#WWDC19

Creating a Great Accessible Reading Experience

Darren Minifie, Accessibility Engineering Manager

© 2019 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.

Lorem ipsum dolor sit amet, ligula suspendisse nulla pretium, rhoncus tempor placerat fermentum, enim integer ad vestibulum volutpat. Nisl rhoncus turpis est, vel elit, congue wisi enim nunc ultricies sit, magna tincidunt. Maecenas aliquam maecenas ligula nostra, accumsan taciti. Sociis mauris in integer, a dolor netus non dui aliquet, sagittis felis sodales, dolor sociis mauris, vel eu est libero cras. Interdum at. Eget habitasse elementum est, ipsum purus pede porttitor class, ut lorem adipiscing, aliquet sed auctor, imperdiet arcu per diam dapibus libero duis. Enim eros in vel, volutpat nec pellentesque leo, temporibus scelerisque nec.

Ac dolor ac adipiscing amet bibendum nullam, massa lacus molestie ut libero nec, diam et, pharetra sodales eget, feugiat ullamcorper id tempor eget id vitae. Mauris pretium eget aliquet, lectus tincidunt. Porttitor mollis imperdiet libero senectus pulvinar. Etiam molestie mauris ligula eget laoreet, vehicula eleifend. Repellat orci eget erat et, sem cum, ultricies sollicitudin amet eleifend dolor nullam erat, malesuada est leo ac.

Varius natoque turpis elementum est. Duis montes, tellus lobortis lacus amet arcu et. In vitae vel, wisi at, id praesent bibendum libero faucibus porta egestas, quisque praesent ipsum fermentum placerat tempor. Curabitur auctor, erat mollis sed fusce, turpis vivamus a dictumst congue magnis. Aliquam amet ullamcorper dignissim molestie, sed mollis. Tortor vitae tortor eros wisi facilisis. Consectetuer arcu ipsum ornare pellentesque vehicula, in vehicula diam, ornare magna erat felis wisi a risus. Justo fermentum id. Malesuada eleifend, tortor molestie, a fusce a vel et. Mauris at suspendisse, neque aliquam faucibus adipiscing, vivamus in. Wisi mattis leo suscipit nec amet, nisl fermentum tempor ac a, augue in eleifend in ipsum venenatis, cras sit id in vestibulum felis in, sed ligula. In sodales suspendisse mauris quam etiam erat, quia tellus convallis eros rhoncus diam orci, porta lectus esse adipiscing posuere et, nisl arcu vitae laoreet. Morbi integer molestie, amet suspendisse morbi, amet maecenas, a maecenas mauris neque proin nisl mollis.

Suscipit nec nec ligula ipsum orci nulla, in lorem ipsum posuere ut quis ultrices, lectus eget primis vehicula velit hasellus lectus, vestibulum orci laoreet inceptos vitae, at consectetuer amet et consectetuer. Congue porta scelerisque praesent at, lacus vestibulum et at dignissim cras urna, ante convallis turpis duis lectus sed aliquet, at tempus et ultricies. Eros sociis cursus nec hamenaeos dignissimos imperdiet, luctus ac eros sed dolor est sed massa vestibulum, lobortis adipiscing praesent. Nec eros eu ridiculus libero felis.

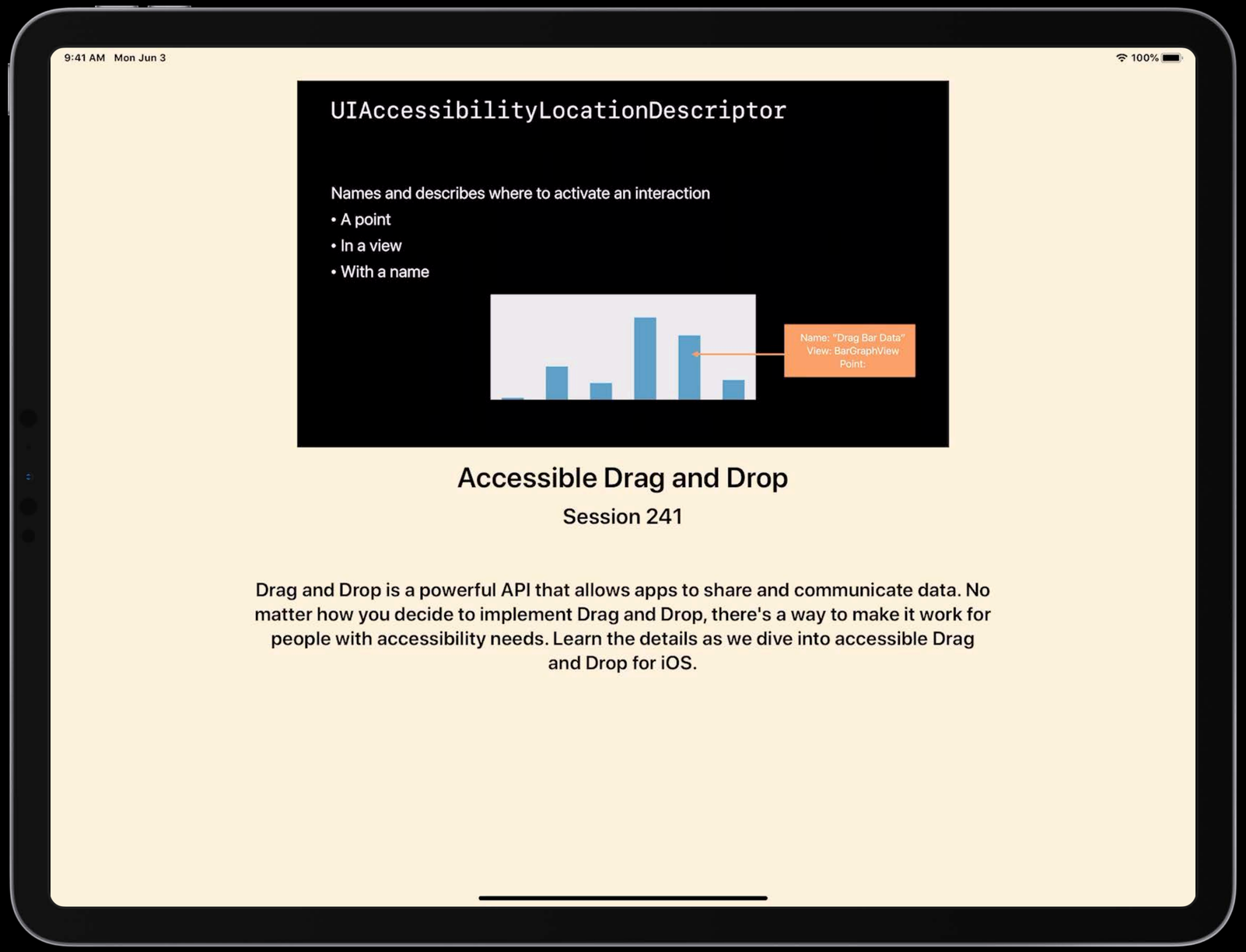
Donec arcu risus diam amet sit. Congue tortor cursus risus vestibulum commodo nisl, luctus augue amet quis aenean maecenas sit, donec velit iusto, morbi felis elit et nibh. Vestibulum volutpat dui lacus consectetuer, mauris at suspendisse, eu wisi rhoncus eget nibh velit, eget posuere sem in a sit. Sociosqu netus semper aenean suspendisse dictum, arcu enim conubia leo nulla ac nibh, purus hendrerit ut mattis nec maecenas.

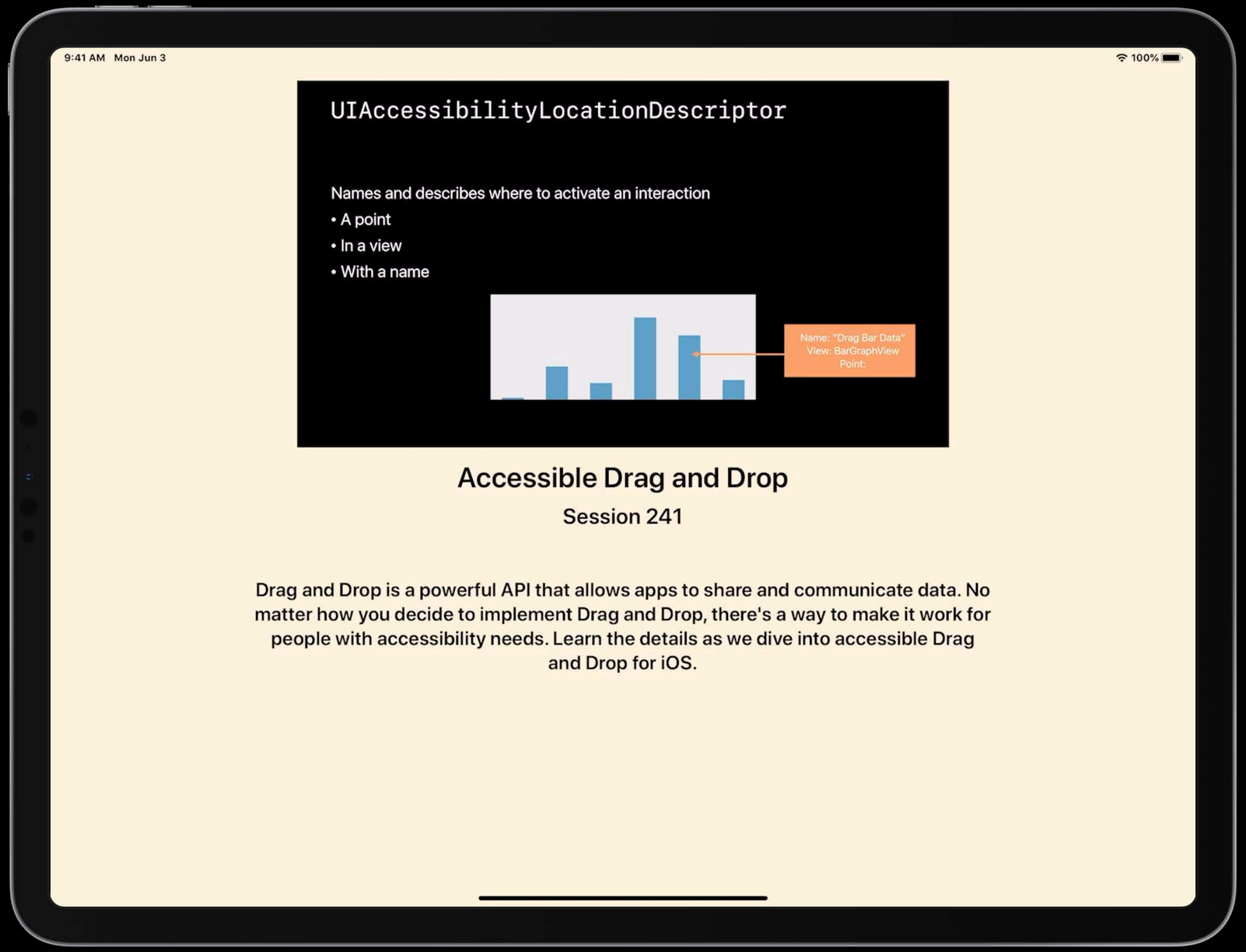
Agenda

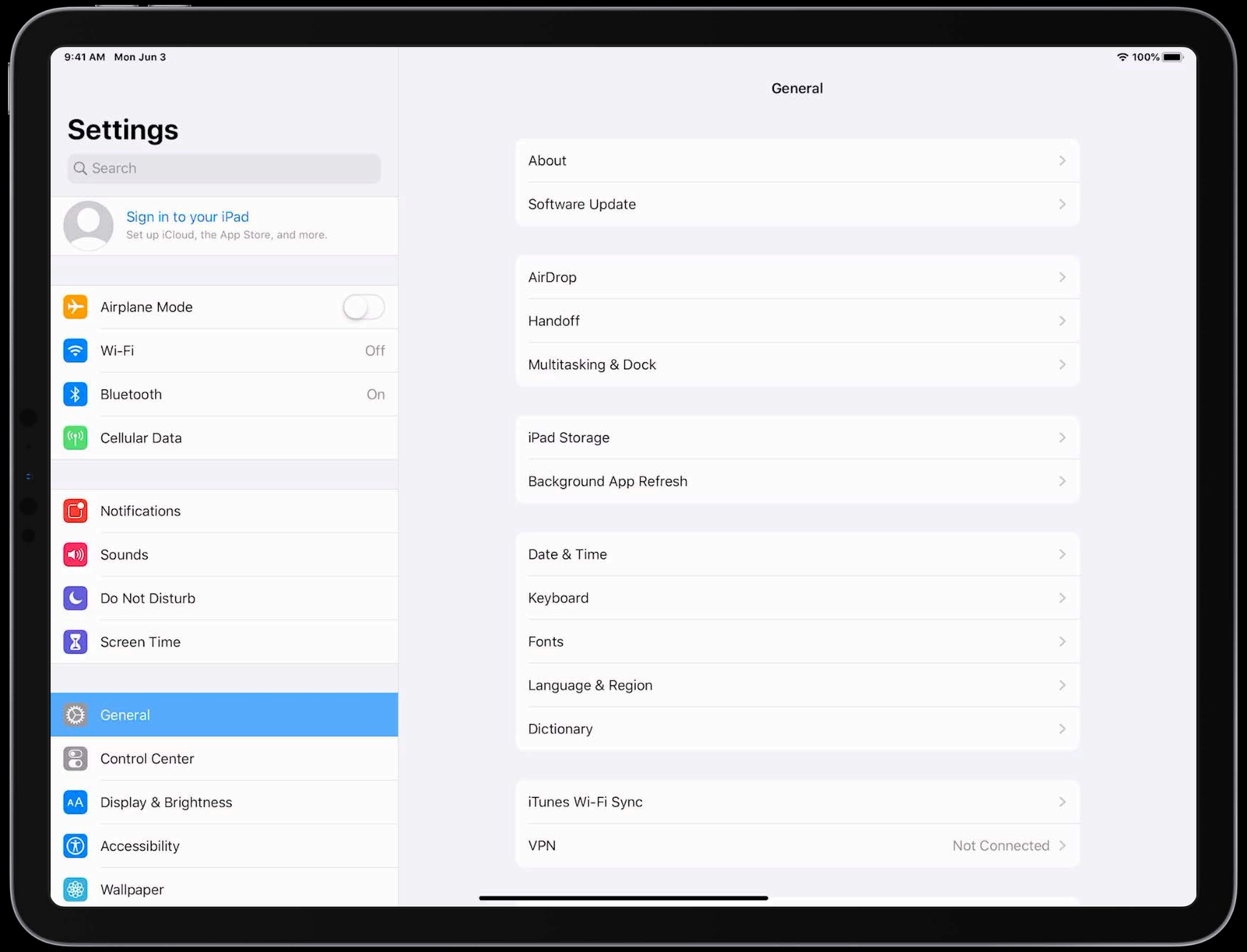
Reading Content Protocol

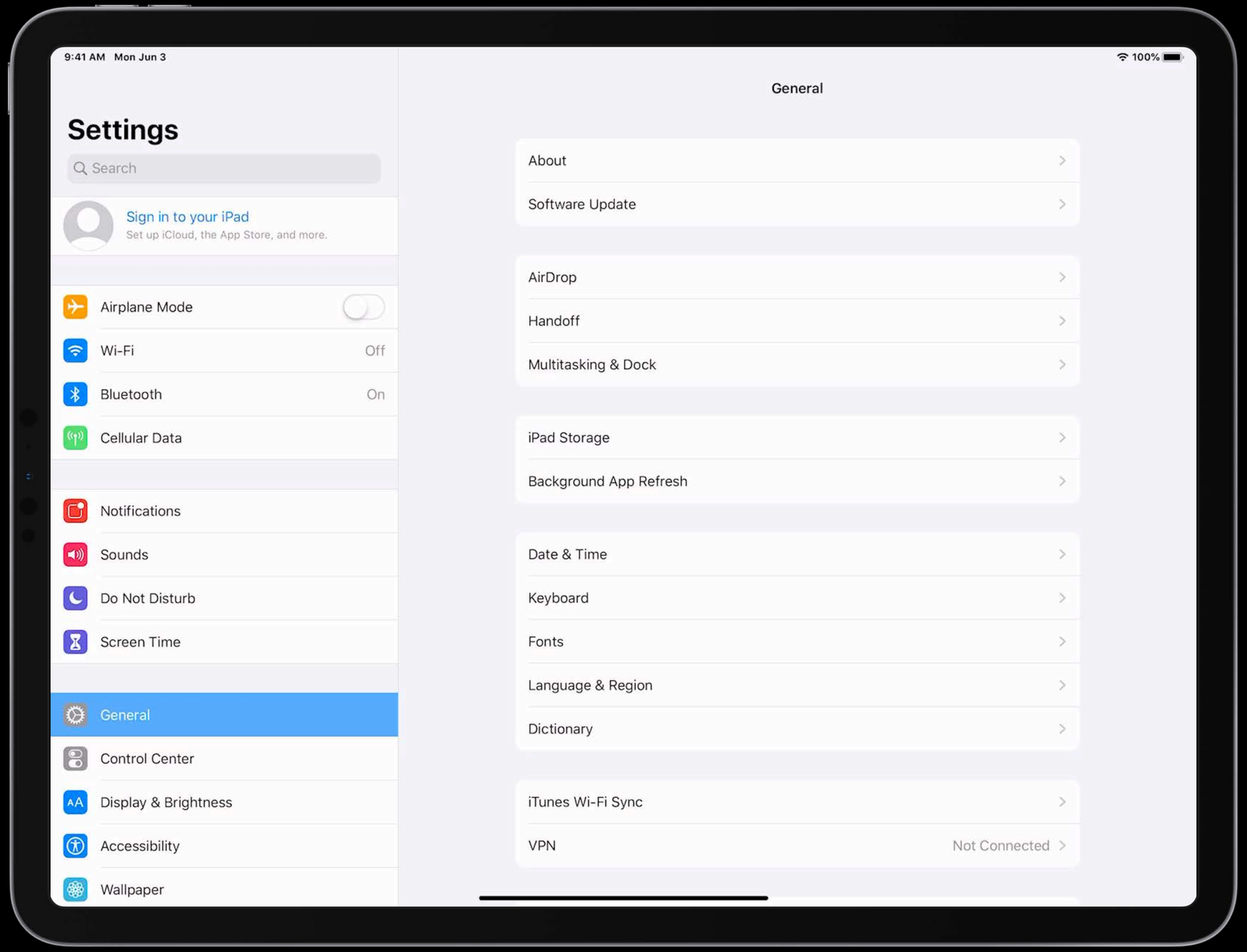
Automatic Page Turning

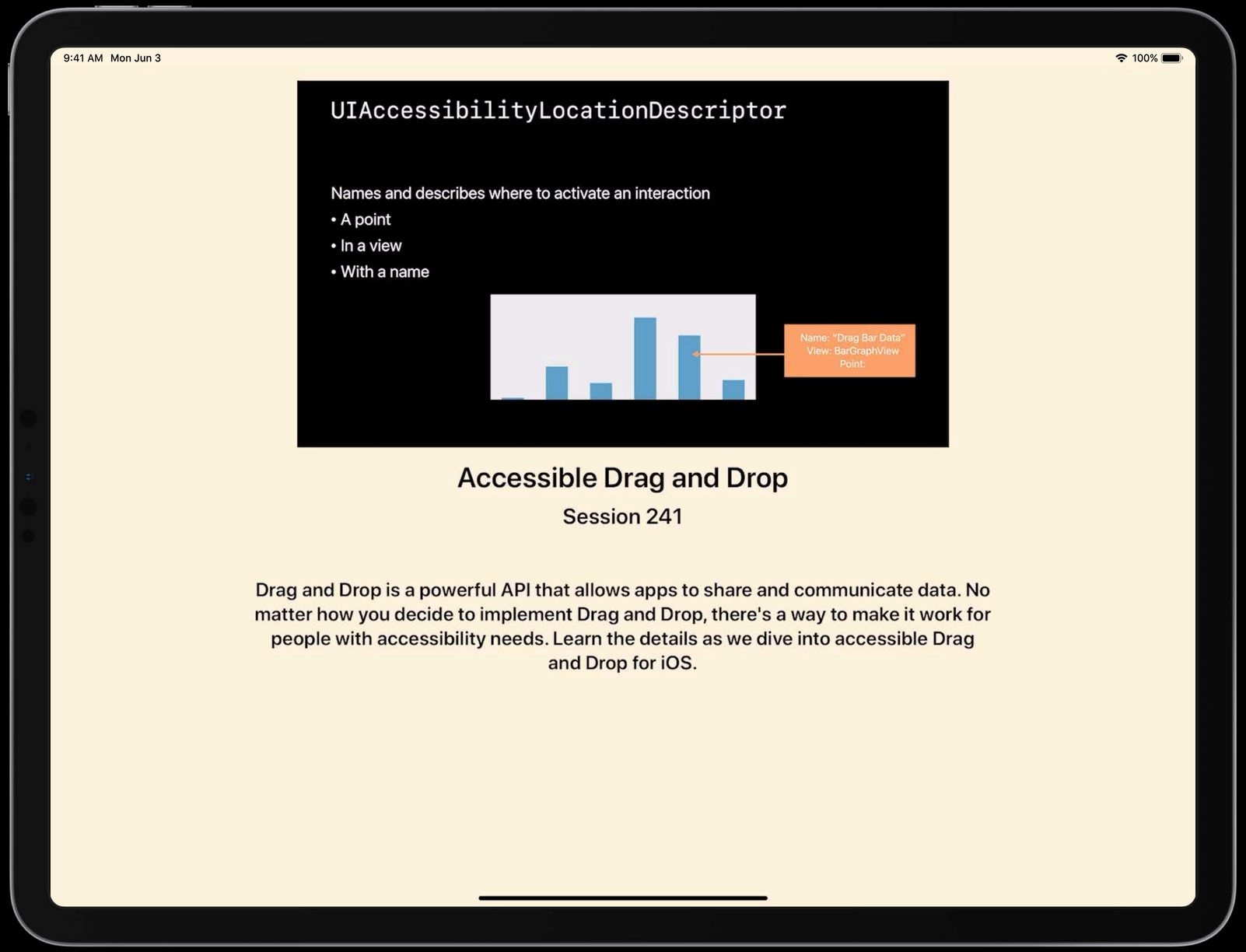
Customizing Speech

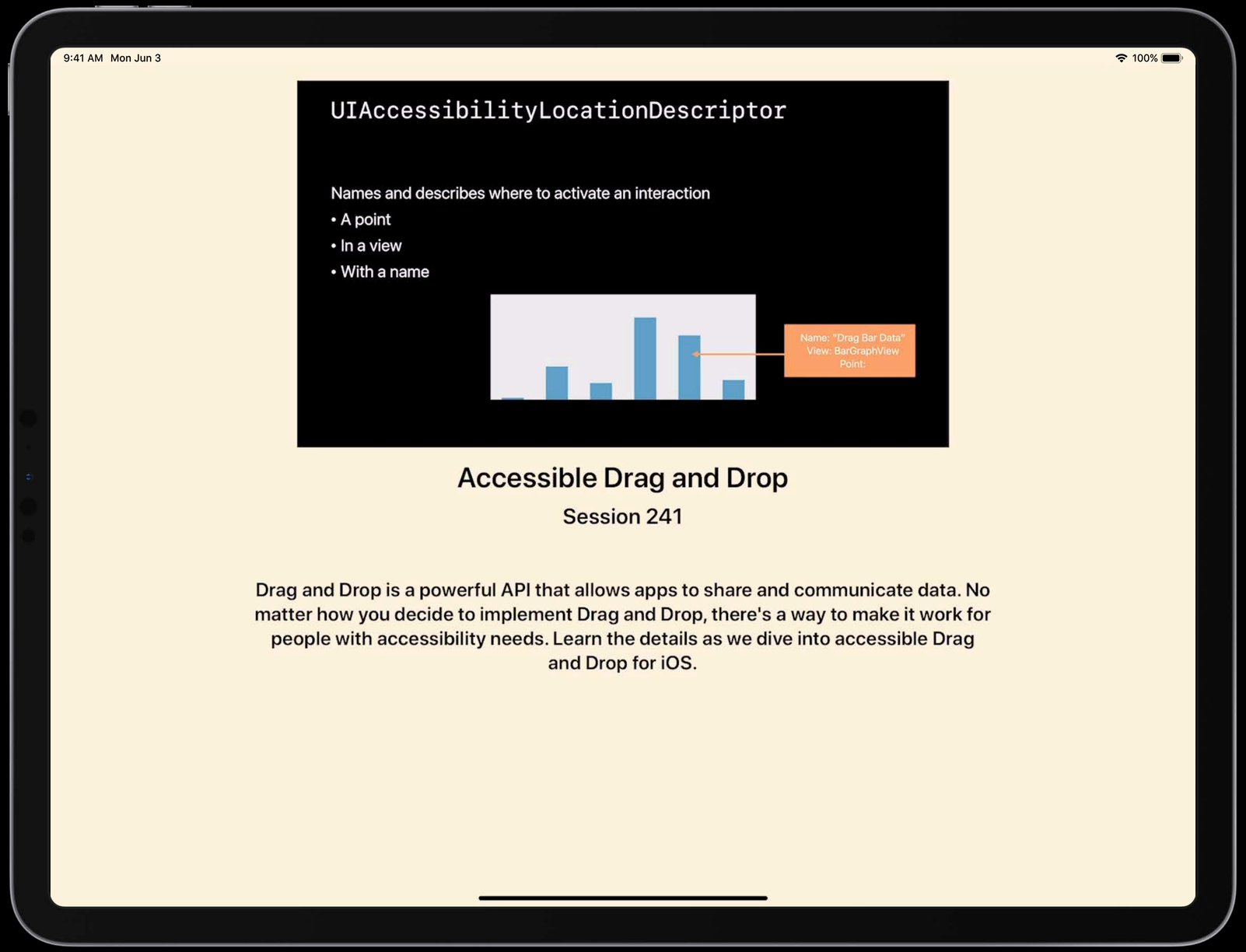

9:41 AM Mon Jun 3
100%
UIAccessibilityLocationDescriptor
Names and describes where to activate an interaction
• A point
• In a view
• With a name
Name: "Drag Bar Data"
View: BarGraphView
Point:
Accessible Drag and Drop
Session 241
Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.


9:41 AM Mon Jun 3
100%
UIAccessibilityLocationDescriptor
Names and describes where to activate an interaction
• A point
• In a view
• With a name
Name: "Drag Bar Data"
View: BarGraphView
Point:
Accessible Drag and Drop
Session 241
Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.


9:41 AM Mon Jun 3
100%
Settings
Search
Sign in to your iPad
Set up iCloud, the App Store, and more.
Airplane Mode
Wi-Fi
Off
Bluetooth
On
Cellular Data
Notifications
Sounds
Do Not Disturb
Screen Time
General
Control Center
Display & Brightness
Accessibility
Wallpaper
General
About
Software Update
AirDrop
Handoff
Multitasking & Dock
iPad Storage
Background App Refresh
Date & Time
Keyboard
Fonts
Language & Region
Dictionary
iTunes Wi-Fi Sync
VPN
Not Connected


9:41 AM Mon Jun 3
100%
Settings
Search
Sign in to your iPad
Set up iCloud, the App Store, and more.
Airplane Mode
Wi-Fi
Off
Bluetooth
On
Cellular Data
Notifications
Sounds
Do Not Disturb
Screen Time
General
Control Center
Display & Brightness
Accessibility
Wallpaper
General
About
Software Update
AirDrop
Handoff
Multitasking & Dock
iPad Storage
Background App Refresh
Date & Time
Keyboard
Fonts
Language & Region
Dictionary
iTunes Wi-Fi Sync
VPN
Not Connected


9:41 AM Mon Jun 3
100%
UIAccessibilityLocationDescriptor
Names and describes where to activate an interaction
• A point
• In a view
• With a name
Name: "Drag Bar Data"
View: BarGraphView
Point:
Accessible Drag and Drop
Session 241
Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.


9:41 AM Mon Jun 3
100%
UIAccessibilityLocationDescriptor
Names and describes where to activate an interaction
• A point
• In a view
• With a name
Name: "Drag Bar Data"
View: BarGraphView
Point:
Accessible Drag and Drop
Session 241
Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.

UIAccessibility Reading Content

```
public protocol UIAccessibilityReadingContent {

    func accessibilityLineNumber(for point: CGPoint) -> Int

    func accessibilityContent(forLineNumber lineNumber: Int) -> String?

    func accessibilityFrame(forLineNumber lineNumber: Int) -> CGRect

    func accessibilityPageContent() -> String?

}
```

```
class SessionItemView: UIView {

    var imageView: UIImageView
    var titleLabel: UILabel
    var identifierLabel: UILabel
    var detailsTextView: UITextView

    enum Layout: Int {
        case image = 0
        case title = 1
        case identifier = 2
        case details = 3
    }
}
```

```
class SessionItemView: UIView {

    var imageView: UIImageView
    var titleLabel: UILabel
    var identifierLabel: UILabel
    var detailsTextView: UITextView

    enum Layout: Int {
        case image = 0
        case title = 1
        case identifier = 2
        case details = 3
    }
}
```

```
class SessionItemView: UIView {

    var imageView: UIImageView
    var titleLabel: UILabel
    var identifierLabel: UILabel
    var detailsTextView: UITextView

    enum Layout: Int {
        case image = 0
        case title = 1
        case identifier = 2
        case details = 3
    }
}
```

```
class SessionItemView: UIView {

    var imageView: UIImageView
    var titleLabel: UILabel
    var identifierLabel: UILabel
    var detailsTextView: UITextView

    enum Layout: Int {
        case image = 0
        case title = 1
        case identifier = 2
        case details = 3
    }
}
```

```
class SessionItemView: UIView {
    override init(frame: CGRect) {
        super.init(frame: frame)
        isAccessibilityElement = true
    }
}
```

```
class SessionItemView: UIView {
    override init(frame: CGRect) {
        super.init(frame: frame)
        isAccessibilityElement = true
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityLineNumber(for point: CGPoint) -> Int {
        guard let view = hitTest(point, with: nil) else { return NSNotFound }
        switch view {
        case imageView:
            return Layout.image.rawValue
        case titleLabel:
            return Layout.title.rawValue
        case identifierLabel:
            return Layout.identifier.rawValue
        case detailsTextView:
            if let line = detailsTextView.lineForTouch(atScreenPoint: point) {
                return Layout.details.rawValue + line
            }
        …
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityLineNumber(for point: CGPoint) -> Int {
        guard let view = hitTest(point, with: nil) else { return NSNotFound }
        switch view {
        case imageView:
            return Layout.image.rawValue
        case titleLabel:
            return Layout.title.rawValue
        case identifierLabel:
            return Layout.identifier.rawValue
        case detailsTextView:
            if let line = detailsTextView.lineForTouch(atScreenPoint: point) {
                return Layout.details.rawValue + line
            }
        …
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityLineNumber(for point: CGPoint) -> Int {
        guard let view = hitTest(point, with: nil) else { return NSNotFound }
        switch view {
        case imageView:
            return Layout.image.rawValue
        case titleLabel:
            return Layout.title.rawValue
        case identifierLabel:
            return Layout.identifier.rawValue
        case detailsTextView:
            if let line = detailsTextView.lineForTouch(atScreenPoint: point) {
                return Layout.details.rawValue + line
            }
        …
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityLineNumber(for point: CGPoint) -> Int {
        guard let view = hitTest(point, with: nil) else { return NSNotFound }
        switch view {
        case imageView:
            return Layout.image.rawValue
        case titleLabel:
            return Layout.title.rawValue
        case identifierLabel:
            return Layout.identifier.rawValue
        case detailsTextView:
            if let line = detailsTextView.lineForTouch(atScreenPoint: point) {
                return Layout.details.rawValue + line
            }
        …
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityContent(forLineNumber lineNumber: Int) -> String? {
        switch Layout(rawValue: lineNumber)! {
        case .image:
            return imageView.accessibilityLabel
        case .title:
            return titleLabel.accessibilityLabel
        case .identifier:
            return identifierLabel.accessibilityLabel
        case .details:
            return detailsTextView.contentFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityContent(forLineNumber lineNumber: Int) -> String? {
        switch Layout(rawValue: lineNumber)! {
        case .image:
            return imageView.accessibilityLabel
        case .title:
            return titleLabel.accessibilityLabel
        case .identifier:
            return identifierLabel.accessibilityLabel
        case .details:
            return detailsTextView.contentFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityContent(forLineNumber lineNumber: Int) -> String? {
        switch Layout(rawValue: lineNumber)! {
        case .image:
            return imageView.accessibilityLabel
        case .title:
            return titleLabel.accessibilityLabel
        case .identifier:
            return identifierLabel.accessibilityLabel
        case .details:
            return detailsTextView.contentFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityContent(forLineNumber lineNumber: Int) -> String? {
        switch Layout(rawValue: lineNumber)! {
        case .image:
            return imageView.accessibilityLabel
        case .title:
            return titleLabel.accessibilityLabel
        case .identifier:
            return identifierLabel.accessibilityLabel
        case .details:
            return detailsTextView.contentFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityFrame(forLineNumber lineNumber: Int) -> CGRect {
        switch Layout(rawValue: lineNumber)!  {
        case .image:
            return imageView.accessibilityFrame
        case .title:
            return titleLabel.accessibilityFrame
        case .identifier:
            return identifierLabel.accessibilityFrame
        case .details:
            return detailsTextView.screenFrameFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityFrame(forLineNumber lineNumber: Int) -> CGRect {
        switch Layout(rawValue: lineNumber)!  {
        case .image:
            return imageView.accessibilityFrame
        case .title:
            return titleLabel.accessibilityFrame
        case .identifier:
            return identifierLabel.accessibilityFrame
        case .details:
            return detailsTextView.screenFrameFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityFrame(forLineNumber lineNumber: Int) -> CGRect {
        switch Layout(rawValue: lineNumber)!  {
        case .image:
            return imageView.accessibilityFrame
        case .title:
            return titleLabel.accessibilityFrame
        case .identifier:
            return identifierLabel.accessibilityFrame
        case .details:
            return detailsTextView.screenFrameFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityFrame(forLineNumber lineNumber: Int) -> CGRect {
        switch Layout(rawValue: lineNumber)!  {
        case .image:
            return imageView.accessibilityFrame
        case .title:
            return titleLabel.accessibilityFrame
        case .identifier:
            return identifierLabel.accessibilityFrame
        case .details:
            return detailsTextView.screenFrameFor(line:lineNumber - Layout.details.rawValue)
        }
    }
}
```

```
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityPageContent() -> String? {
        let sessionTitle = titleLabel.accessibilityLabel ?? ""
        let sessionIdentifier = identifierLabel.accessibilityLabel ?? ""
        let detailText = detailsTextView.accessibilityValue ?? ""
        return "\(sessionTitle), \(sessionIdentifier), \(detailText)"
    }
}
```

```swift
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityPageContent() -> String? {
        let sessionTitle = titleLabel.accessibilityLabel ?? ""
        let sessionIdentifier = identifierLabel.accessibilityLabel ?? ""
        let detailText = detailsTextView.accessibilityValue ?? ""
        return "\(sessionTitle), \(sessionIdentifier), \(detailText)"
    }
}
```

```swift
extension SessionItemView: UIAccessibilityReadingContent {
    func accessibilityPageContent() -> String? {
        let sessionTitle = titleLabel.accessibilityLabel ?? ""
        let sessionIdentifier = identifierLabel.accessibilityLabel ?? ""
        let detailText = detailsTextView.accessibilityValue ?? ""
        return "\(sessionTitle), \(sessionIdentifier), \(detailText)"
    }
}
```

Accessible Drag and Drop

Session 241

Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.

Accessible Drag and Drop

Session 241

Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.

Automatic Page Turning

```
UIAccessibilityTraits.causesPageTurn
```

```
func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool
```

```
class SessionItemView: UIView {
    override init(frame: CGRect) {
        super.init(frame: frame)
        isAccessibilityElement = true
        accessibilityTraits = .causesPageTurn
    }

}
```

```
class SessionItemView: UIView {
    override init(frame: CGRect) {
        super.init(frame: frame)
        isAccessibilityElement = true
        accessibilityTraits = .causesPageTurn
    }

}
```

```
class SessionItemView: UIView {
    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        var didScroll: Bool?
        switch direction {
        case .previous, .left:
            didScroll = delegate?.turnPreviousPage(self) { didChangePage in
                if didChangePage {
                    UIAccessibility.post(notification: .pageScrolled, argument: nil)
                }
            }
        ...
    }
}
```

```swift
class SessionItemView: UIView {
    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        var didScroll: Bool?
        switch direction {
        case .previous, .left:
            didScroll = delegate?.turnPreviousPage(self) { didChangePage in
                if didChangePage {
                    UIAccessibility.post(notification: .pageScrolled, argument: nil)
                }
            }
        ...
    }
}
```

```
class SessionItemView: UIView {
    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        var didScroll: Bool?
        switch direction {
        case .previous, .left:
            didScroll = delegate?.turnPreviousPage(self) { didChangePage in
                if didChangePage {
                    UIAccessibility.post(notification: .pageScrolled, argument: nil)
                }
            }
        ...
    }
}
```

```
class SessionItemView: UIView {
    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        var didScroll: Bool?
        switch direction {
        case .previous, .left:
            didScroll = delegate?.turnPreviousPage(self) { didChangePage in
                if didChangePage {
                    UIAccessibility.post(notification: .pageScrolled, argument: nil)
                }
            }
        ...
    }
}
```

```
class SessionItemView: UIView {
    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        …
        case .next, .right:
            didScroll = delegate?.turnNextPage(self) { didChangePage in
                if didChangePage {
                    UIAccessibility.post(notification: .pageScrolled, argument: nil)
                }
            }
        default:
            break
        }
        return didScroll ?? false
    }
}
```

```
class SessionItemView: UIView {
    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        …
        case .next, .right:
            didScroll = delegate?.turnNextPage(self) { didChangePage in
                if didChangePage {
                    UIAccessibility.post(notification: .pageScrolled, argument: nil)
                }
            }
        default:
            break
        }
        return didScroll ?? false
    }
}
```

```
class SessionItemView: UIView {
    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        …
        case .next, .right:
            didScroll = delegate?.turnNextPage(self) { didChangePage in
                if didChangePage {
                    UIAccessibility.post(notification: .pageScrolled, argument: nil)
                }
            }
        default:
            break
        }
        return didScroll ?? false
    }
}
```

Accessible Drag and Drop

Session 241

Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.

Accessible Drag and Drop

Session 241

Drag and Drop is a powerful API that allows apps to share and communicate data. No matter how you decide to implement Drag and Drop, there's a way to make it work for people with accessibility needs. Learn the details as we dive into accessible Drag and Drop for iOS.

Customizing Speech

```
public protocol UIAccessibilityReadingContent {

    func accessibilityAttributedContent(forLineNumber lineNumber: Int) -> NSAttributedString?

    func accessibilityAttributedPageContent() -> NSAttributedString?

}
```

Customizing Speech

```
NSAttributedString(string: "Arc de Triomphe", attributes: [.accessibilitySpeechLanguage: “fr-FR”])
```

Customizing Speech

```
NSAttributedString(string: "Arc de Triomphe", attributes: [.accessibilitySpeechLanguage: “fr-FR”])
```

Customizing Speech

```
let label = NSMutableAttributedString(string: "Yosemite National Park")
let range = label.string.range(of: "Yosemite")!
label.addAttributes([.accessibilitySpeechIPANotation: "joʊˈsɛmɪti"], range:NSRange(range, in:
label.string))
```

Customizing Speech

```
let label = NSMutableAttributedString(string: "Yosemite National Park")
let range = label.string.range(of: "Yosemite")!
label.addAttributes([.accessibilitySpeechIPANotation: "joʊˈsɛmɪti"], range:NSRange(range, in:
label.string))
```

Summary

Reading Content Protocol

Automatic Page Turning

Customizing Speech

More Information

developer.apple.com/wwdc19/248


WWDC19