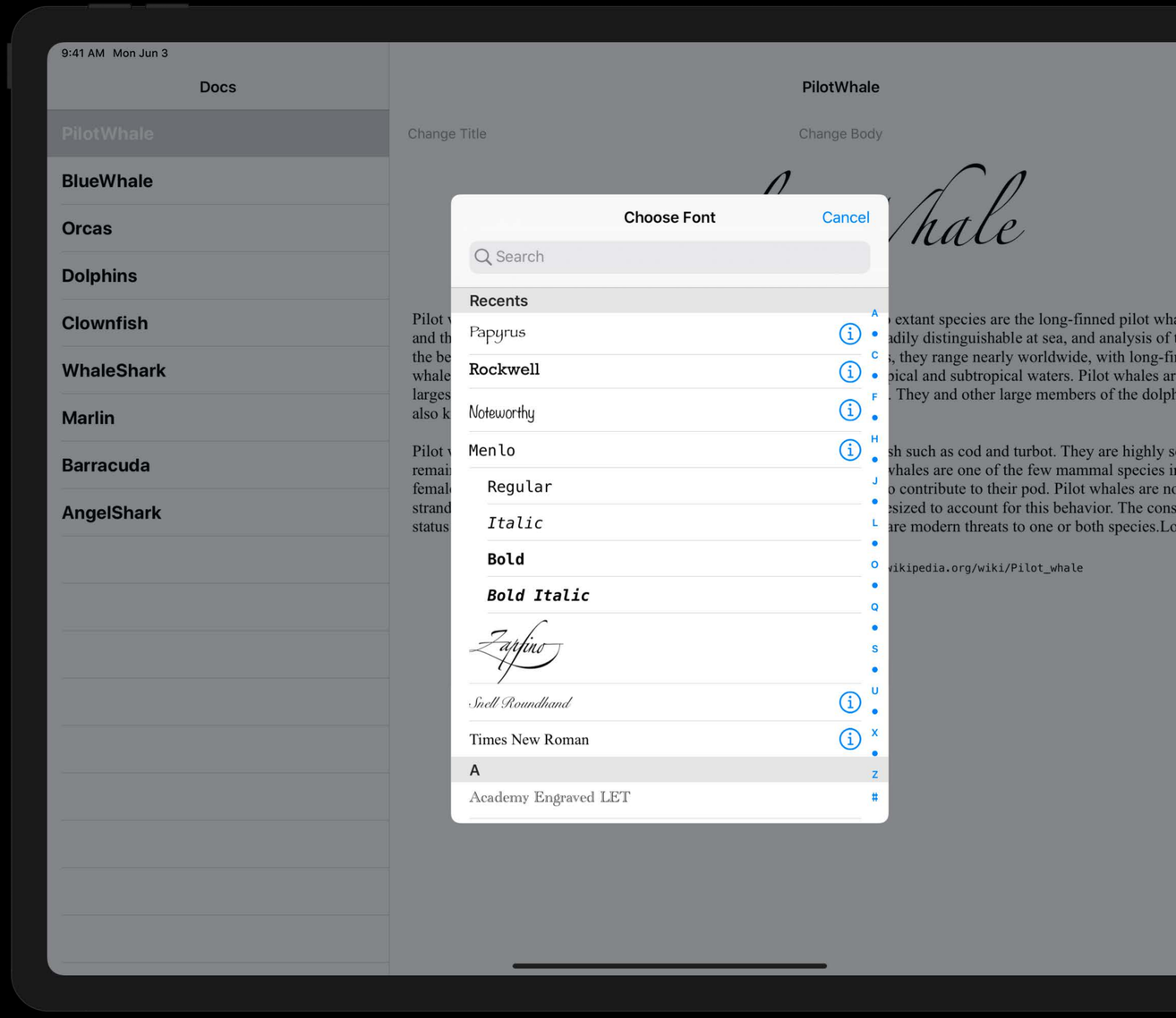
Customization



UIFontPickerViewController

Customization

- Show faces
- WYSIWYG



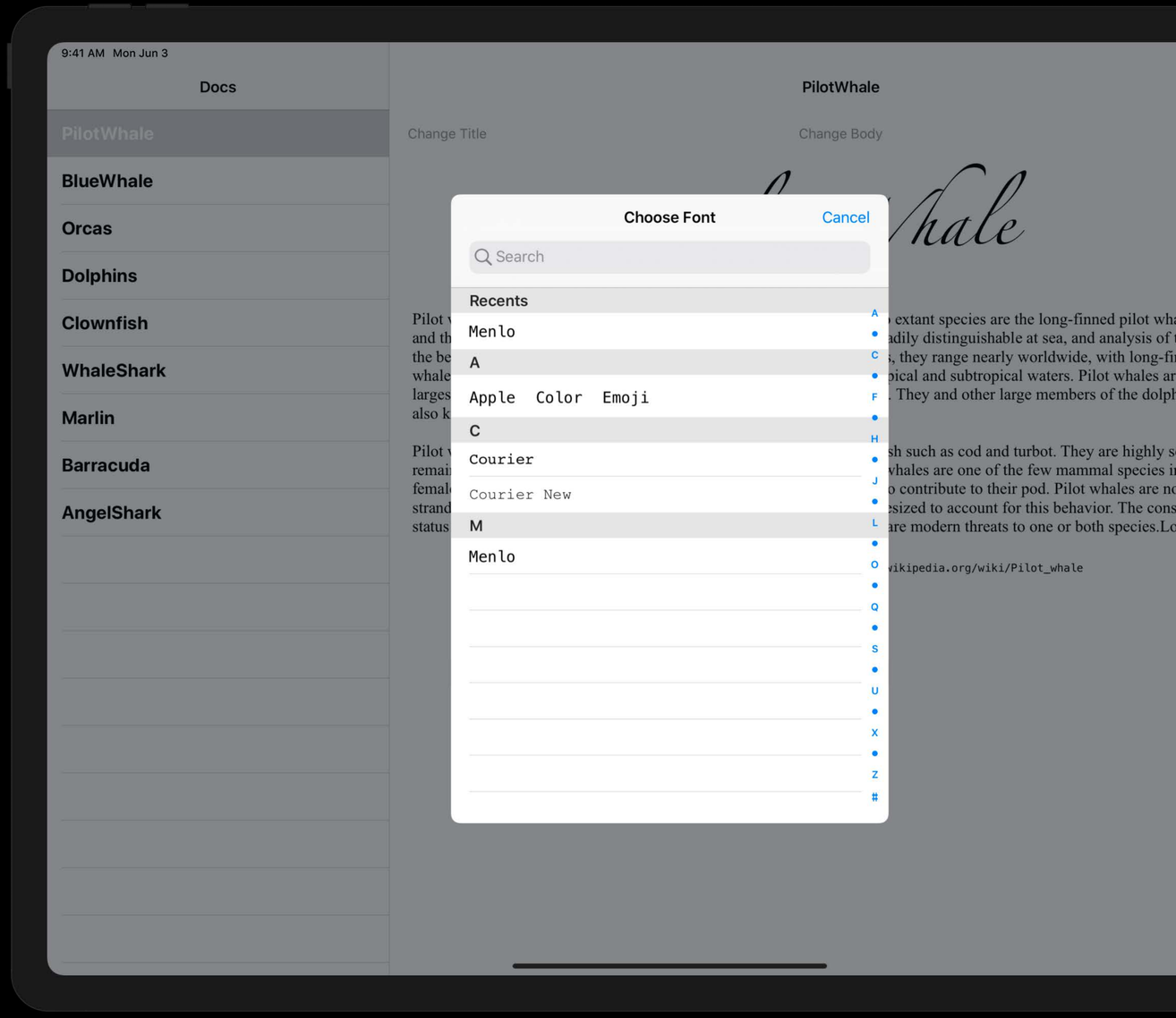
UIFontPickerViewController

Customization

- Show faces
- WYSIWYG

Filtering

- Traits



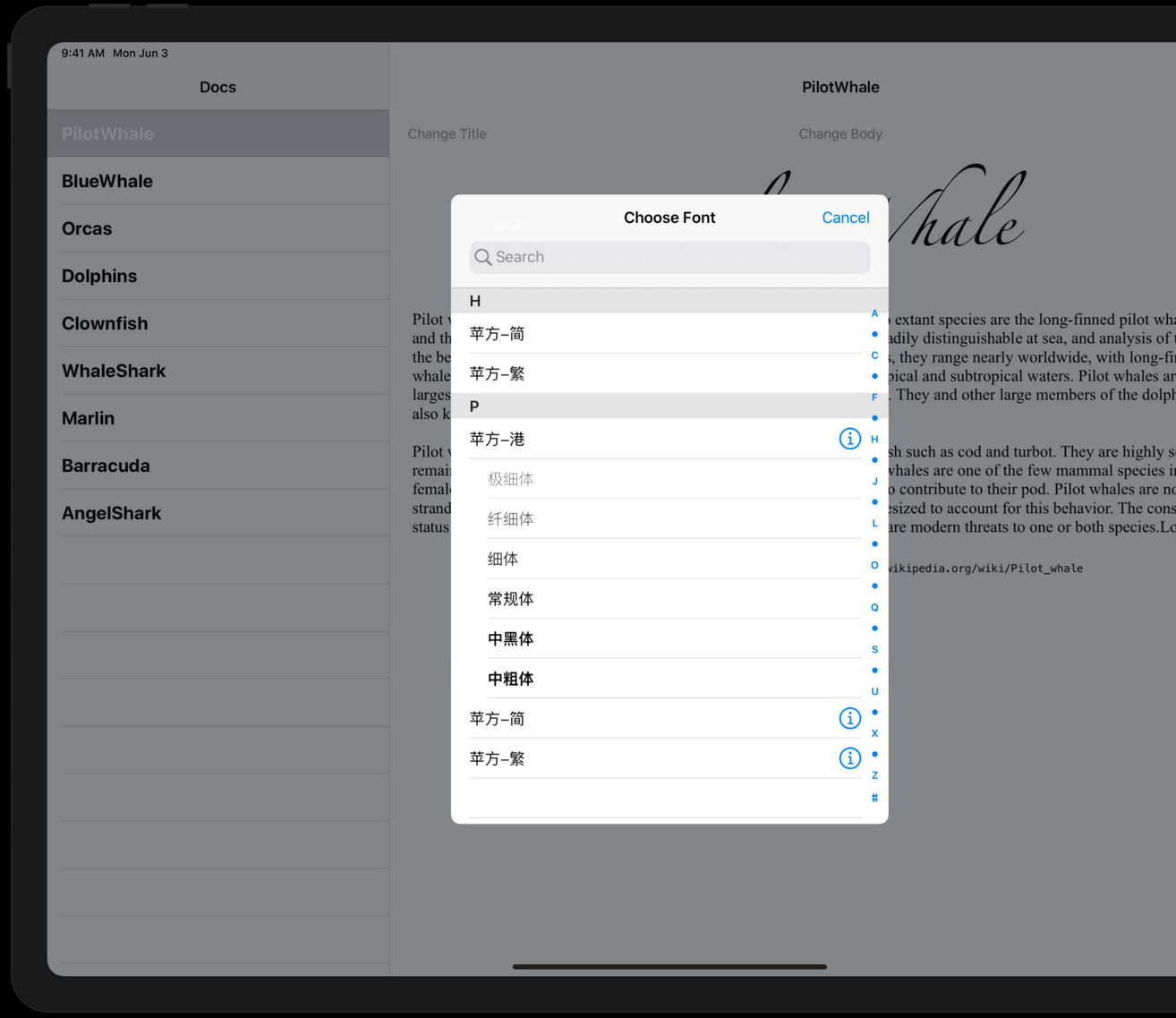
UIFontPickerViewController

Customization

- Show faces
- WYSIWYG

Filtering

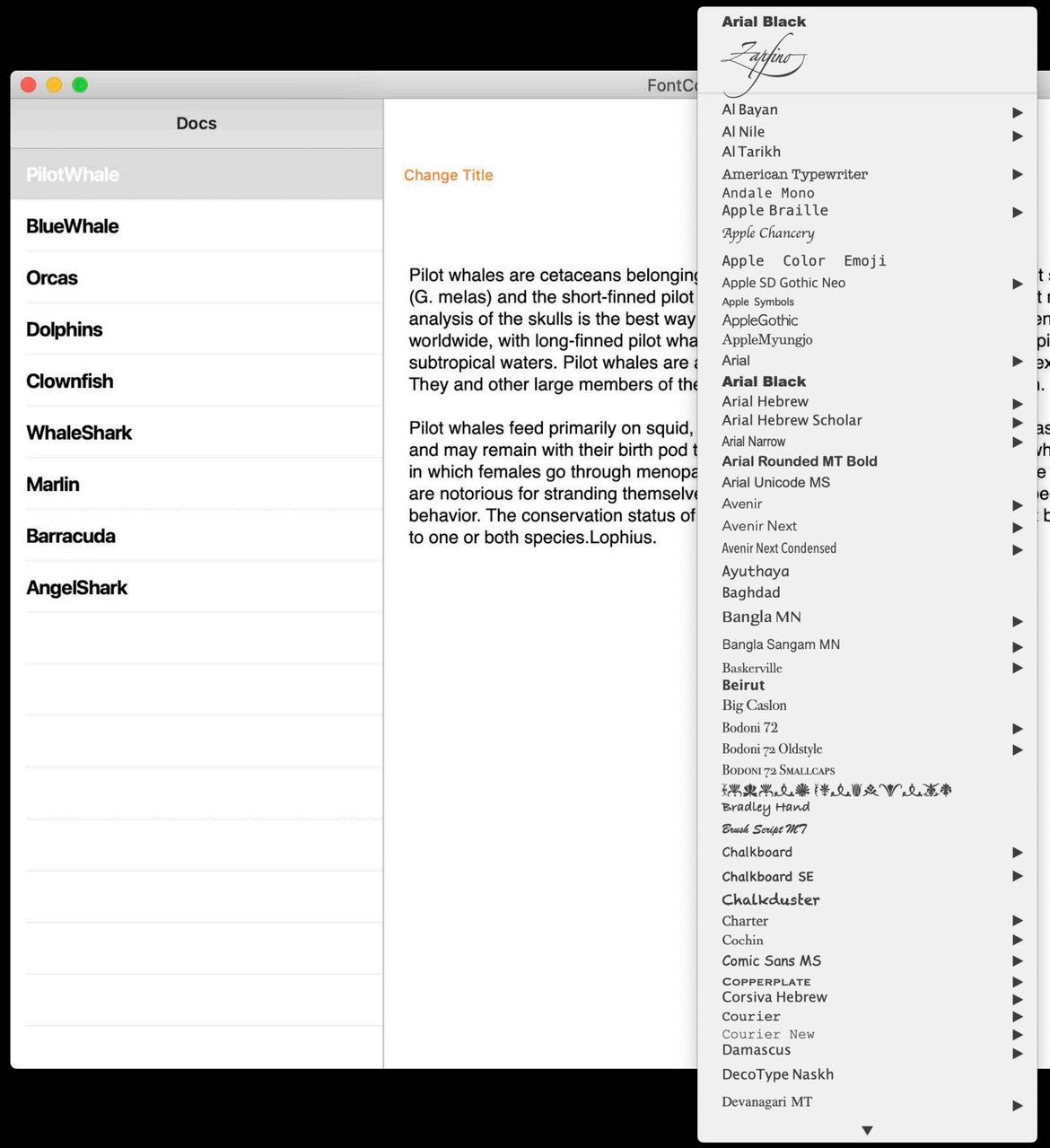
- Traits
- Languages



macOS

Presentation

- Menu by default
- Use UIPopoverPresentationController
  - Determines menu position
- Semantics the same



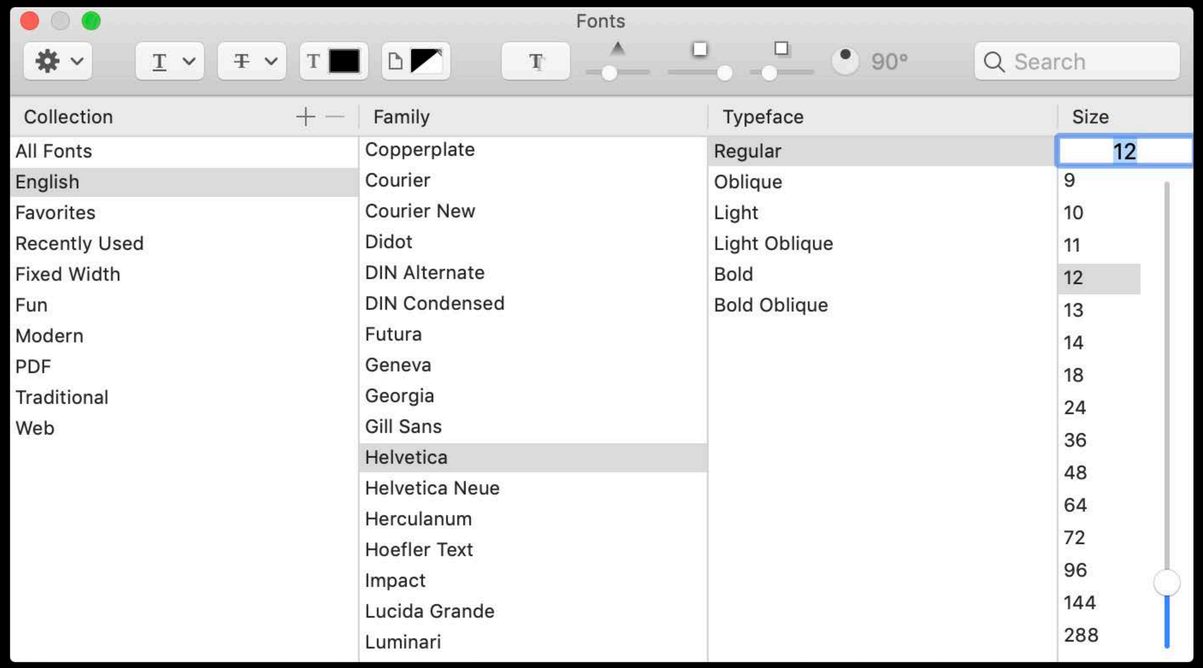
macOS Font Panel

Available on Mac

In default menu bar menus

UITextFormattingCoordinator

- Toggle
- Detect visibility



Responder Chain

NEW

UITextFormattingCoordinator

- Uses responder chain
- Can be used as font picker delegate

UIResponderStandardEditActions protocol

- Respond to changes
- Custom controls

Demo

```
// Create a configuration to customize
let fontPickerConfig = UIFontPickerViewControllerConfiguration()
fontPickerConfig.includeFaces = true
```

```
// Create a configuration to customize
let fontPickerConfig = UIFontPickerViewControllerConfiguration()
fontPickerConfig.includeFaces = true
```

```swift
// Create a configuration to customize
let fontPickerConfig = UIFontPickerViewControllerConfiguration()
fontPickerConfig.includeFaces = true

// Create the font picker
let fontPicker = UIFontPickerViewController(configuration: configuration)
fontPicker.delegate = self
```

```swift
// Create a configuration to customize
let fontPickerConfig = UIFontPickerViewControllerConfiguration()
fontPickerConfig.includeFaces = true

// Create the font picker
let fontPicker = UIFontPickerViewController(configuration: configuration)
fontPicker.delegate = self
```

```swift
// Create a configuration to customize
let fontPickerConfig = UIFontPickerViewControllerConfiguration()
fontPickerConfig.includeFaces = true

// Create the font picker
let fontPicker = UIFontPickerViewController(configuration: configuration)
fontPicker.delegate = self

// Present the font picker
self.present(fontPicker, animated: true completion: nil)
```

```swift
// Create a configuration to customize
let fontPickerConfig = UIFontPickerViewControllerConfiguration()
fontPickerConfig.includeFaces = true

// Create the font picker
let fontPicker = UIFontPickerViewController(configuration: configuration)
fontPicker.delegate = self

// Present the font picker
self.present(fontPicker, animated: true completion: nil)
```

```
func fontPickerViewControllerDidPickFont(_ fontPicker: UIFontPickerViewController) {
    _ = detailItem?.replaceFont(style: changeFontStyle ?? "Body", fontName:
fontPicker.selectedFontDescriptor?.postscriptName ?? "Helvetica" )
    changeFontStyle = nil
}
```

```
func fontPickerViewControllerDidPickFont(_ fontPicker: UIFontPickerViewController) {
    _ = detailItem?.replaceFont(style: changeFontStyle ?? "Body", fontName:
fontPicker.selectedFontDescriptor?.postscriptName ?? "Helvetica" )
    changeFontStyle = nil
}
```

```
func fontPickerViewControllerDidPickFont(_ fontPicker: UIFontPickerViewController) {
    _ = detailItem?.replaceFont(style: changeFontStyle ?? "Body", fontName:
fontPicker.selectedFontDescriptor?.postscriptName ?? "Helvetica" )
    changeFontStyle = nil
}

func fontPickerViewControllerDidCancel(_ viewController: UIFontPickerViewController) {
    changeFontStyle = nil
}
```

```
func fontPickerViewControllerDidPickFont(_ fontPicker: UIFontPickerViewController) {
    _ = detailItem?.replaceFont(style: changeFontStyle ?? "Body", fontName:
fontPicker.selectedFontDescriptor?.postscriptName ?? "Helvetica" )
    changeFontStyle = nil
}

func fontPickerViewControllerDidCancel(_ viewController: UIFontPickerViewController) {
    changeFontStyle = nil
}
```

```
// UIResponderStandardEditActions protocol
func updateTextAttributes(conversionHandler: UITextAttributesConversionHandler) {

    // Create a new mutable string
    let newString = NSMutableAttributedString(string: attributedString.string)

    // Enumerate attributes to modify
    attributedString.enumerateAttributes(in: NSMakeRange(0, attributedString.length),
        options: []) { (attributeDictionary, range, stop) in

        // Get the updated attributes
        newString.setAttributes(conversionHandler(attributeDictionary), range: range)
    }
}
```

Font Selection

New font picker

Font panel on macOS

Attributes for custom responders

Text Scaling

Donna Tom, TextKit Engineer

Text Scaling

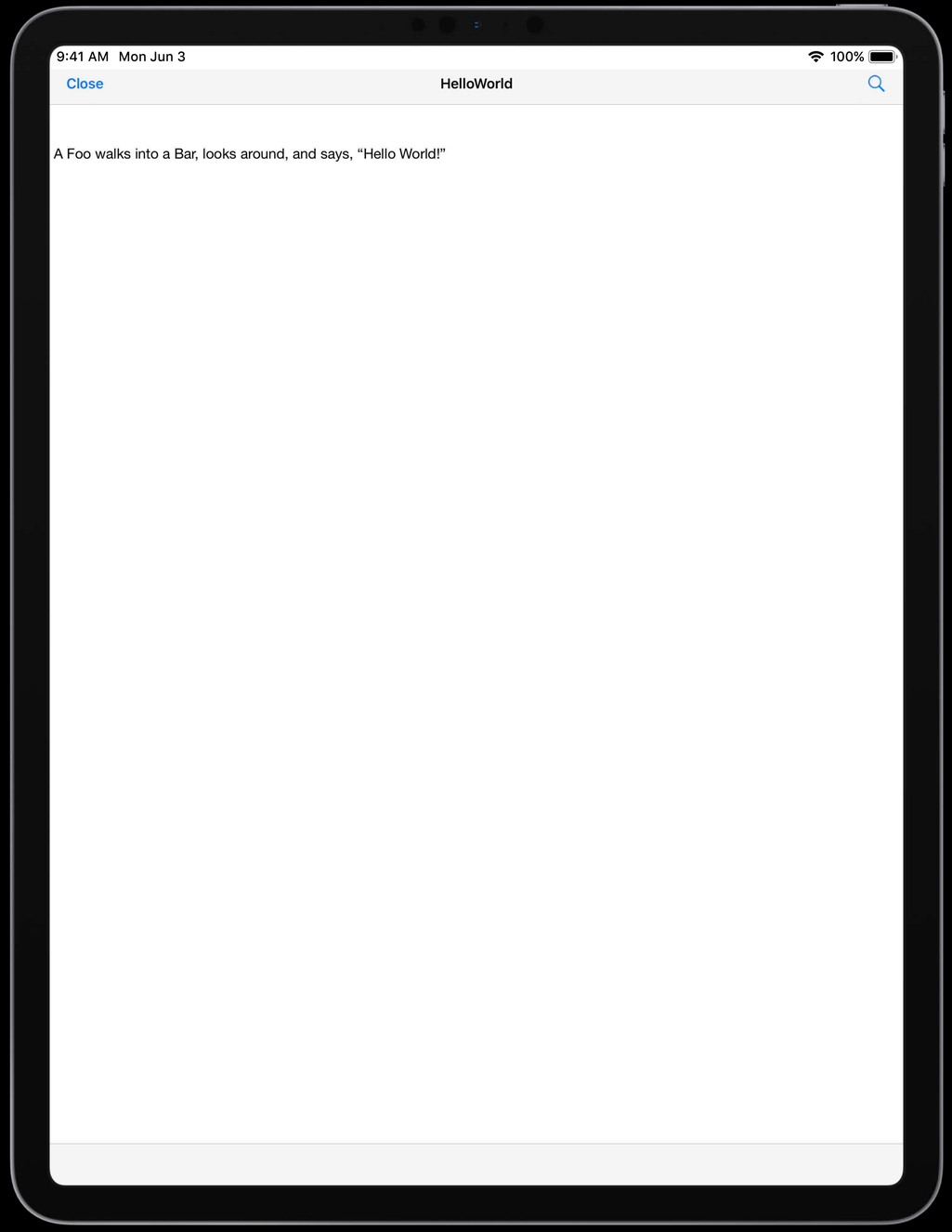
Dynamic Type size table for Large (Default)

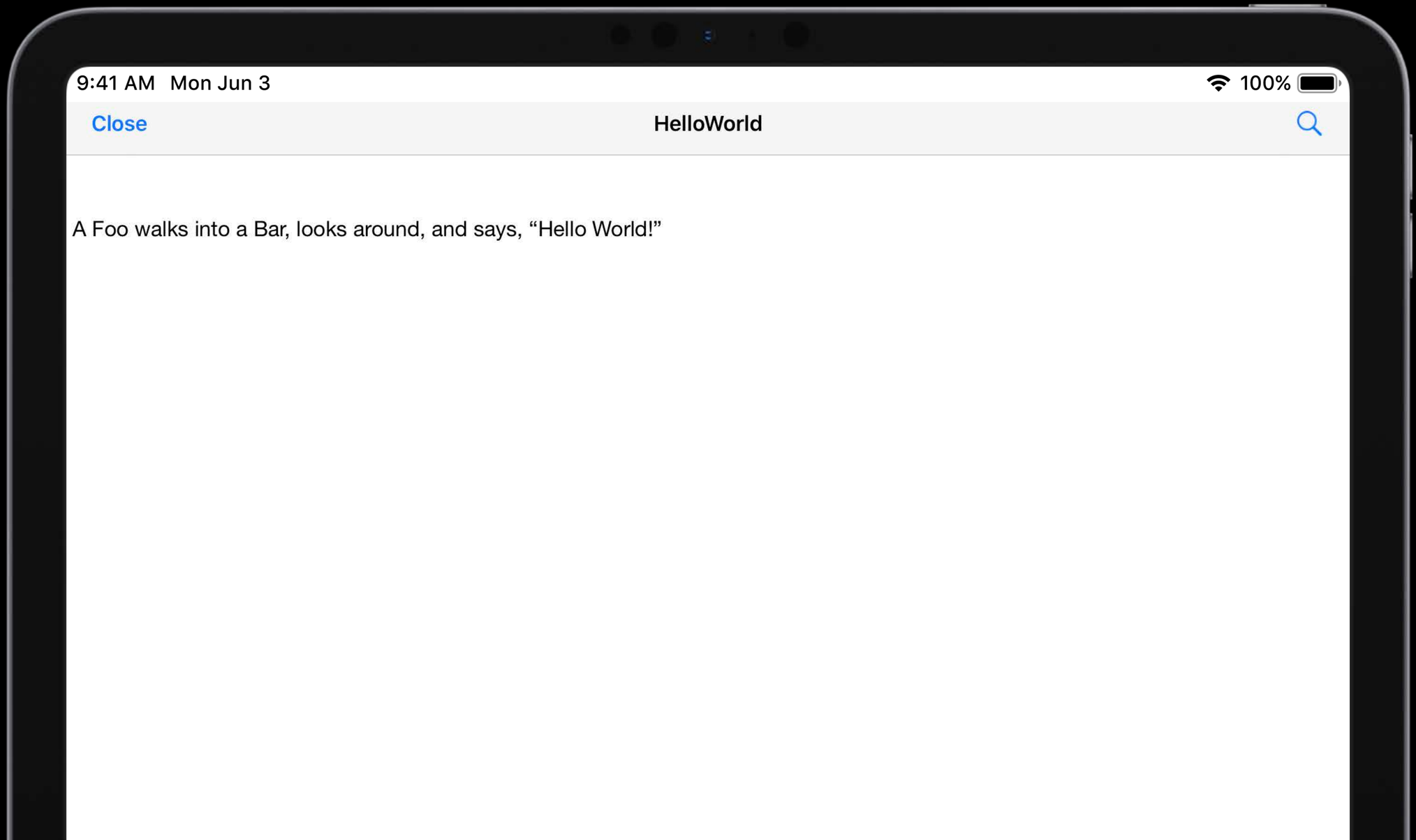
Style	Weight	Size (Points)
Headline	Semibold	17
Body	Regular	17
Callout	Regular	16
Subhead	Regular	15
Footnote	Regular	13

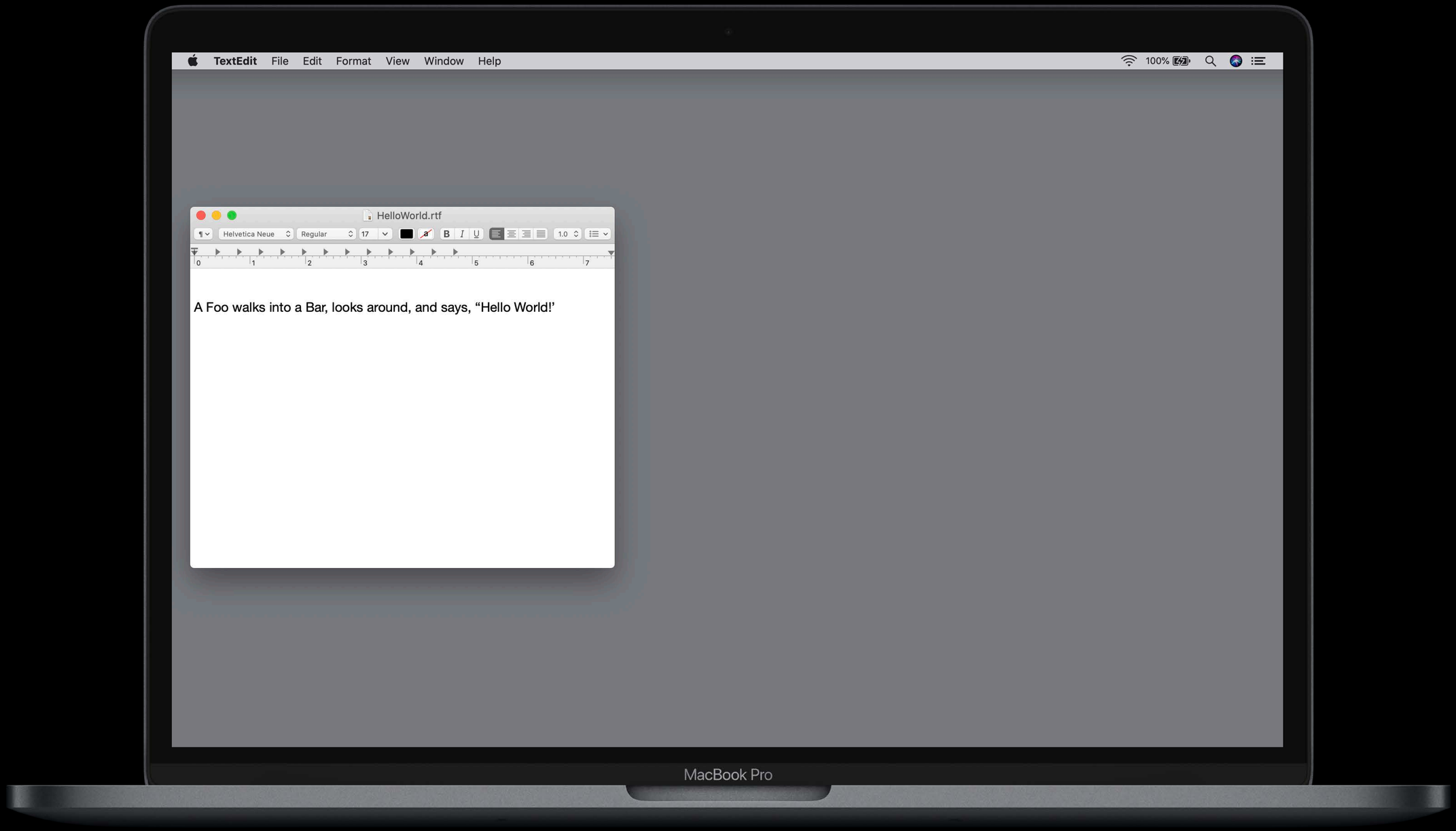
Text Scaling

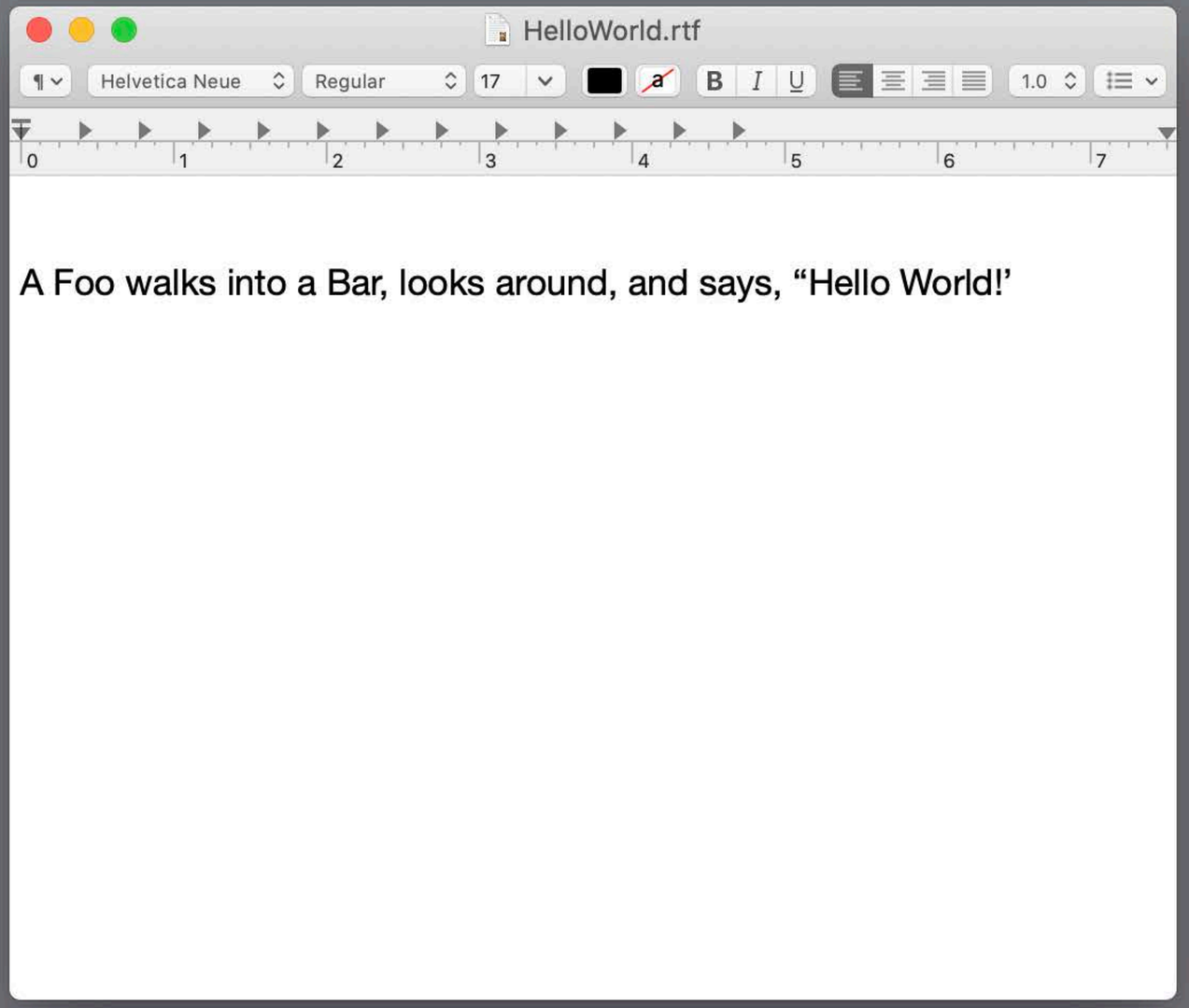
Dynamic Type size table for Large (Default)

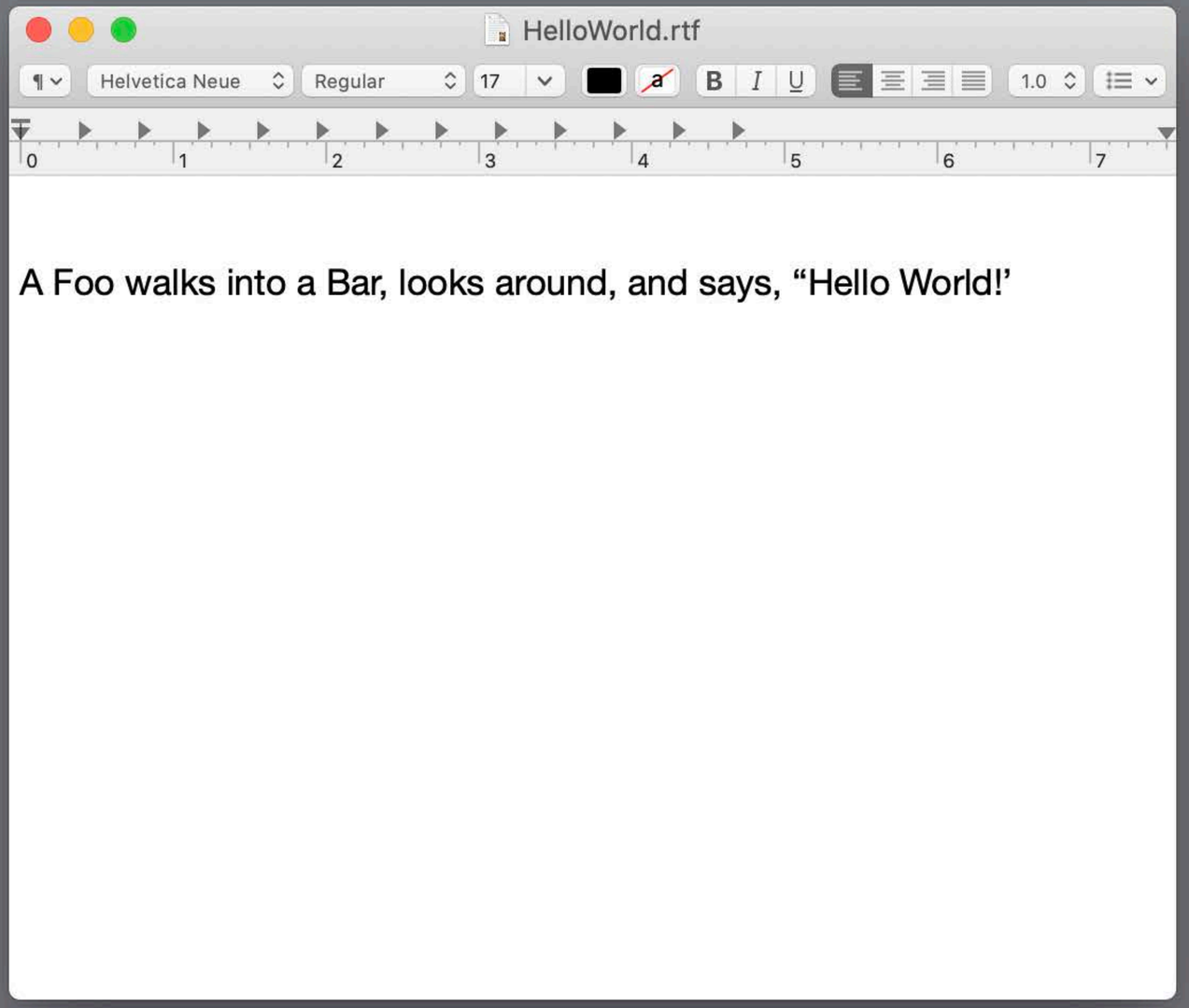
Style	Weight	Size (Points)
Headline	Semibold	17
Body	Regular	17
Callout	Regular	16
Subhead	Regular	15
Footnote	Regular	13

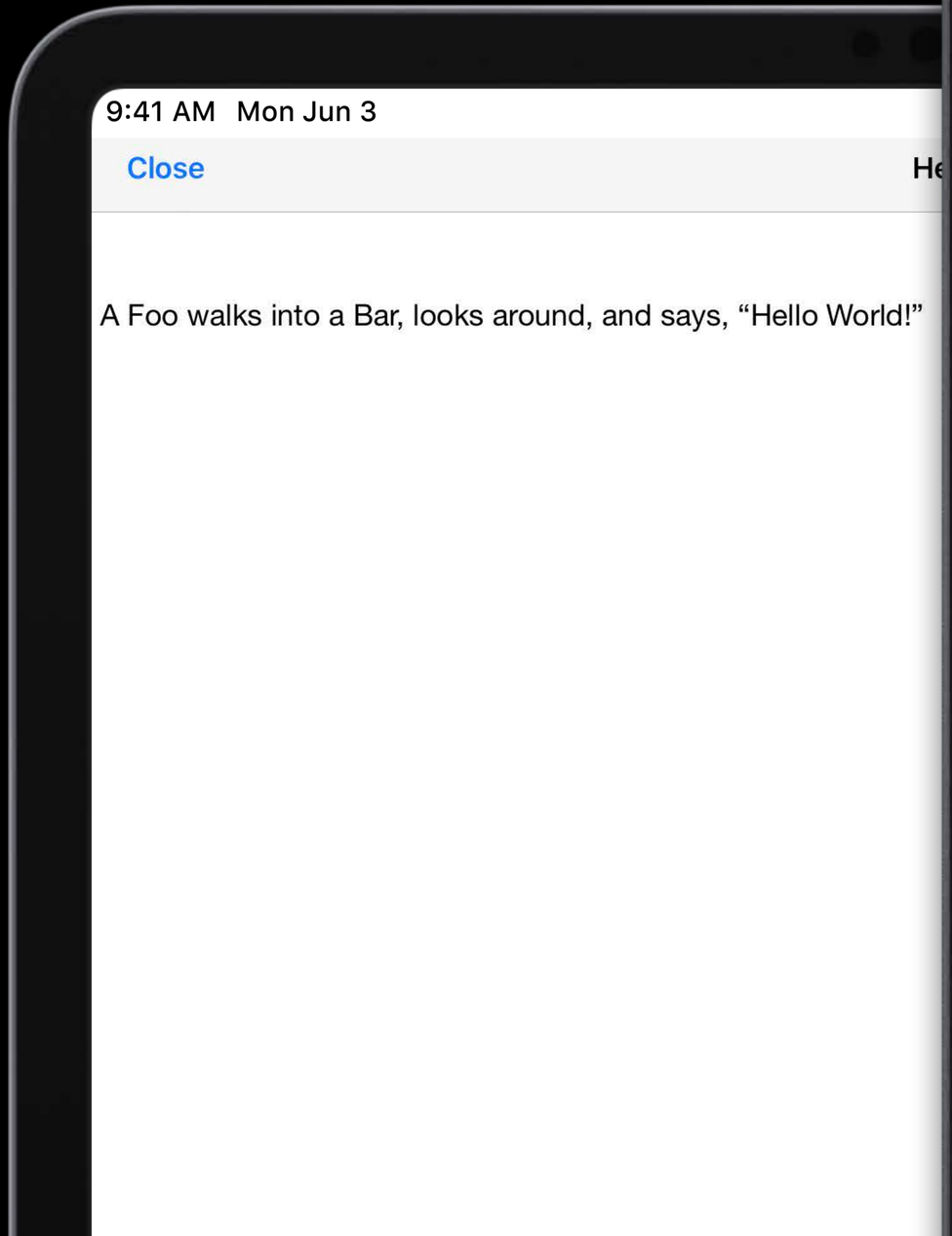

9:41 AM Mon Jun 3
100%
Close
HelloWorld
A Foo walks into a Bar, looks around, and says, “Hello World!”

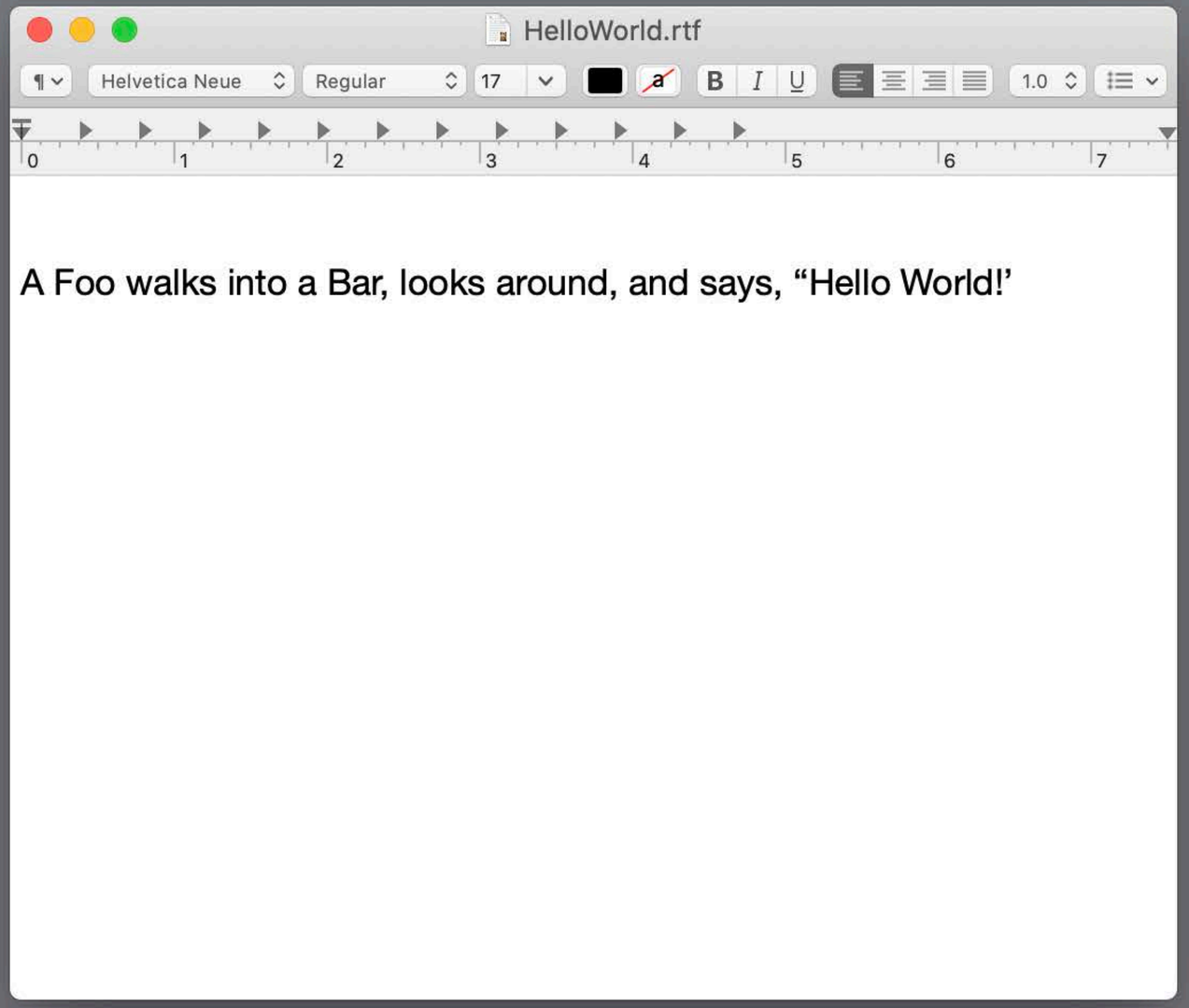

9:41 AM Mon Jun 3
100%
Close
HelloWorld
A Foo walks into a Bar, looks around, and says, “Hello World!”

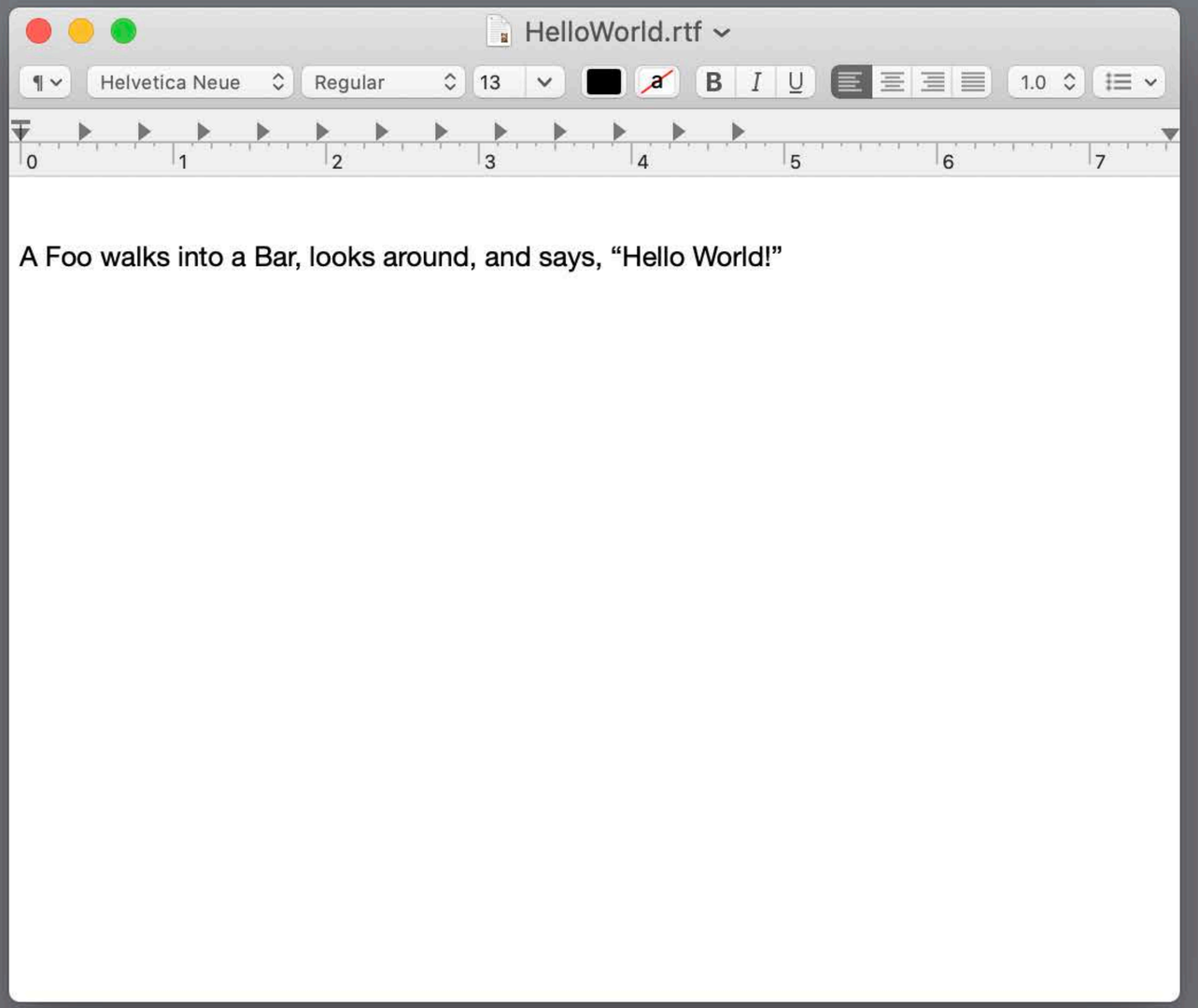

TextEdit File Edit Format View Window Help
100%
HelloWorld.rtf
Helvetica Neue
Regular
17
1.0
A Foo walks into a Bar, looks around, and says, “Hello World!’
MacBook Pro

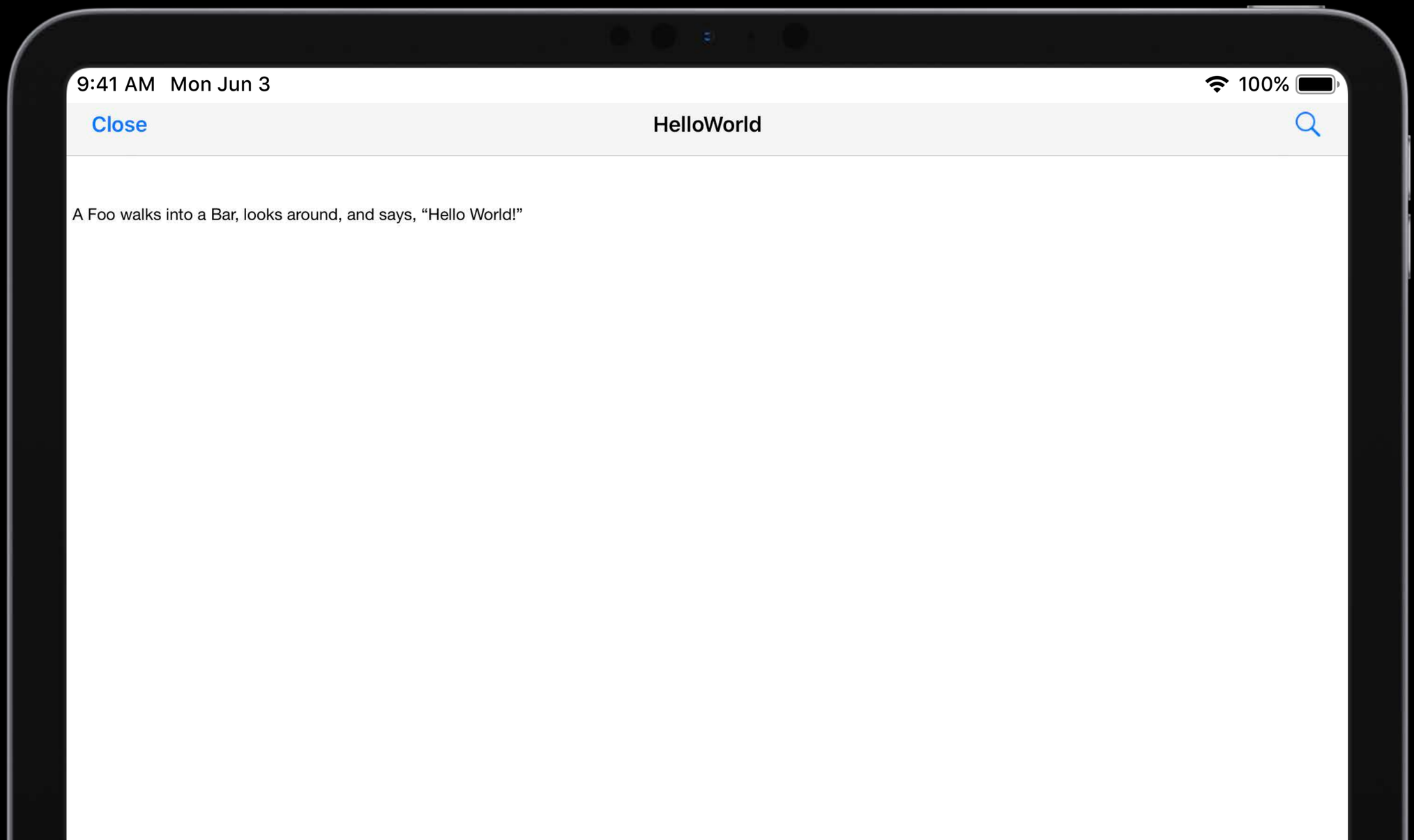

HelloWorld.rtf
Helvetica Neue
Regular
17
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!’

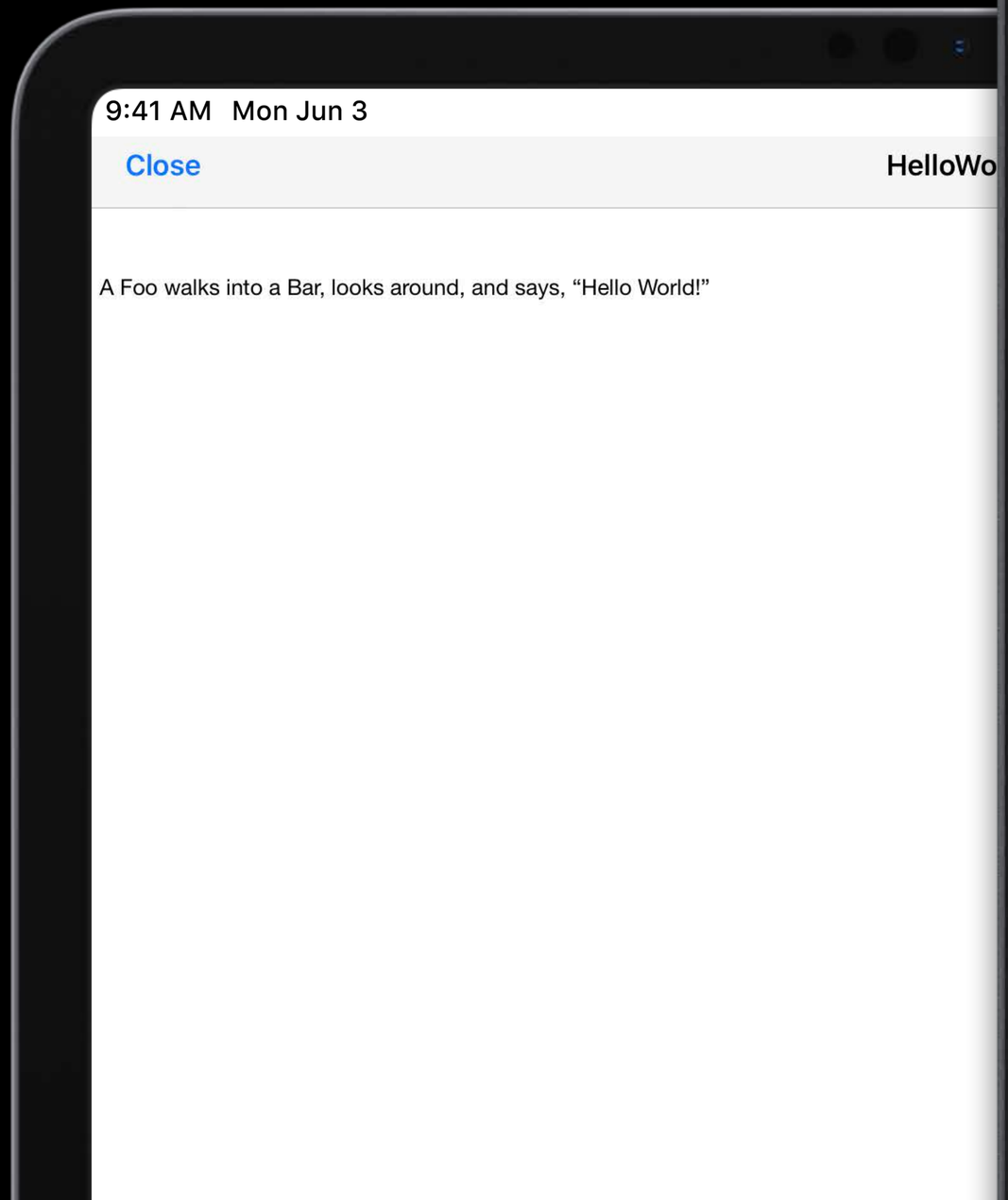

HelloWorld.rtf
Helvetica Neue
Regular
17
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!’

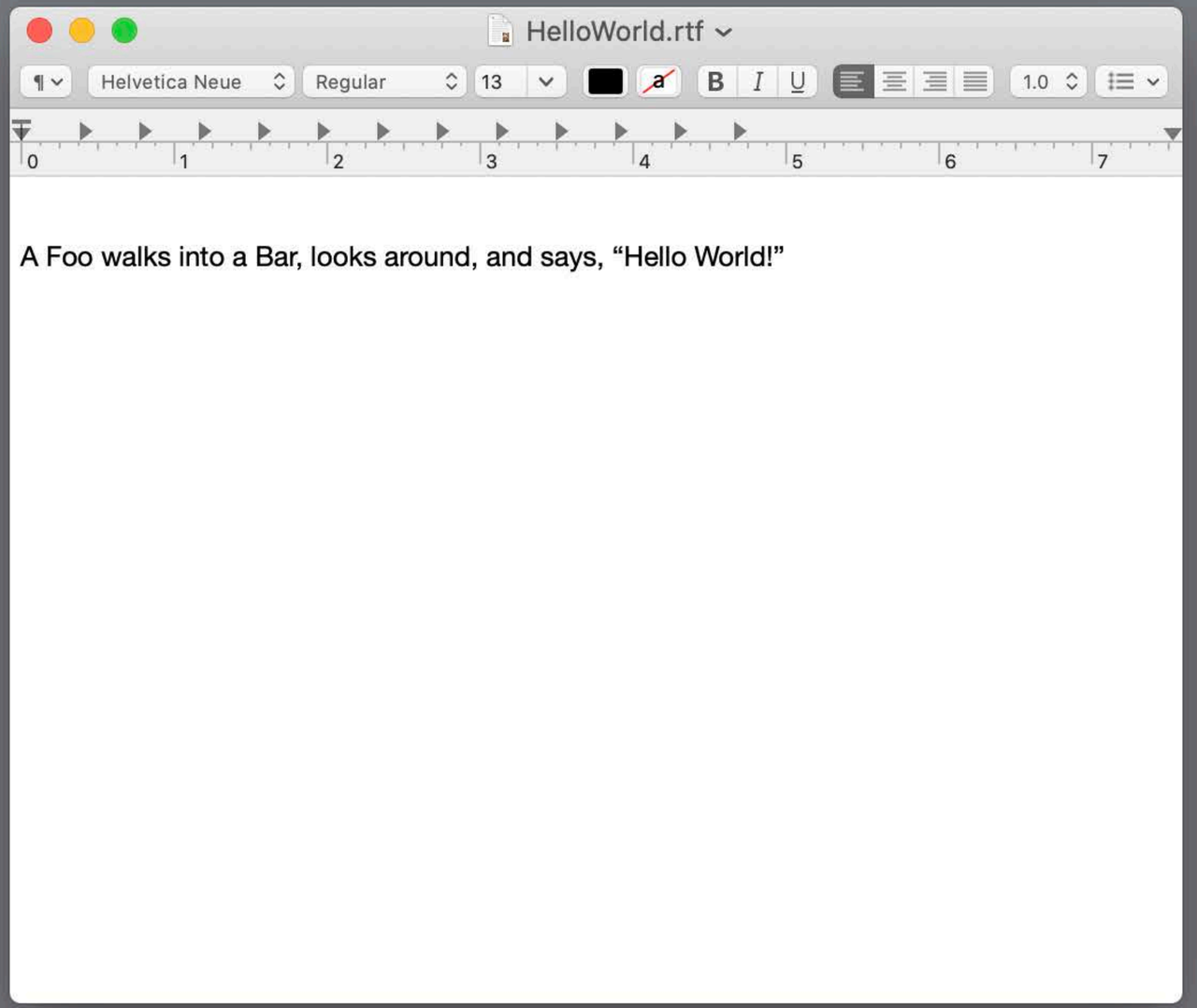

9:41 AM Mon Jun 3
Close
A Foo walks into a Bar, looks around, and says, “Hello World!”


HelloWorld.rtf
Helvetica Neue
Regular
17
1.0
A Foo walks into a Bar, looks around, and says, “Hello World!’


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!”


9:41 AM Mon Jun 3
100%
Close
HelloWorld
A Foo walks into a Bar, looks around, and says, “Hello World!”


9:41 AM Mon Jun 3
Close
HelloWo
A Foo walks into a Bar, looks around, and says, “Hello World!”


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!”

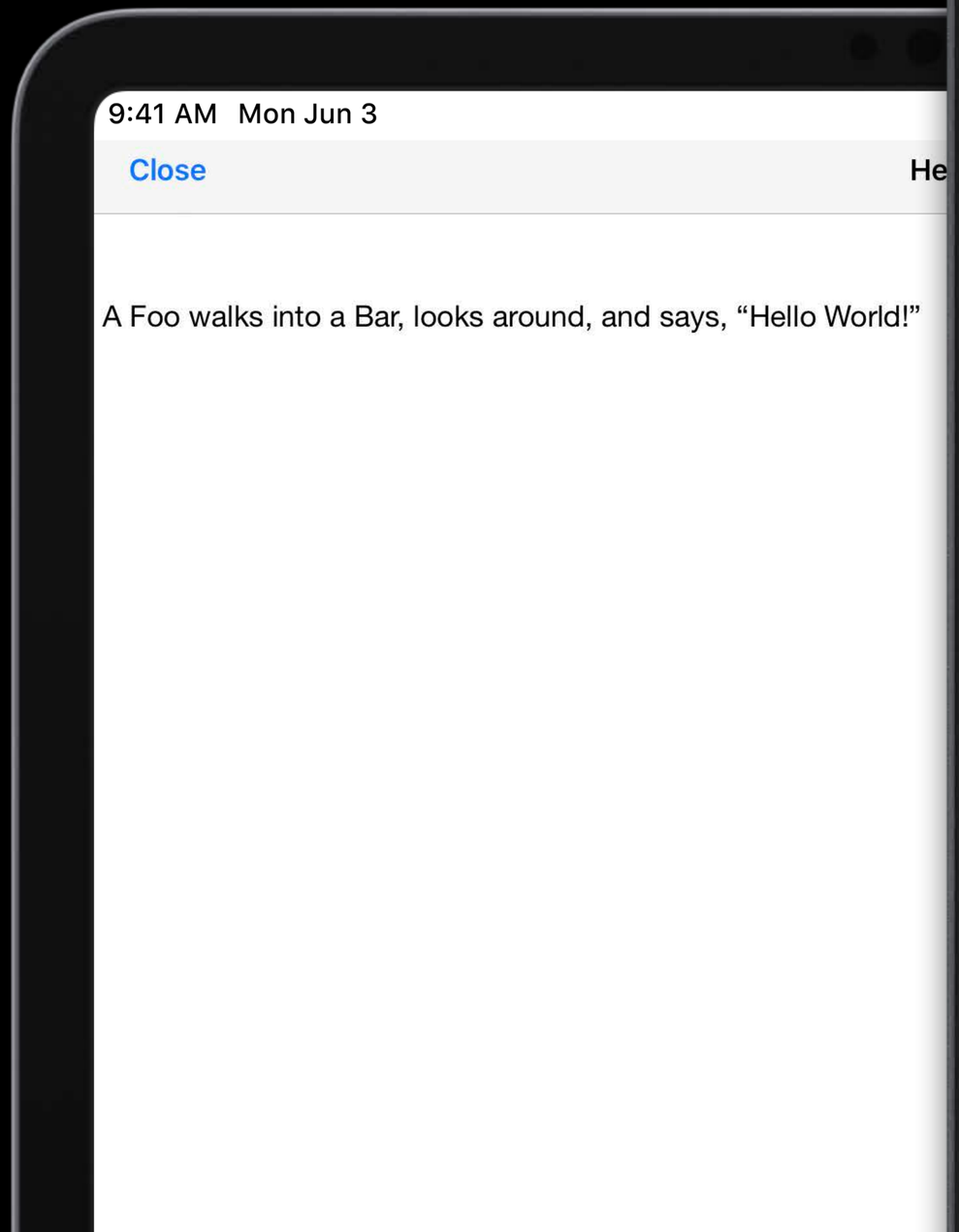
Text Scaling

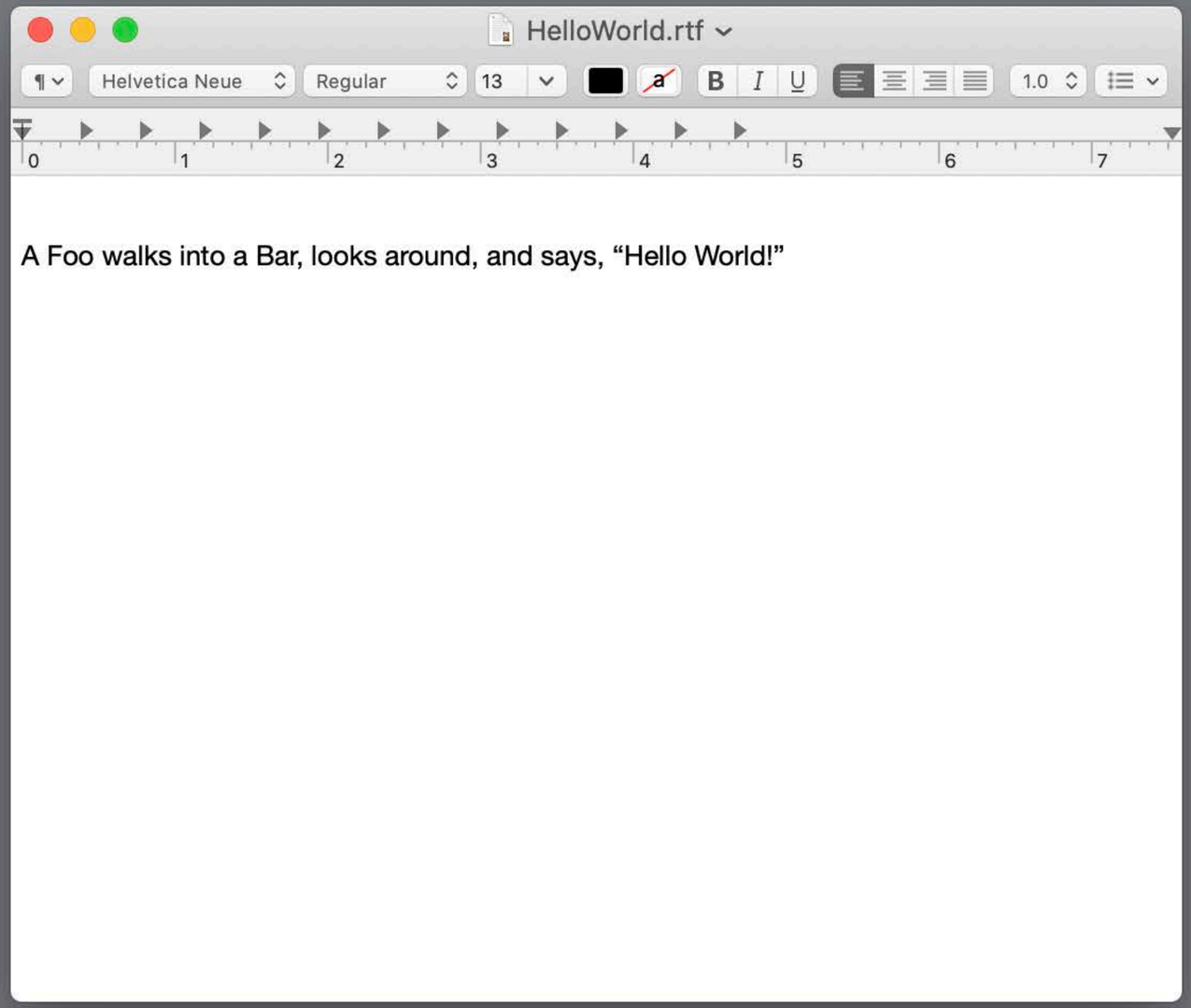
Visual consistency impacts cross-platform user experience

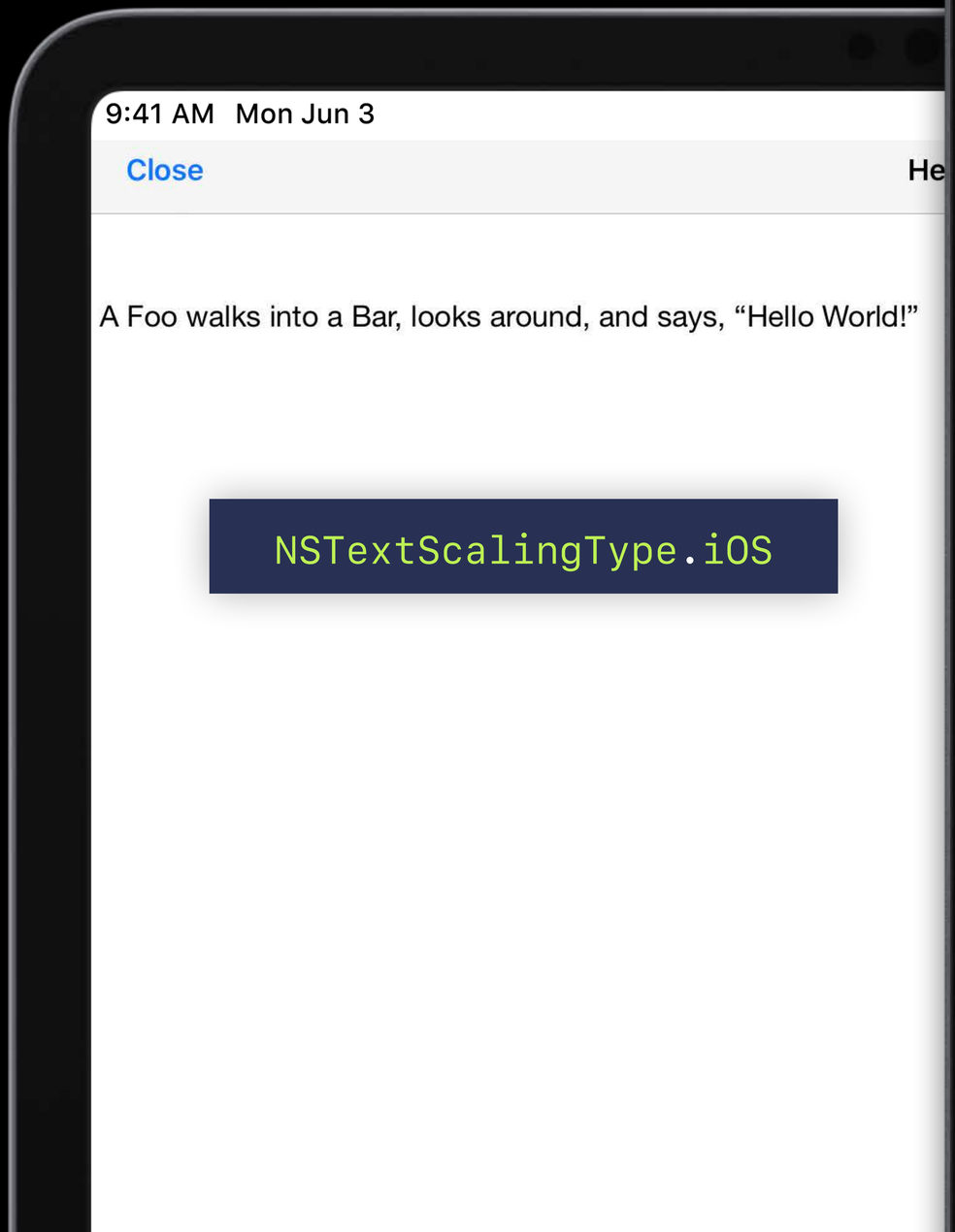
iPad Apps for Mac NEW

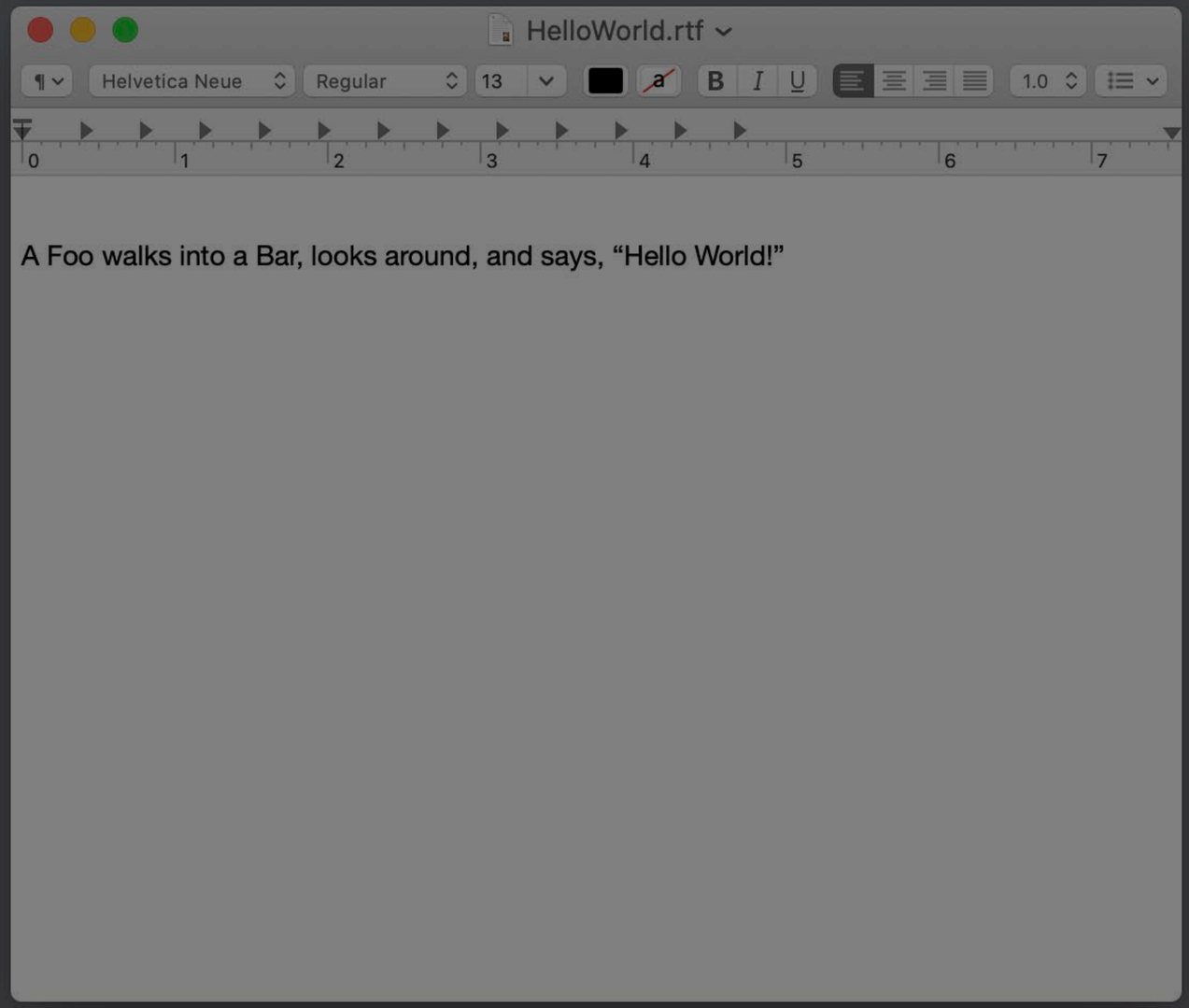
Copy and Paste

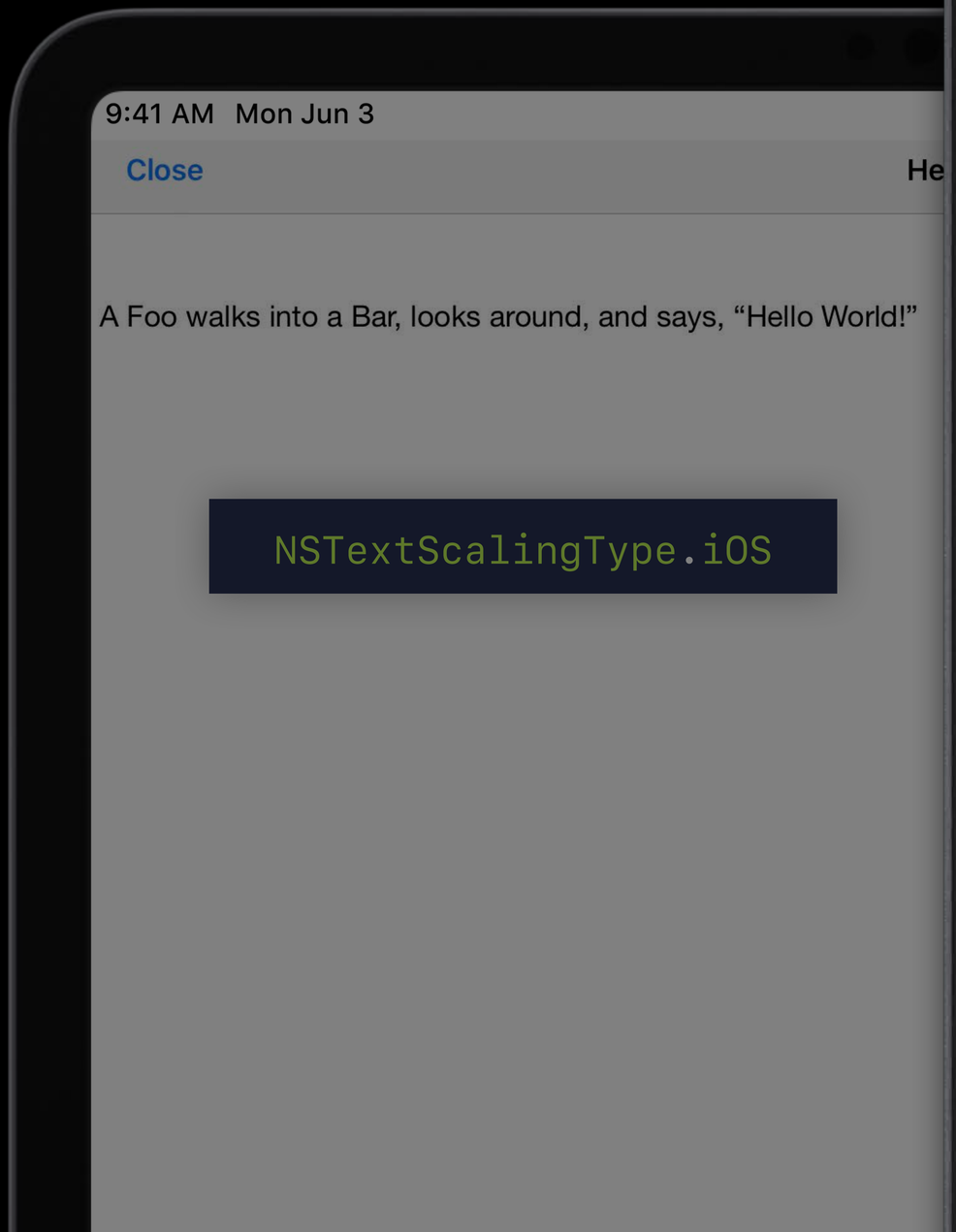
Document interchange

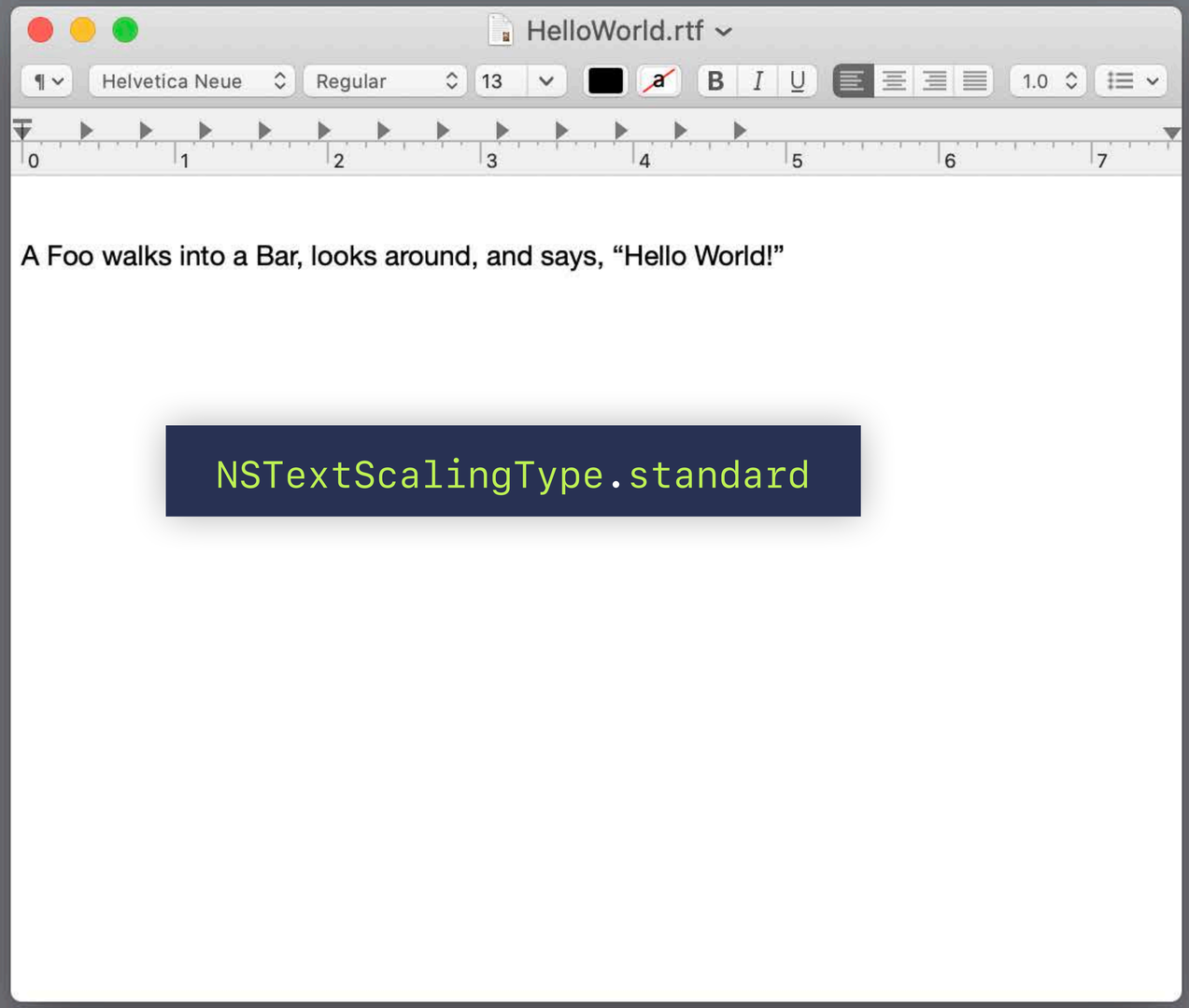

9:41 AM Mon Jun 3
Close
He
A Foo walks into a Bar, looks around, and says, “Hello World!”


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!”


9:41 AM Mon Jun 3
Close
He
A Foo walks into a Bar, looks around, and says, “Hello World!”
NSTextScalingType.iOS


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!”

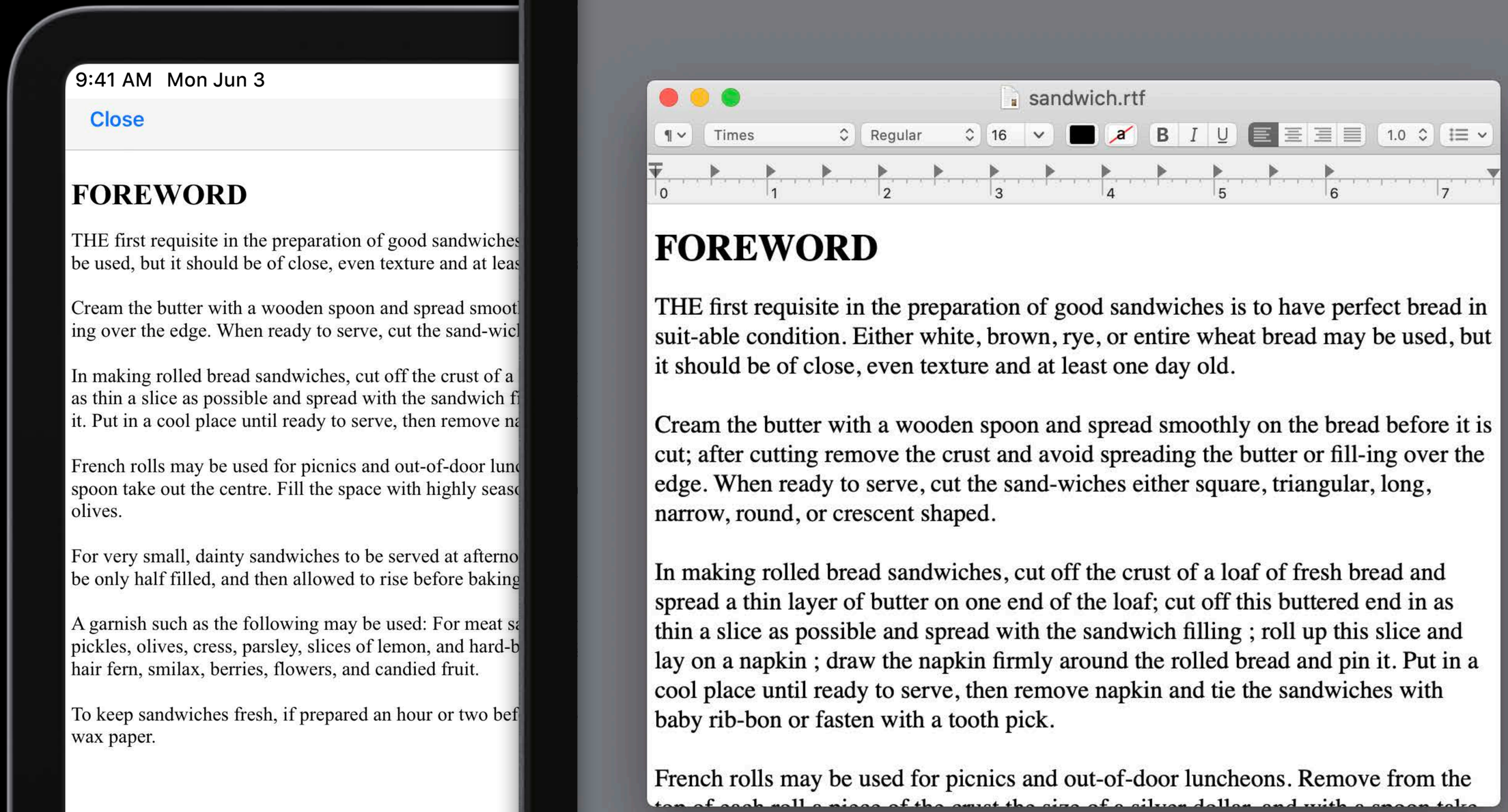

9:41 AM Mon Jun 3
Close
He
A Foo walks into a Bar, looks around, and says, “Hello World!”
NSTextScalingType.iOS


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!”
NSTextScalingType.standard

Use standard text scaling for best cross-platform user experience

Situation

iPad Apps for Mac


9:41 AM Mon Jun 3
Close
FOREWORD
THE first requisite in the preparation of good sandwiches
be used, but it should be of close, even texture and at leas
Cream the butter with a wooden spoon and spread smoot
ing over the edge. When ready to serve, cut the sand-wicl
In making rolled bread sandwiches, cut off the crust of a
as thin a slice as possible and spread with the sandwich fi
it. Put in a cool place until ready to serve, then remove na
French rolls may be used for picnics and out-of-door lunc
spoon take out the centre. Fill the space with highly seas
olives.
For very small, dainty sandwiches to be served at afterno
be only half filled, and then allowed to rise before baking
A garnish such as the following may be used: For meat sa
pickles, olives, cress, parsley, slices of lemon, and hard-b
hair fern, smilax, berries, flowers, and candied fruit.
To keep sandwiches fresh, if prepared an hour or two bef
wax paper.
sandwich.rtf
Times
Regular
16
1.0
0
1
2
3
4
5
6
7
FOREWORD
THE first requisite in the preparation of good sandwiches is to have perfect bread in suit-able condition. Either white, brown, rye, or entire wheat bread may be used, but it should be of close, even texture and at least one day old.
Cream the butter with a wooden spoon and spread smoothly on the bread before it is cut; after cutting remove the crust and avoid spreading the butter or fill-ing over the edge. When ready to serve, cut the sand-wiches either square, triangular, long, narrow, round, or crescent shaped.
In making rolled bread sandwiches, cut off the crust of a loaf of fresh bread and spread a thin layer of butter on one end of the loaf; cut off this buttered end in as thin a slice as possible and spread with the sandwich filling ; roll up this slice and lay on a napkin ; draw the napkin firmly around the rolled bread and pin it. Put in a cool place until ready to serve, then remove napkin and tie the sandwiches with baby rib-bon or fasten with a tooth pick.
French rolls may be used for picnics and out-of-door luncheons. Remove from the

Text Scaling

Situation — iPad Apps for Mac

NEW

Adjust UITextView rendering to match standard text scaling

```
// New UITextView property usesStandardTextScaling
let myTextView = UITextView(frame: myRect, textContainer: myTextContainer)
myTextView.usesStandardTextScaling = true
```

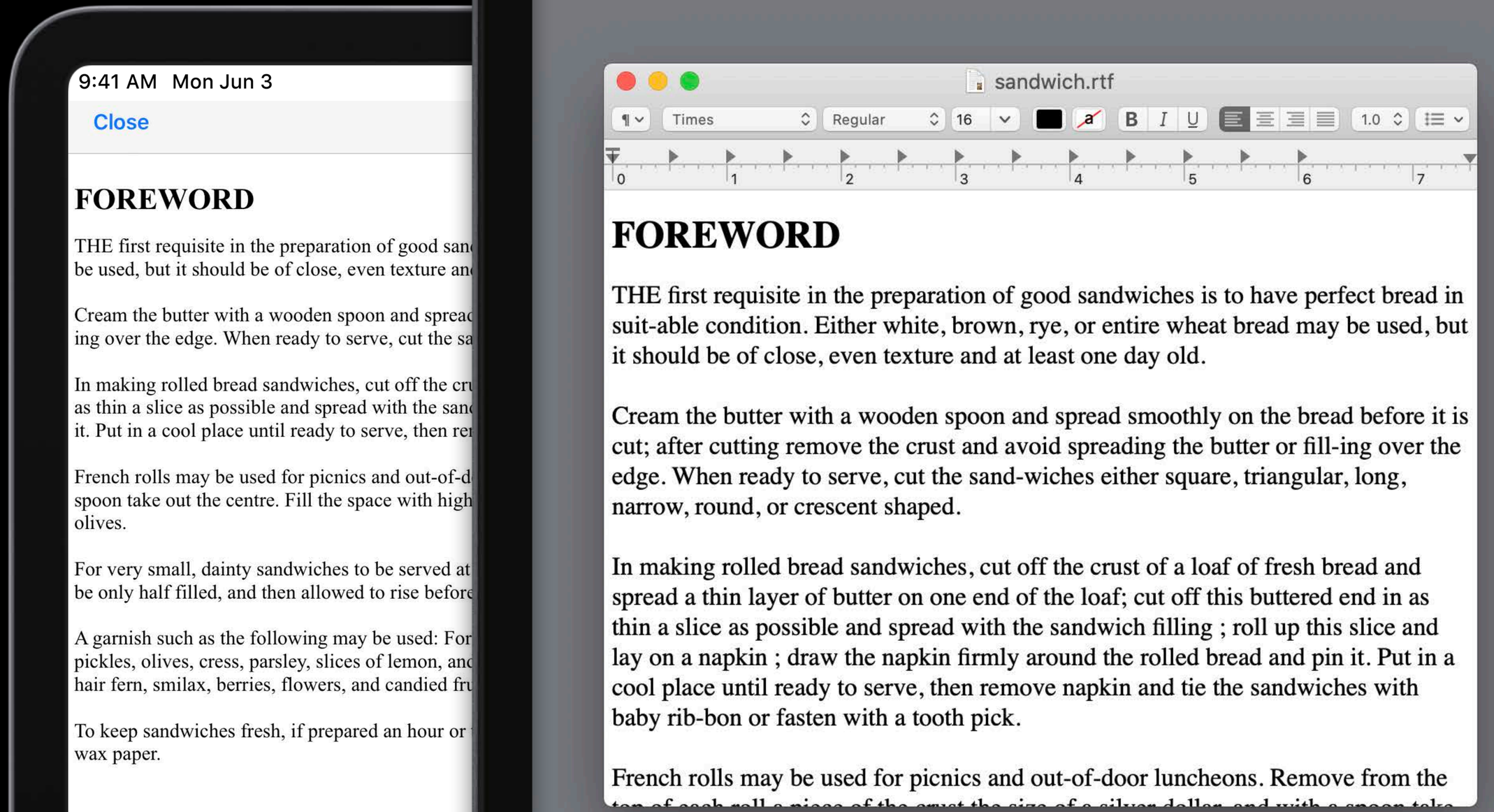
Text Scaling

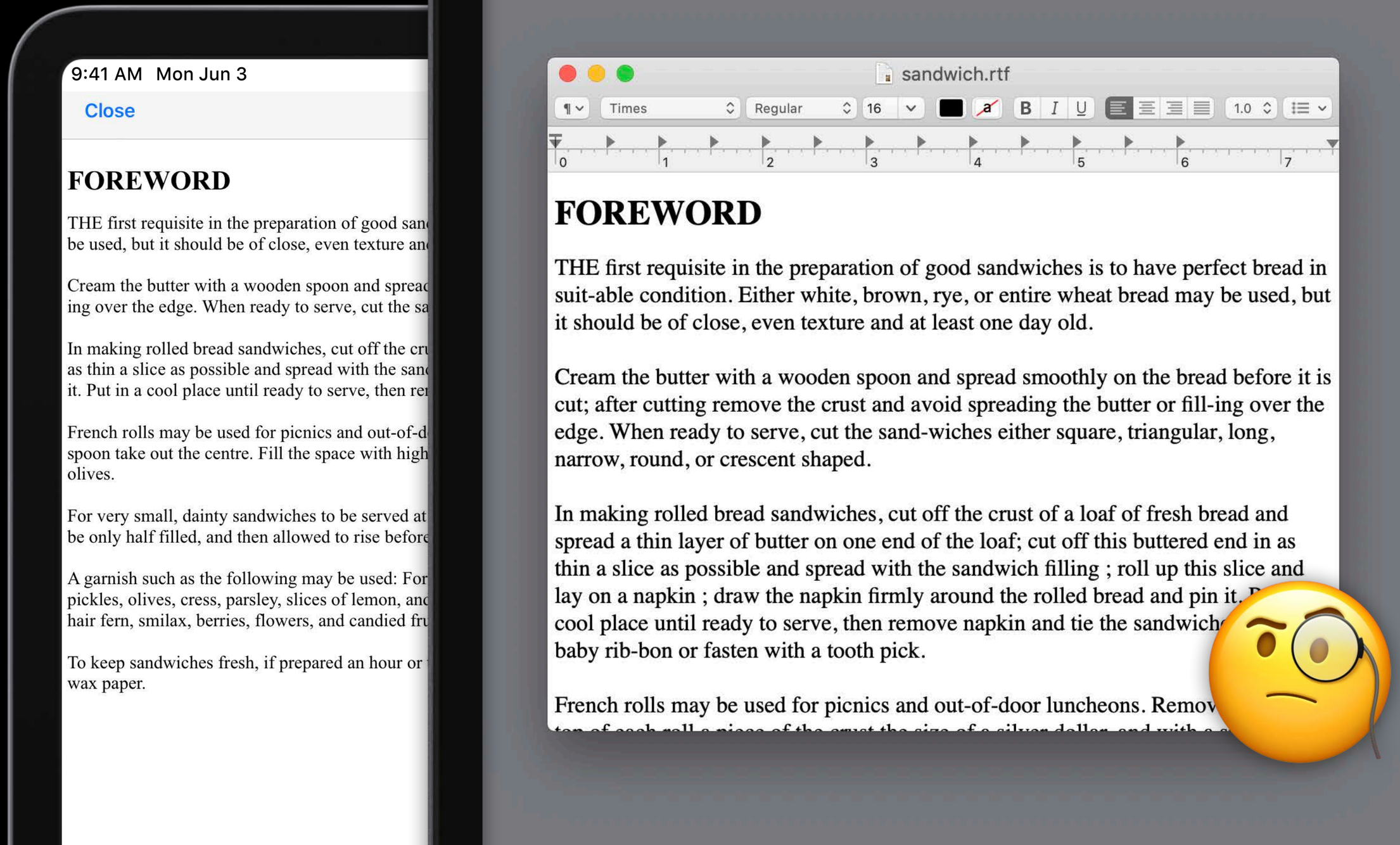
Situation — iPad Apps for Mac

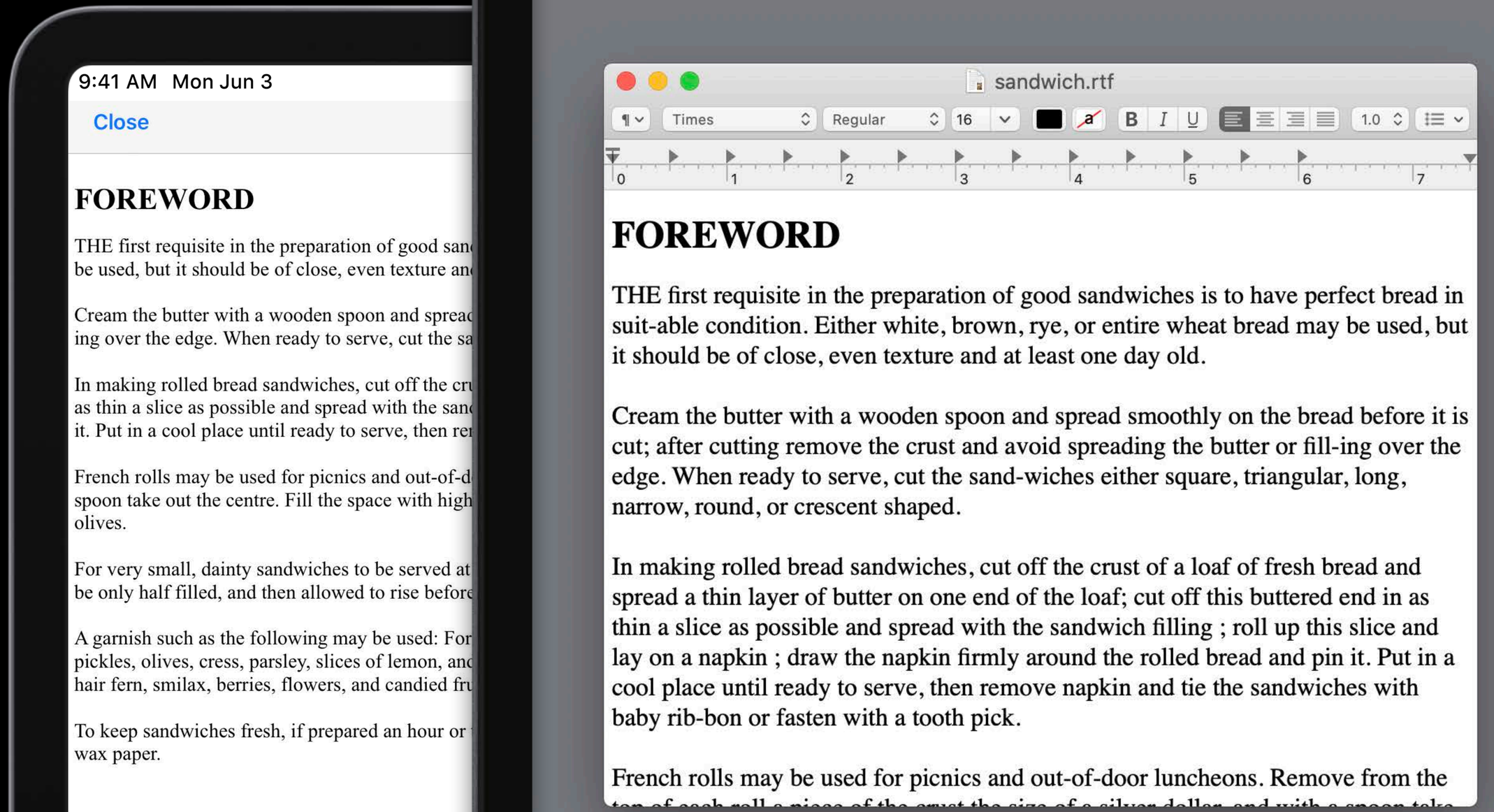
NEW

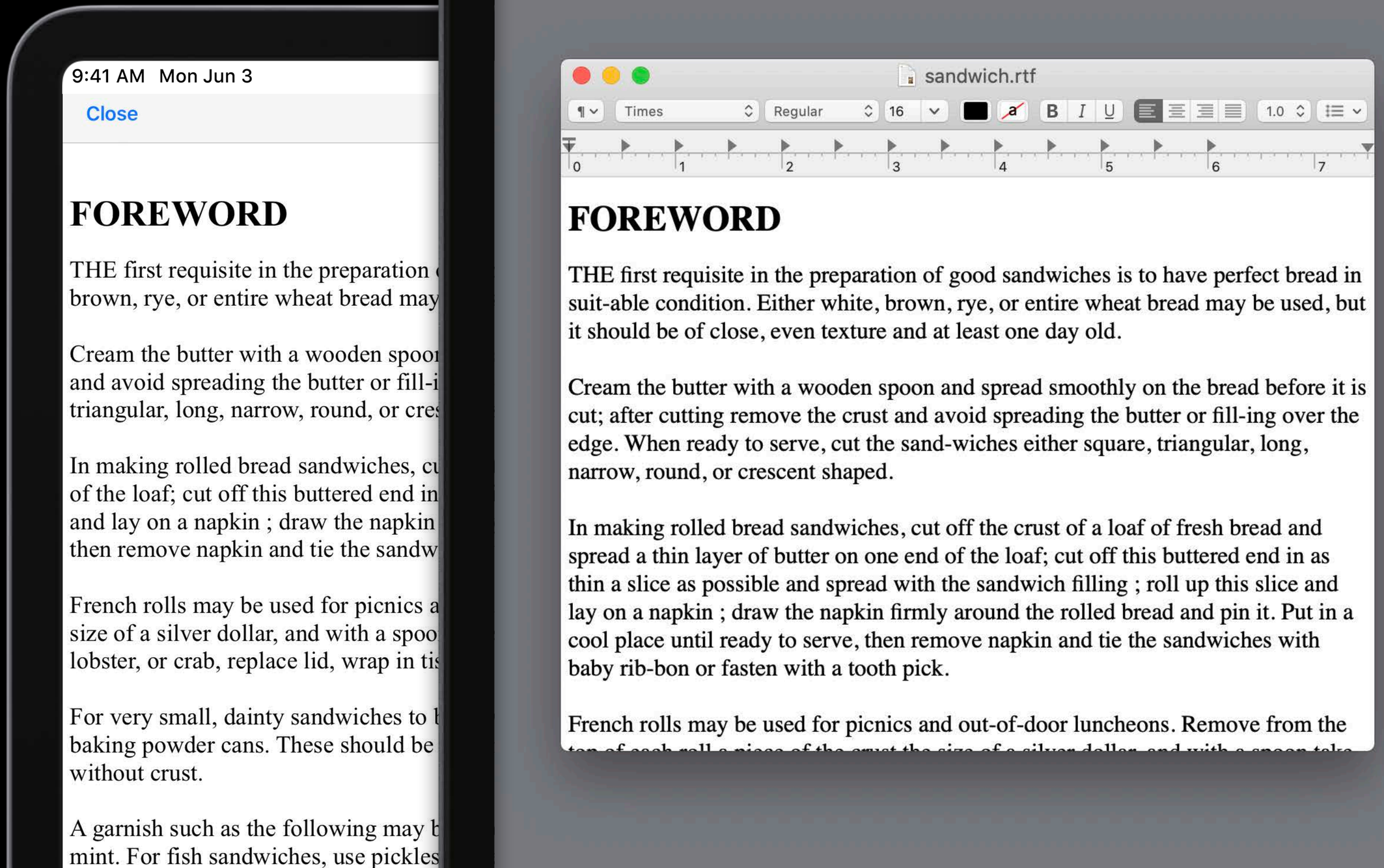
Adjust UITextView rendering to match standard text scaling

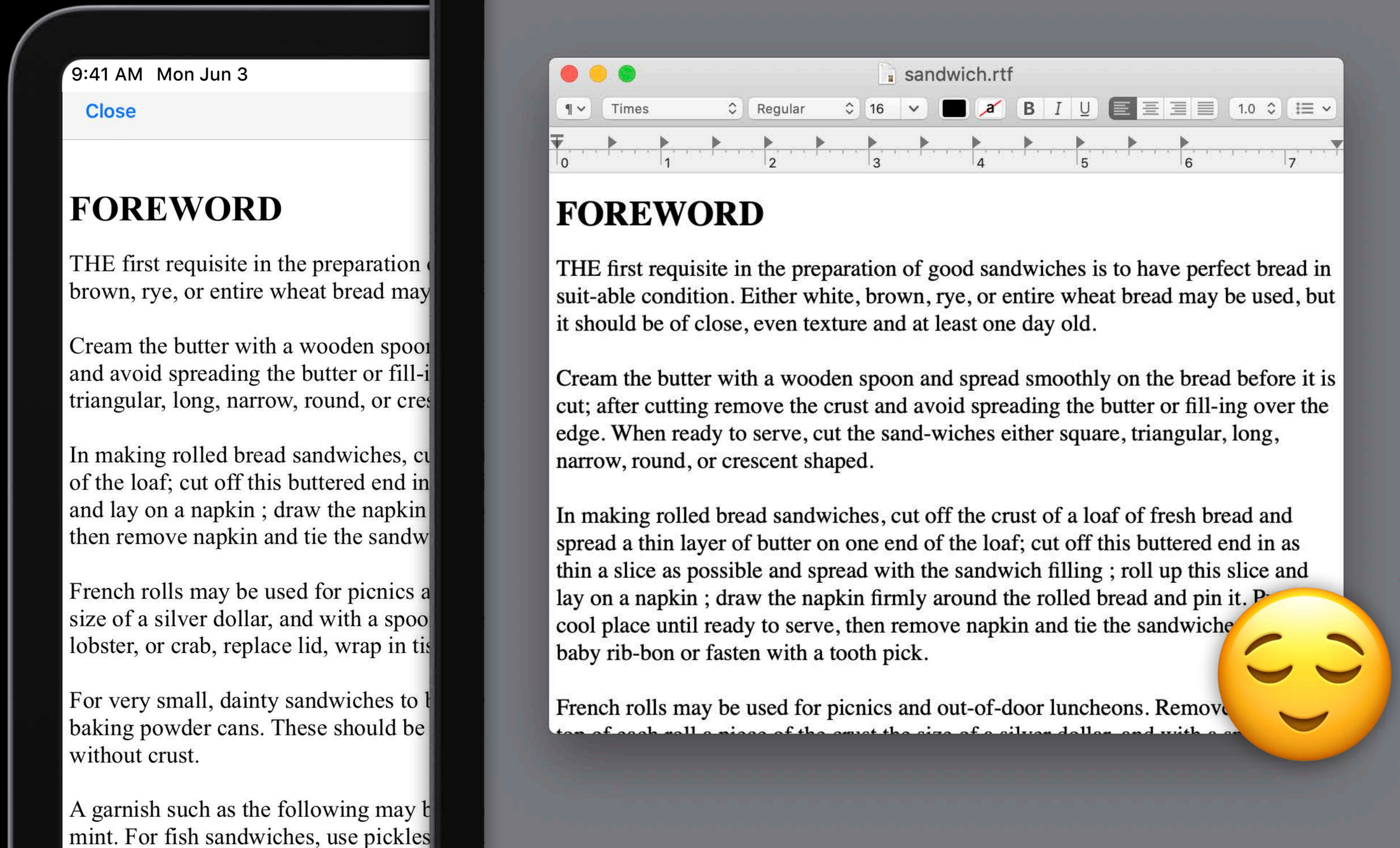
```
// New UITextView property usesStandardTextScaling
let myTextView = UITextView(frame: myRect, textContainer: myTextContainer)
myTextView.usesStandardTextScaling = true
```


9:41 AM Mon Jun 3
Close
FOREWORD
THE first requisite in the preparation of good san
be used, but it should be of close, even texture an
Cream the butter with a wooden spoon and spread
ing over the edge. When ready to serve, cut the sa
In making rolled bread sandwiches, cut off the cru
as thin a slice as possible and spread with the san
it. Put in a cool place until ready to serve, then re
French rolls may be used for picnics and out-of-d
spoon take out the centre. Fill the space with high
olives.
For very small, dainty sandwiches to be served at
be only half filled, and then allowed to rise before
A garnish such as the following may be used: For
pickles, olives, cress, parsley, slices of lemon, and
hair fern, smilax, berries, flowers, and candied fru
To keep sandwiches fresh, if prepared an hour or
wax paper.
sandwich.rtf
Times
Regular
16
1.0
0
1
2
3
4
5
6
7
FOREWORD
THE first requisite in the preparation of good sandwiches is to have perfect bread in suit-able condition. Either white, brown, rye, or entire wheat bread may be used, but it should be of close, even texture and at least one day old.
Cream the butter with a wooden spoon and spread smoothly on the bread before it is cut; after cutting remove the crust and avoid spreading the butter or fill-ing over the edge. When ready to serve, cut the sand-wiches either square, triangular, long, narrow, round, or crescent shaped.
In making rolled bread sandwiches, cut off the crust of a loaf of fresh bread and spread a thin layer of butter on one end of the loaf; cut off this buttered end in as thin a slice as possible and spread with the sandwich filling ; roll up this slice and lay on a napkin ; draw the napkin firmly around the rolled bread and pin it. Put in a cool place until ready to serve, then remove napkin and tie the sandwiches with baby rib-bon or fasten with a tooth pick.
French rolls may be used for picnics and out-of-door luncheons. Remove from the


9:41 AM Mon Jun 3
Close
FOREWORD
THE first requisite in the preparation of good san
be used, but it should be of close, even texture an
Cream the butter with a wooden spoon and spread
ing over the edge. When ready to serve, cut the sa
In making rolled bread sandwiches, cut off the cr
as thin a slice as possible and spread with the san
it. Put in a cool place until ready to serve, then re
French rolls may be used for picnics and out-of-d
spoon take out the centre. Fill the space with high
olives.
For very small, dainty sandwiches to be served at
be only half filled, and then allowed to rise before
A garnish such as the following may be used: For
pickles, olives, cress, parsley, slices of lemon, an
hair fern, smilax, berries, flowers, and candied fr
To keep sandwiches fresh, if prepared an hour or
wax paper.
sandwich.rtf
Times
Regular
16
1.0
0
1
2
3
4
5
6
7
FOREWORD
THE first requisite in the preparation of good sandwiches is to have perfect bread in suit-able condition. Either white, brown, rye, or entire wheat bread may be used, but it should be of close, even texture and at least one day old.
Cream the butter with a wooden spoon and spread smoothly on the bread before it is cut; after cutting remove the crust and avoid spreading the butter or fill-ing over the edge. When ready to serve, cut the sand-wiches either square, triangular, long, narrow, round, or crescent shaped.
In making rolled bread sandwiches, cut off the crust of a loaf of fresh bread and spread a thin layer of butter on one end of the loaf; cut off this buttered end in as thin a slice as possible and spread with the sandwich filling ; roll up this slice and lay on a napkin ; draw the napkin firmly around the rolled bread and pin it.
cool place until ready to serve, then remove napkin and tie the sandwich
baby rib-bon or fasten with a tooth pick.
French rolls may be used for picnics and out-of-door luncheons. Remov


9:41 AM Mon Jun 3
Close
FOREWORD
THE first requisite in the preparation of good san
be used, but it should be of close, even texture an
Cream the butter with a wooden spoon and spread
ing over the edge. When ready to serve, cut the sa
In making rolled bread sandwiches, cut off the cru
as thin a slice as possible and spread with the san
it. Put in a cool place until ready to serve, then re
French rolls may be used for picnics and out-of-d
spoon take out the centre. Fill the space with high
olives.
For very small, dainty sandwiches to be served at
be only half filled, and then allowed to rise before
A garnish such as the following may be used: For
pickles, olives, cress, parsley, slices of lemon, and
hair fern, smilax, berries, flowers, and candied fru
To keep sandwiches fresh, if prepared an hour or
wax paper.
sandwich.rtf
Times
Regular
16
1.0
0
1
2
3
4
5
6
7
FOREWORD
THE first requisite in the preparation of good sandwiches is to have perfect bread in suit-able condition. Either white, brown, rye, or entire wheat bread may be used, but it should be of close, even texture and at least one day old.
Cream the butter with a wooden spoon and spread smoothly on the bread before it is cut; after cutting remove the crust and avoid spreading the butter or fill-ing over the edge. When ready to serve, cut the sand-wiches either square, triangular, long, narrow, round, or crescent shaped.
In making rolled bread sandwiches, cut off the crust of a loaf of fresh bread and spread a thin layer of butter on one end of the loaf; cut off this buttered end in as thin a slice as possible and spread with the sandwich filling ; roll up this slice and lay on a napkin ; draw the napkin firmly around the rolled bread and pin it. Put in a cool place until ready to serve, then remove napkin and tie the sandwiches with baby rib-bon or fasten with a tooth pick.
French rolls may be used for picnics and out-of-door luncheons. Remove from the


9:41 AM Mon Jun 3
Close
FOREWORD
THE first requisite in the preparation
brown, rye, or entire wheat bread may
Cream the butter with a wooden spoo
and avoid spreading the butter or fill-i
triangular, long, narrow, round, or cre
In making rolled bread sandwiches, cu
of the loaf; cut off this buttered end in
and lay on a napkin ; draw the napkin
then remove napkin and tie the sandw
French rolls may be used for picnics a
size of a silver dollar, and with a spoo
lobster, or crab, replace lid, wrap in tis
For very small, dainty sandwiches to b
baking powder cans. These should be
without crust.
A garnish such as the following may b
mint. For fish sandwiches, use pickles
sandwich.rtf
Times
Regular
16
1.0
0
1
2
3
4
5
6
7
FOREWORD
THE first requisite in the preparation of good sandwiches is to have perfect bread in suit-able condition. Either white, brown, rye, or entire wheat bread may be used, but it should be of close, even texture and at least one day old.
Cream the butter with a wooden spoon and spread smoothly on the bread before it is cut; after cutting remove the crust and avoid spreading the butter or fill-ing over the edge. When ready to serve, cut the sand-wiches either square, triangular, long, narrow, round, or crescent shaped.
In making rolled bread sandwiches, cut off the crust of a loaf of fresh bread and spread a thin layer of butter on one end of the loaf; cut off this buttered end in as thin a slice as possible and spread with the sandwich filling ; roll up this slice and lay on a napkin ; draw the napkin firmly around the rolled bread and pin it. Put in a cool place until ready to serve, then remove napkin and tie the sandwiches with baby rib-bon or fasten with a tooth pick.
French rolls may be used for picnics and out-of-door luncheons. Remove from the


9:41 AM Mon Jun 3
Close
FOREWORD
THE first requisite in the preparation
brown, rye, or entire wheat bread may
Cream the butter with a wooden spoo
and avoid spreading the butter or fill-i
triangular, long, narrow, round, or cre
In making rolled bread sandwiches, cu
of the loaf; cut off this buttered end in
and lay on a napkin ; draw the napkin
then remove napkin and tie the sandw
French rolls may be used for picnics a
size of a silver dollar, and with a spoo
lobster, or crab, replace lid, wrap in tis
For very small, dainty sandwiches to b
baking powder cans. These should be
without crust.
A garnish such as the following may b
mint. For fish sandwiches, use pickles
sandwich.rtf
Times
Regular
16
1.0
0
1
2
3
4
5
6
7
FOREWORD
THE first requisite in the preparation of good sandwiches is to have perfect bread in suit-able condition. Either white, brown, rye, or entire wheat bread may be used, but it should be of close, even texture and at least one day old.
Cream the butter with a wooden spoon and spread smoothly on the bread before it is cut; after cutting remove the crust and avoid spreading the butter or fill-ing over the edge. When ready to serve, cut the sand-wiches either square, triangular, long, narrow, round, or crescent shaped.
In making rolled bread sandwiches, cut off the crust of a loaf of fresh bread and spread a thin layer of butter on one end of the loaf; cut off this buttered end in as thin a slice as possible and spread with the sandwich filling ; roll up this slice and lay on a napkin ; draw the napkin firmly around the rolled bread and pin it. P
cool place until ready to serve, then remove napkin and tie the sandwiche
baby rib-bon or fasten with a tooth pick.
French rolls may be used for picnics and out-of-door luncheons. Remov

Text Scaling

Situation — iPad Apps for Mac

`usesStandardTextScaling = false` by default

Custom view transforms may give unexpected results

Situation

Copy and Paste

Text Scaling

Situation — Copy and Paste

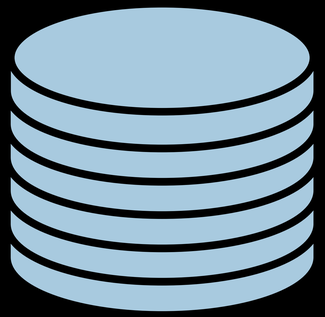
NEW

Visually consistent copy and paste happens automatically

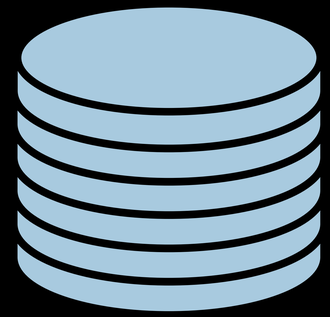
iOS

macOS

iOS

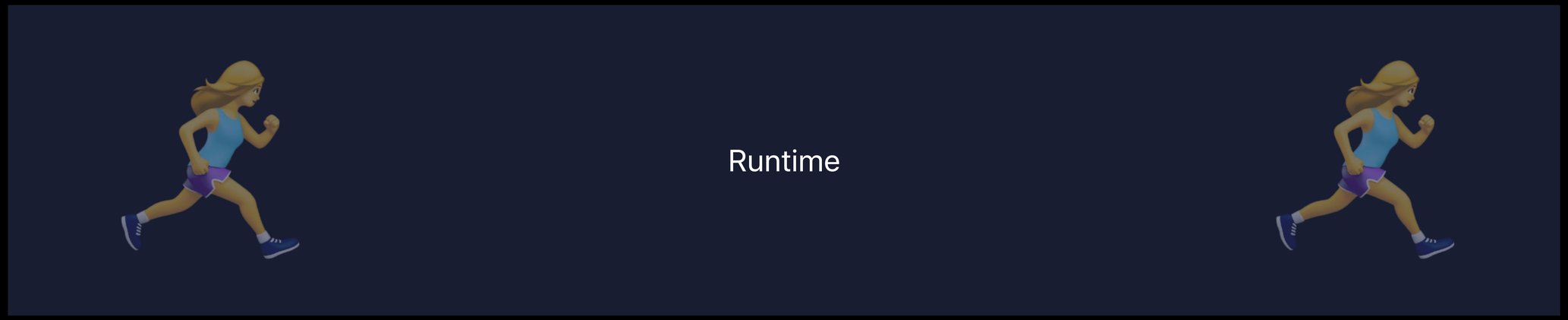


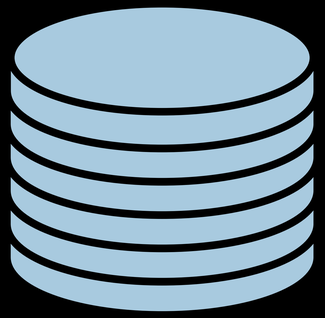
macOS

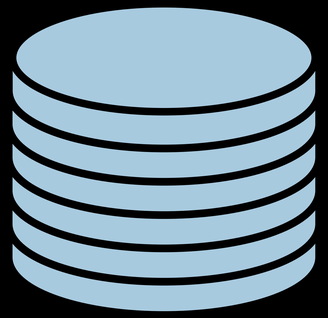


iOS

macOS







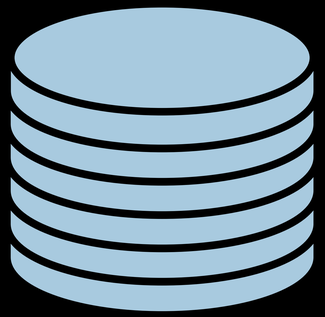
iOS

macOS

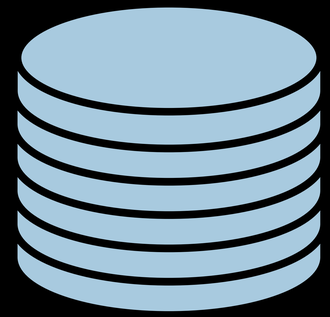
Runtime

Persistence

iOS



macOS

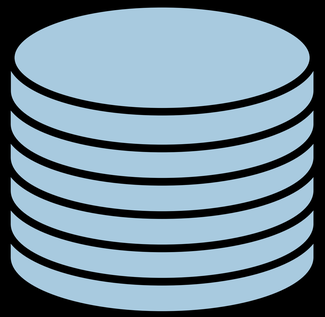


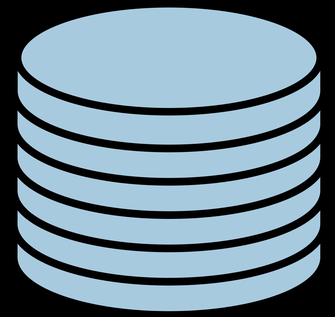
iOS

macOS

Copy

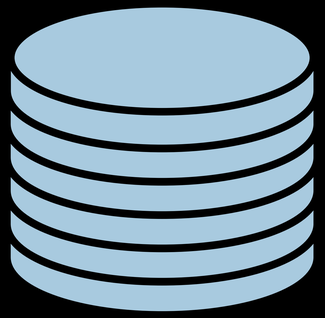
Simulacrum



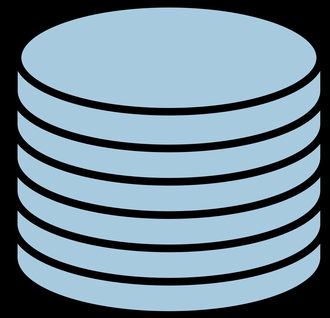


iOS

macOS



```
{\rtf1\ansi\ansicpg1252\cocoartf2467
{\fonttbl\f0\fnil\fcharset0
 AvenirNext-Heavy;}…
```



Text Scaling

Situation — Copy and Paste

NSAttributedString API for writing RTF adds metadata for text scaling NEW BEHAVIOR

```
func data(from: NSRange,
          documentAttributes: [NSAttributedString.DocumentAttributeKey : Any]) throws -> Data
```

iOS

macOS

Simulacrum

iOS

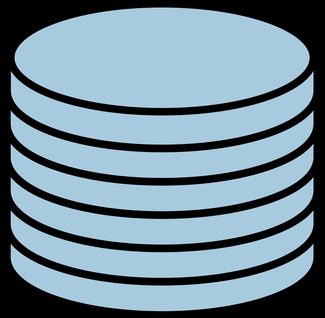
macOS

Copy

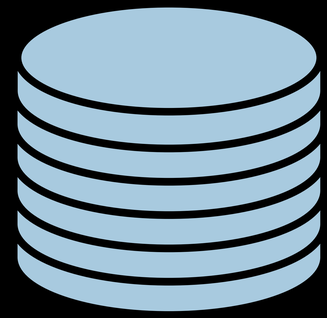
Simulacrum

iOS

macOS



```
{\rtf1\ansi\ansicpg1252\cocoartf2467
\cocoatextscaling1\cocoaplatform1
{\fonttbl\f0\fnil\fcharset0
 AvenirNext-Heavy;}…
```



Text Scaling

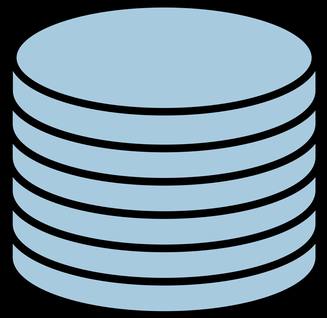
Situation — Copy and Paste

NSAttributedString API for reading RTF adjusts font sizes based on text scaling metadata NEW BEHAVIOR

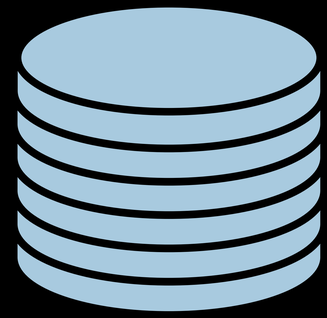
```
init(data: Data,
     options: [NSAttributedString.DocumentReadingOptionKey : Any],
     documentAttributes: AutoreleasingUnsafeMutablePointer<NSDictionary?>?) throws

init(url: URL,
     options: [NSAttributedString.DocumentReadingOptionKey : Any],
     documentAttributes: AutoreleasingUnsafeMutablePointer<NSDictionary?>?) throws
```

iOS

macOS



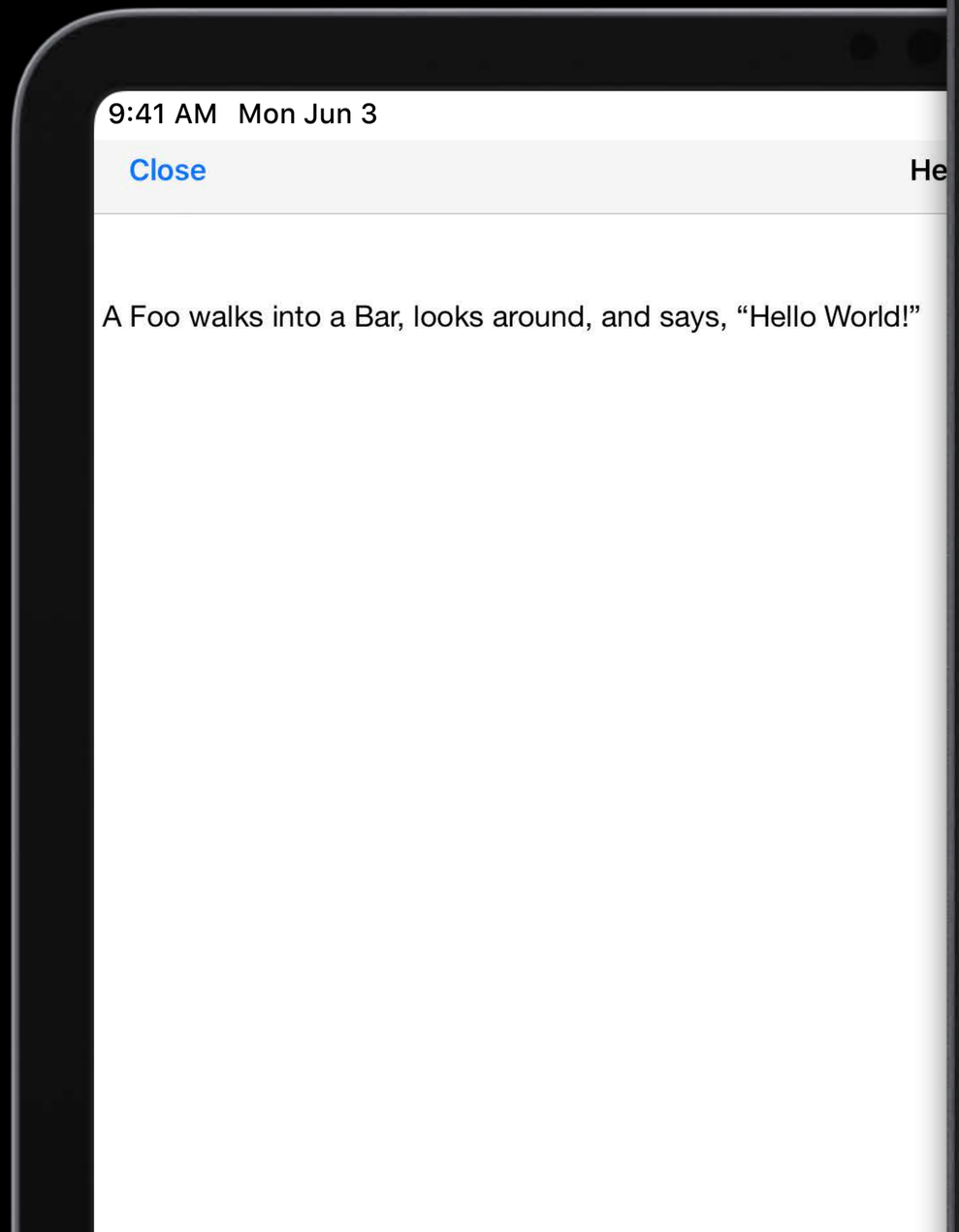
```
{\rtf1\ansi\ansicpg1252\cocoartf2467
\cocoatextscaling1\cocoaplatform1
{\fonttbl\f0\fnil\fcharset0
 AvenirNext-Heavy;}…
```

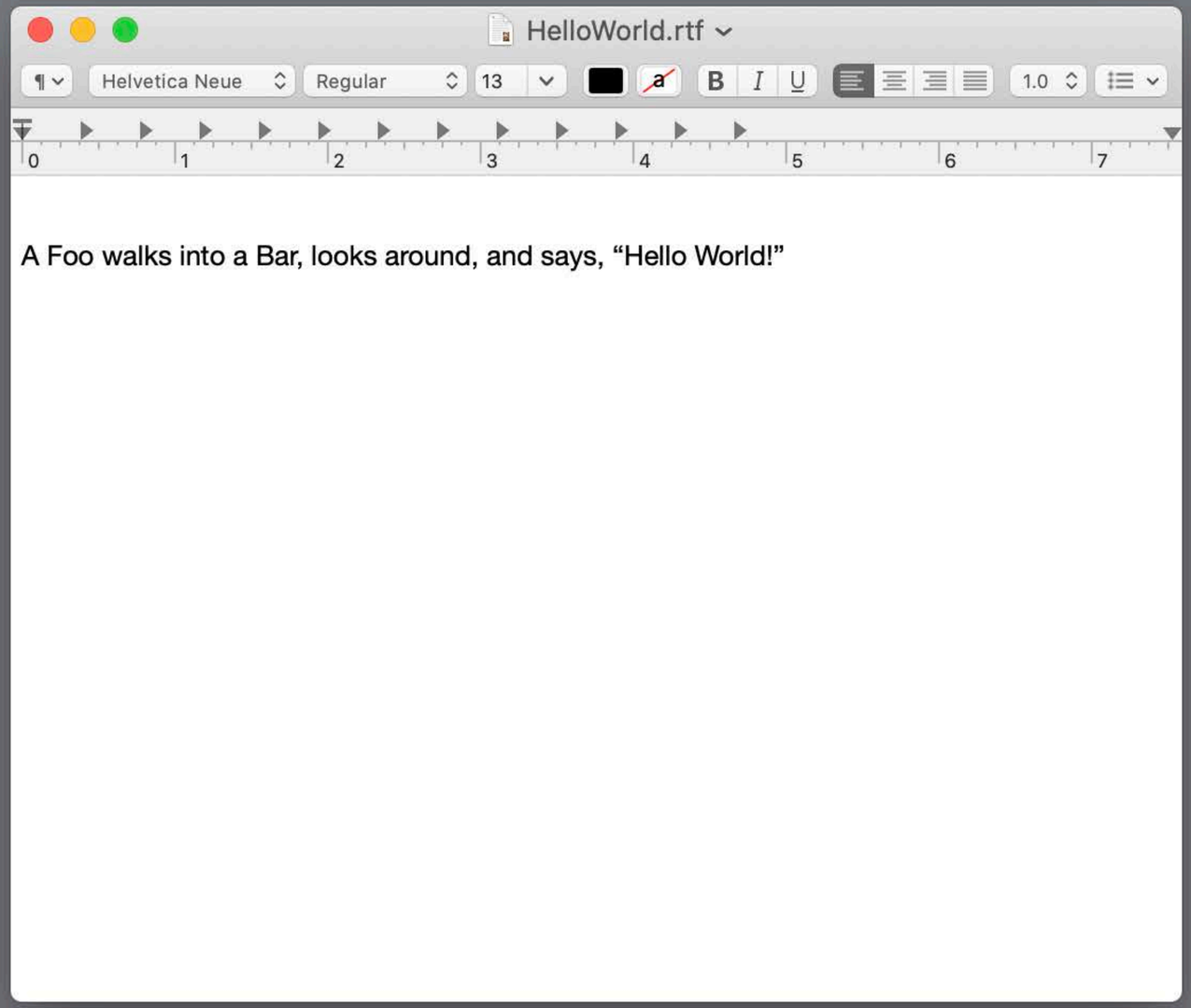


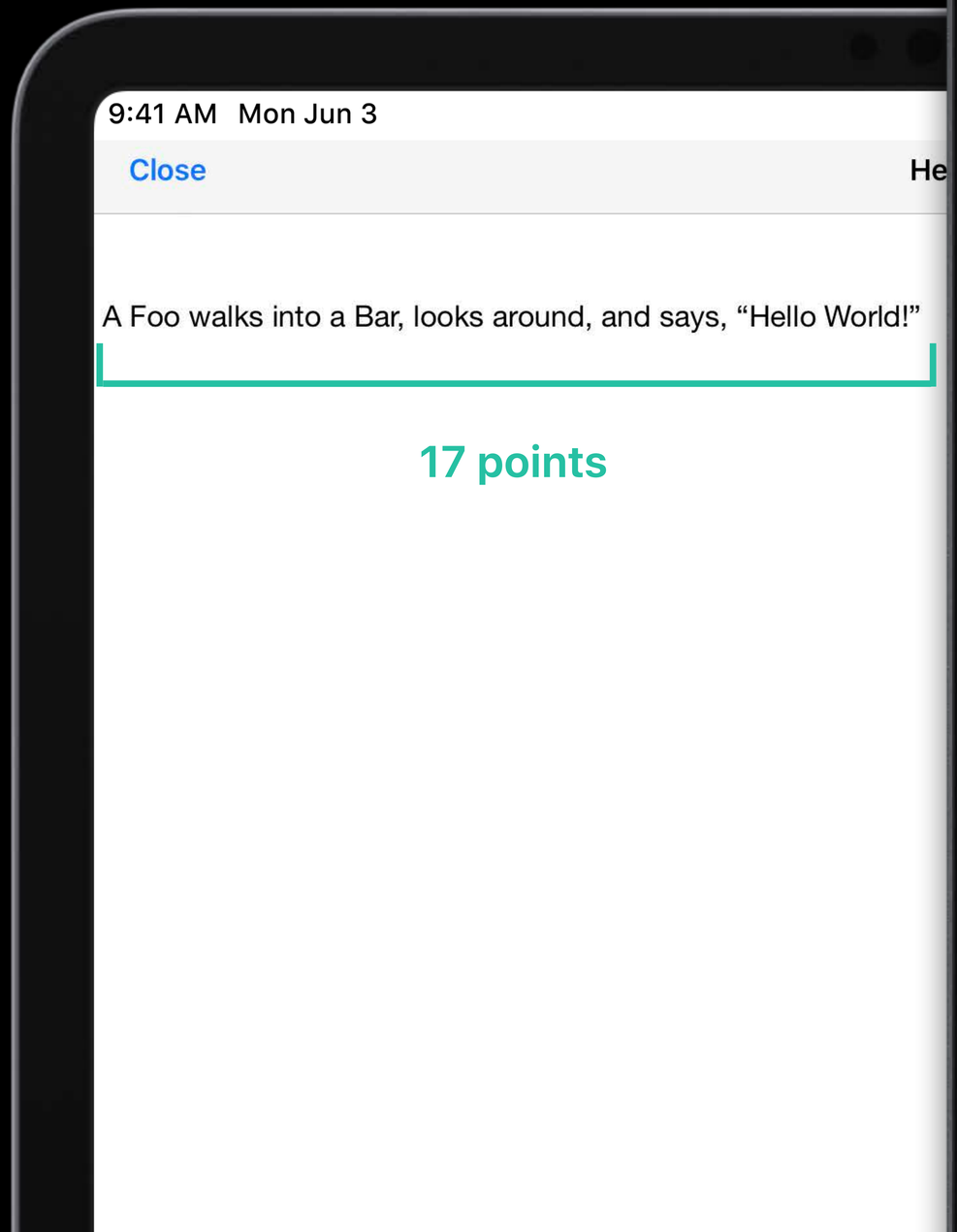
iOS

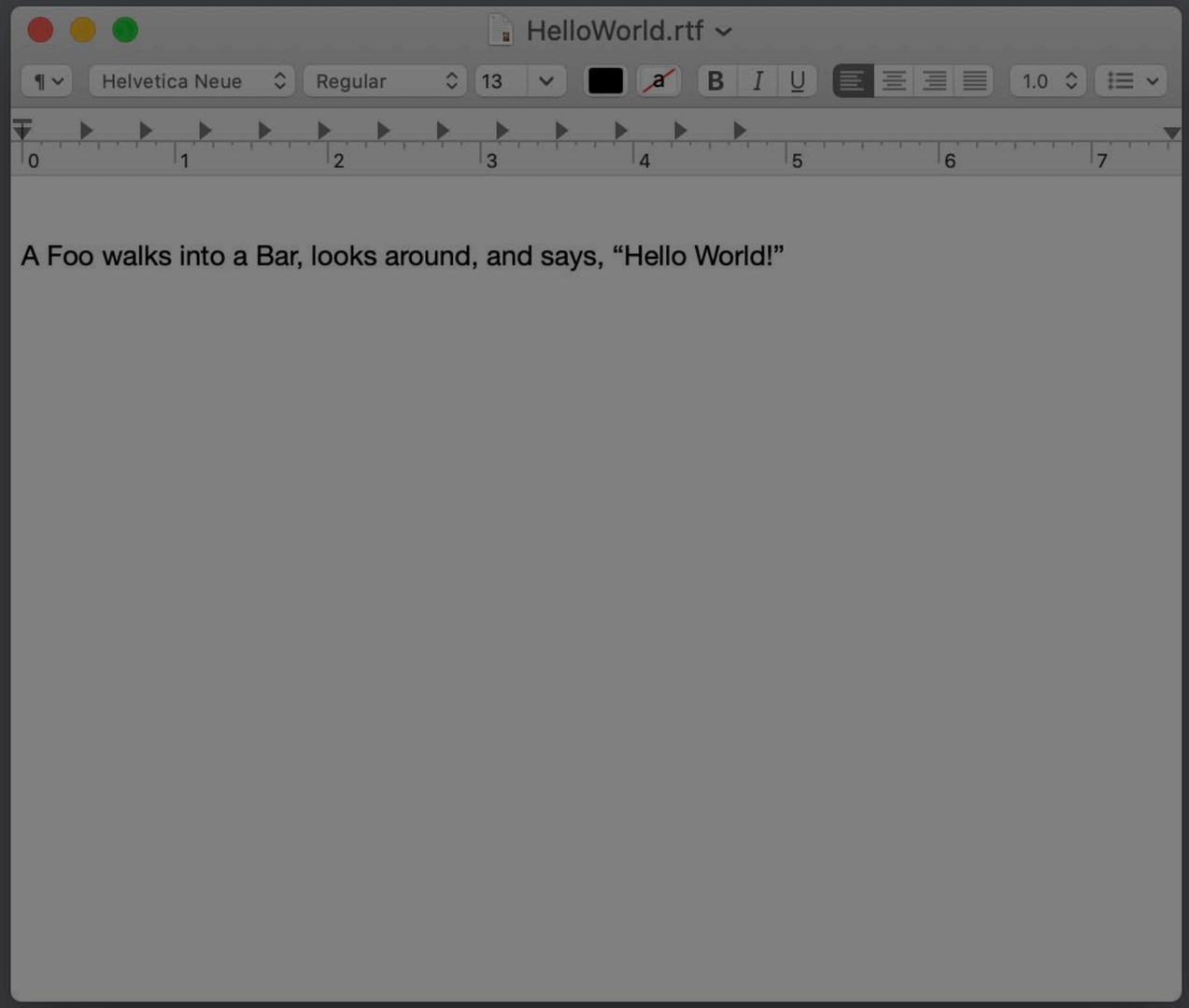
macOS

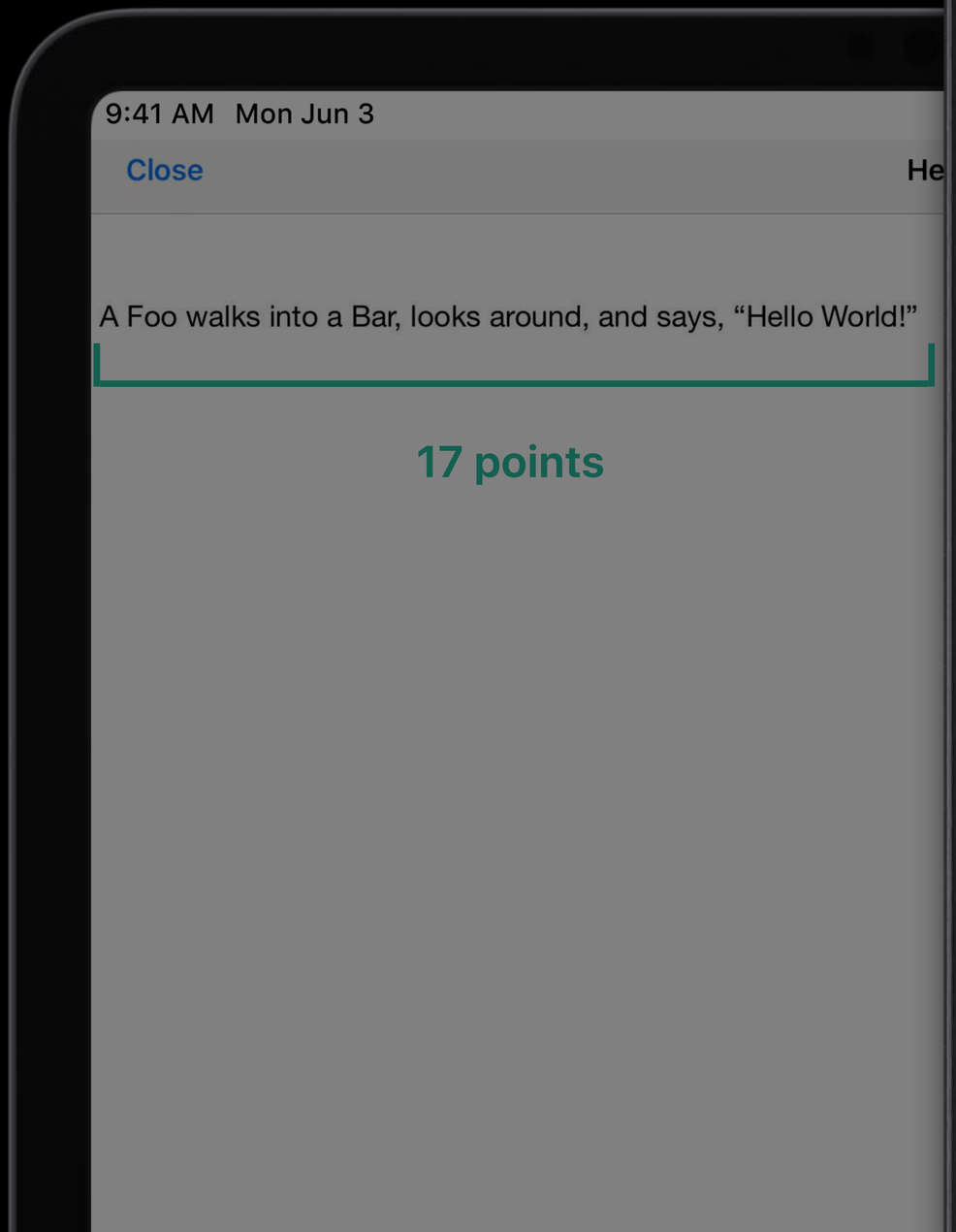
Simulacrum

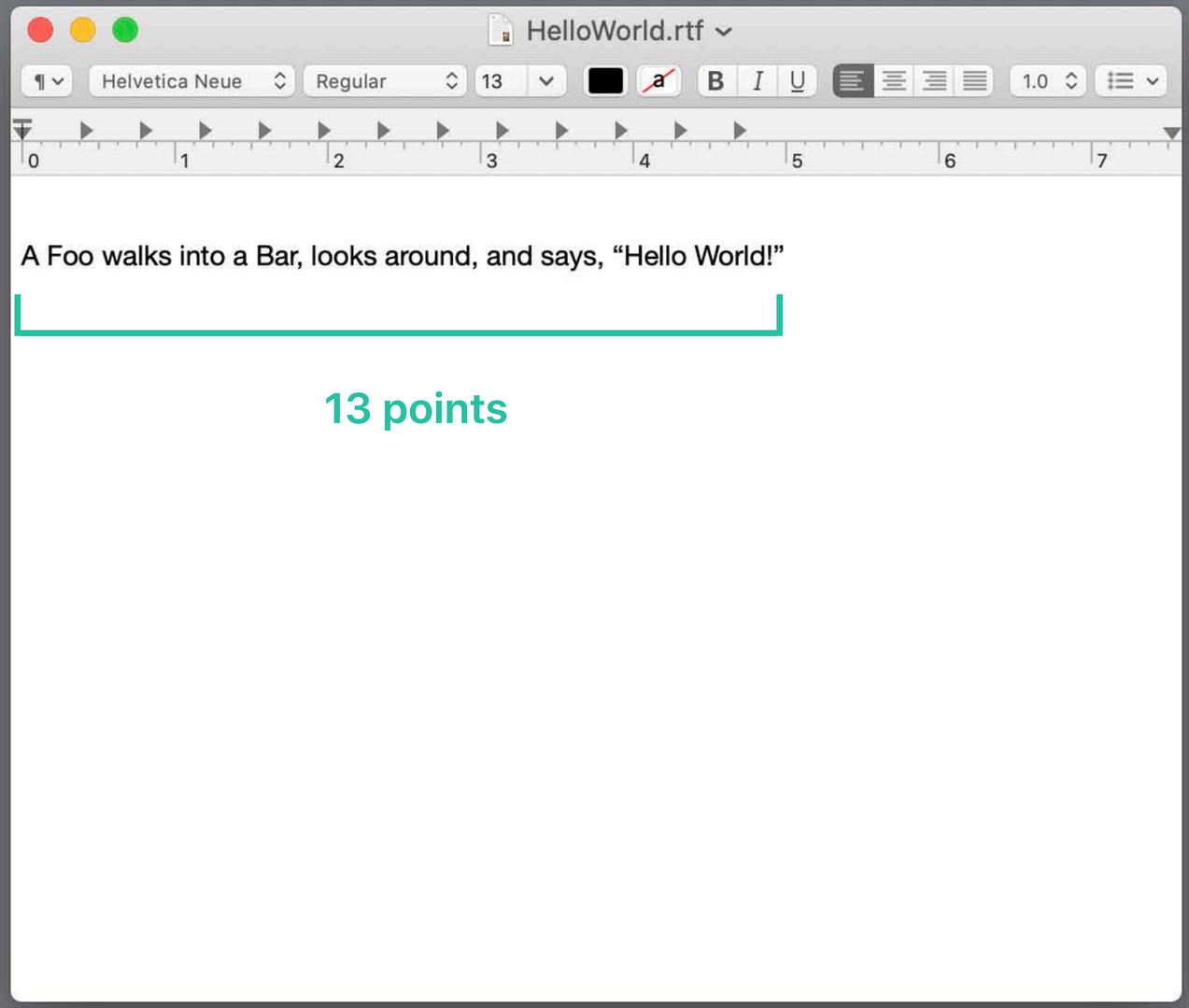

9:41 AM Mon Jun 3
Close
He
A Foo walks into a Bar, looks around, and says, “Hello World!”


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!”


9:41 AM Mon Jun 3
Close
He
A Foo walks into a Bar, looks around, and says, “Hello World!”
17 points


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
A Foo walks into a Bar, looks around, and says, “Hello World!”


9:41 AM Mon Jun 3
Close
He
A Foo walks into a Bar, looks around, and says, “Hello World!”
17 points


HelloWorld.rtf
Helvetica Neue
Regular
13
1.0
0
1
2
3
4
5
6
7
A Foo walks into a Bar, looks around, and says, “Hello World!”
13 points

Text Scaling

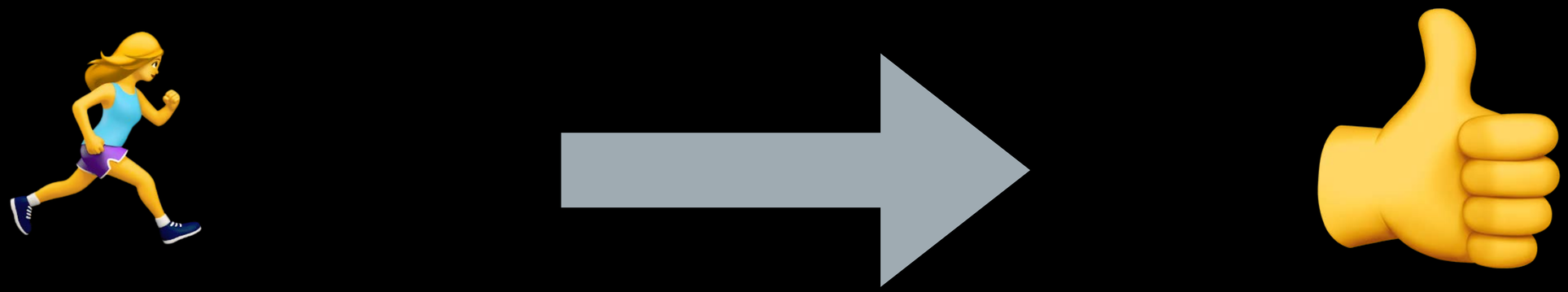
Situation — Copy and Paste

Font size is different depending on text scaling

Text Scaling

Situation — Copy and Paste

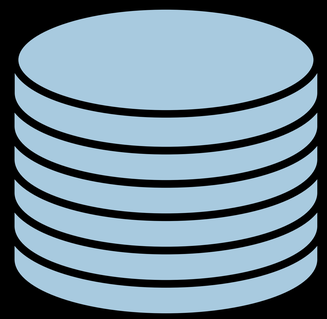
Font size is different depending on text scaling

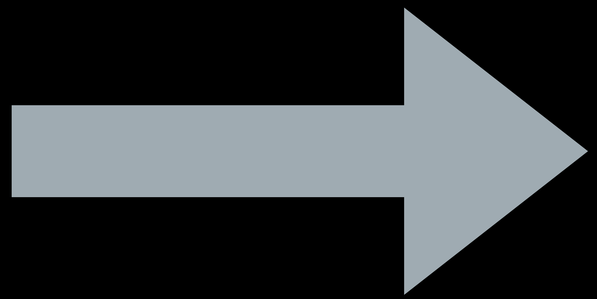


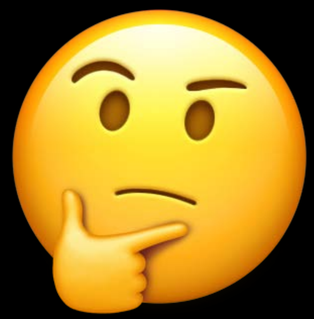
Text Scaling

Situation — Copy and Paste

Font size is different depending on text scaling







Situation

Document interchange

cross-platform
visual consistency

document model
integrity

17 point Helvetica Neue

13 point Helvetica Neue

cross-platform
visual consistency

document model
integrity

17 point Helvetica Neue

13 point Helvetica Neue

13 point Helvetica Neue

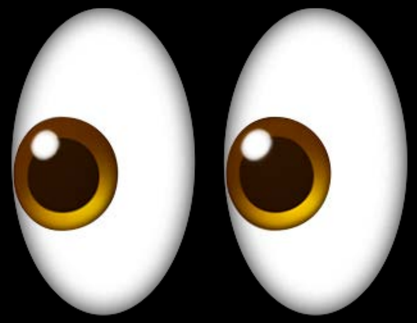
13 point Helvetica Neue

cross-platform
visual consistency

document model
integrity

13 point Helvetica Neue

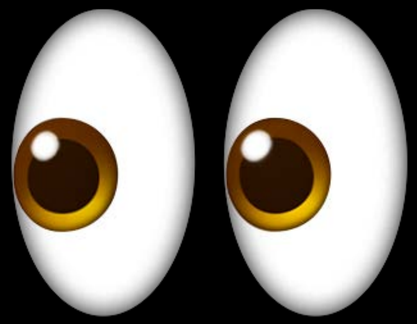
13 point Helvetica Neue



13 point Helvetica Neue

13 point Helvetica Neue

`NSTextScalingType.iOS`



13 point Helvetica Neue

13 point Helvetica Neue

`NSTextScalingType.iOS`

`NSTextScalingType.standard`

Text Scaling

Situation — Document interchange

Two choices

- Use different font sizes for viewing and saving MODEL
- Change rendering scale when viewing VIEW

Case Study

RTF document interchange

Text Scaling

Case study — RTF document interchange

Tag documents with text scaling metadata

```
{\rtf1\ansi\ansicpg1252\cocoartf2467
\cocoatextscaling1\cocoaplatform1
{\fonttbl\f0\fnil\fcharset0
 AvenirNext-Heavy;}…
```

Text Scaling

Case study — RTF document interchange

Migrate pre-existing documents to new tagged format

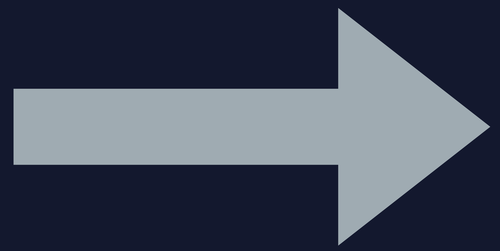
Latest OS

Text Scaling

Case study — RTF document interchange

Migrate pre-existing documents to new tagged format

Latest OS



Text Scaling

Case study — RTF document interchange

NEW

Ensure documents contain accurate text scaling metadata

```
public static let textScaling: NSAttributedString.DocumentAttributeKey
public static let sourceTextScaling: NSAttributedString.DocumentAttributeKey
```

```
// Text Scaling
// Case study — RTF document interchange
// Ensure documents contain accurate text scaling metadata

// Save document with standard text scaling (sets the metadata)
let myRTFData = myAttributedString.data(from: NSMakeRange(0, myAttributedString.length),
    documentAttributes: [.documentType : NSAttributedString.DocumentType.rtf,
                         .textScaling : NSTextScalingType.standard.rawValue])
```

```
// Text Scaling
// Case study — RTF document interchange
// Ensure documents contain accurate text scaling metadata

// Save document with standard text scaling (sets the metadata)
let myRTFData = myAttributedString.data(from: NSMakeRange(0, myAttributedString.length),
    documentAttributes: [.documentType : NSAttributedString.DocumentType.rtf,
                         .textScaling : NSTextScalingType.standard.rawValue])
```

```
// Text Scaling
// Case study — RTF document interchange
// Ensure documents contain accurate text scaling metadata

// Save document with standard text scaling (sets the metadata)
let myRTFData = myAttributedString.data(from: NSMakeRange(0, myAttributedString.length),
    documentAttributes: [.documentType : NSAttributedString.DocumentType.rtf,
                         .textScaling : NSTextScalingType.standard.rawValue])

// Or on save, convert document from iOS text scaling to standard
let myRTFData = myAttributedString.data(from: NSMakeRange(0, myAttributedString.length),
    documentAttributes: [.documentType : NSAttributedString.DocumentType.rtf,
                         .textScaling : NSTextScalingType.standard.rawValue,
                         .sourceTextScaling : NSTextScalingType.iOS.rawValue])
```

```
// Text Scaling
// Case study — RTF document interchange
// Ensure documents contain accurate text scaling metadata

// Save document with standard text scaling (sets the metadata)
let myRTFData = myAttributedString.data(from: NSMakeRange(0, myAttributedString.length),
    documentAttributes: [.documentType : NSAttributedString.DocumentType.rtf,
                         .textScaling : NSTextScalingType.standard.rawValue])

// Or on save, convert document from iOS text scaling to standard
let myRTFData = myAttributedString.data(from: NSMakeRange(0, myAttributedString.length),
    documentAttributes: [.documentType : NSAttributedString.DocumentType.rtf,
                         .textScaling : NSTextScalingType.standard.rawValue,
                         .sourceTextScaling : NSTextScalingType.iOS.rawValue])
```

Text Scaling

Case study — RTF document interchange

NEW

Control conversions when opening document MODEL

```
public static let targetTextScaling: NSAttributedString.DocumentReadingOptionKey
public static let sourceTextScaling: NSAttributedString.DocumentReadingOptionKey
```

```
// Text Scaling
// Case study — RTF document interchange
// Control conversions when reading doucment

// Read in RTF file and convert it from iOS text scaling to standard text scaling
let myDocument = NSAttributedString(data: myRTFData,
                                    options: [.documentType : NSAttributedString.documentType.rtf
                                              .targetTextScaling : NSTextScalingType.standard.rawValue,
                                              .sourceTextScaling : NSTextScalingType.iOS.rawValue],
                           documentAttributes: &myDocumentAttributes)
```

```
// Text Scaling
// Case study — RTF document interchange
// Control conversions when reading doucment

// Read in RTF file and convert it from iOS text scaling to standard text scaling
let myDocument = NSAttributedString(data: myRTFData,
                                    options: [.documentType : NSAttributedString.documentType.rtf
                                              .targetTextScaling : NSTextScalingType.standard.rawValue,
                                              .sourceTextScaling : NSTextScalingType.iOS.rawValue],
                                    documentAttributes: &myDocumentAttributes)
```

Text Scaling

Case study — RTF document interchange

Use standard scaling for viewing document VIEW

```
let myDocument = NSAttributedString(data: myRTFData,
                                    options: [.documentType : NSAttributedString.documentType.rtf,
                                              .targetTextScaling : NSTextScalingType.standard.rawValue],
                                    documentAttributes: &myDocumentAttributes)

myTextView.usesStandardTextScaling = true
myTextView.attributedText = myDocument
```

Text Scaling

Case study — RTF document interchange

Use standard scaling for viewing document VIEW

```
let myDocument = NSAttributedString(data: myRTFData,
                                    options: [.documentType : NSAttributedString.documentType.rtf,
                                              .targetTextScaling : NSTextScalingType.standard.rawValue],
                                    documentAttributes: &myDocumentAttributes)

myTextView.usesStandardTextScaling = true
myTextView.attributedText = myDocument
```

Text Scaling

Case study — RTF document interchange

Use standard scaling for viewing document VIEW

```
let myDocument = NSAttributedString(data: myRTFData,
                                    options: [.documentType : NSAttributedString.documentType.rtf,
                                              .targetTextScaling : NSTextScalingType.standard.rawValue],
                                    documentAttributes: &myDocumentAttributes)

myTextView.usesStandardTextScaling = true
myTextView.attributedText = myDocument
```

Summary

Use name instantiation only for non-System fonts

Use On-Demand Resources to deliver fonts to the OS

Use standard text scaling for best cross-platform user experience

More Information

developer.apple.com/wwdc19/227

Text, Fonts, and SF Symbols Lab	Thursday, 12:00


WWDC19