What’s New in Accessibility

Session 215

Skylar Peterson, Software Engineer
Making technology usable by everyone
Cognitive
Motor
Vision
Hearing
Over 1 Billion

People have a disability worldwide

http://www.who.int/disabilities/en/
1 in 7

People have a disability worldwide

http://www.who.int/disabilities/en/
New assistive features
New assistive features

Auditing your app for accessibility
New assistive features
Auditing your app for accessibility
Accessibility API basics
New assistive features
Auditing your app for accessibility
Accessibility API basics
Beyond the basics
New assistive features
Auditing your app for accessibility
Accessibility API basics
Beyond the basics
Drag and Drop accessibility
New Assistive Features
CAN'T BELIEVE I TOOK THIS WITH MY IPHONE 😍
Improved Photo Descriptions
Large Text
## Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design for Everyone</td>
<td>Hall 3</td>
<td>Thursday 11:00AM</td>
</tr>
<tr>
<td>Auto Layout Techniques in Interface Builder</td>
<td>Hall 3</td>
<td>Friday 9:00AM</td>
</tr>
<tr>
<td>Building Apps with Dynamic Type</td>
<td>Executive Ballroom</td>
<td>Friday 1:50PM</td>
</tr>
</tbody>
</table>
Accessibility Keyboard
Type to Siri
Smart Invert Colors
Demo
Smart Invert Colors
Auditing Your App for Accessibility
Auditing
Auditing
Auditing
Auditing

Up vote, 3 votes, Button
Welcome to Xcode
Version 9.2 beta (GN736)

- Get started with a playground
- Explore new capabilities easily
- Create a new macOS project
- Develop apps for iPhone, iPad, Mac, Apple Watch or Apple TV
- Clone an existing project
- Start working on an existing project from an XSM repository.
Accessibility Inspector
More Information

Auditing your Apps for Accessibility
Demo

Auditing an app
Audit Results
Audit Results
Audit Results
Audit Results
Audit Results
Audit Results
Accessibility API Basics
What are you?
UIAccessibility Protocol

What are you?

Button
UIAccessibility Protocol

What are you?

Button
UIAccessibility Protocol

Who are you?

What are you?

Button

Who are you?
UIAccessibility Protocol

What are you?

Button

Who are you?
UIAccessibility Protocol

Who are you?

Notifications

What are you?

Button

Who are you?

Notifications
UIAccessibility Protocol

Who are you?

What are you?

Button

Notifications
UIAccessibility Protocol

Who are you?

What are you?

Notifications

What’s your value?

Button
UIAccessibility Protocol

Who are you?

What's your value?

Notifications

What are you?

Button

Who are you?

Notifications

What's your value?
UIAccessibility Protocol

- What are you?
- Button
- Who are you?
- Notifications
- What’s your value?
- 3 Unread
UIAccessibility Protocol

- `control.accessibilityTraits`
- `UIAccessibilityTraitButton`
- `control.accessibilityLabel` = "Notifications"
- `control.accessibilityValue` = "3 Unread"
// UIAccessibility Basics

extension NSObject {
    open var isAccessibilityElement: Bool
    open var accessibilityLabel: NSString?
    open var accessibilityTraits: UIAccessibilityTraits
    open var accessibilityValue: NSString?
    open var accessibilityHint: NSString?
}

open var isAccessibilityElement: Bool

// Example
memoryView.isAccessibilityElement = true
open var isAccessibilityElement: Bool

// Example
memoryView.isAccessibilityElement = true
open var isAccessibilityElement: Bool

// Example
memoryView.isAccessibilityElement = true
open var isAccessibilityElement: Bool

// Example
playButton.isAccessibilityElement = false
open var accessibilityLabel: NSString?

// Example
memoryView.accessibilityLabel = "Memory, February 18 2017"
open var accessibilityLabel: NSString?

// Example
memoryView.accessibilityLabel = "Memory, February 18 2017"
open var accessibilityLabel: NSString?

// Example
memoryView.accessibilityLabel = "Memory, February 18 2017"
open var accessibilityTraits: UIAccessibilityTraits

// Example
memoryView.accessibilityTraits |= UIAccessibilityTraitButton
open var accessibilityTraits: UIAccessibilityTraits

// Example
memoryView.accessibilityTraits |= UIAccessibilityTraitButton
open var accessibilityTraits: UIAccessibilityTraits

// Example
memoryView.accessibilityTraits |= UIAccessibilityTraitButton
open var accessibilityValue: NSString?

// Example
videoScrubber.accessibilityValue = "\(elapsedTime) seconds"
UIAccessibility
Basics

```swift
open var accessibilityValue: NSString?

// Example
videoScrubber.accessibilityValue = "\(elapsedTime) seconds"
```
open var accessibilityValue: NSString?

// Example
videoScrubber.accessibilityValue = "\(elapsedTime) seconds"
UIAccessibility
Basics

```swift
open var accessibilityHint: NSString?

// Example
videoScrubber.accessibilityHint = "Swipe up or down with one finger to adjust the value"
```
UIAccessibility
Basics

open var accessibilityHint: NSString?

// Example
videoScrubber.accessibilityHint = "Swipe up or down with one finger to adjust the value"
UIAccessibility
Basics

```swift
open var accessibilityHint: NSString?

// Example
videoScrubber.accessibilityHint = "Swipe up or down with one finger to adjust the value"
```
extension NSObject {
    open var accessibilityAttributedLabel: NSAttributedString?
    open var accessibilityAttributedValue: NSAttributedString?
    open var accessibilityAttributedHint: NSAttributedString?
}

// Attributed Properties
extension NSObject {
    open var accessibilityAttributedLabel: NSAttributedString?
    open var accessibilityAttributedValue: NSAttributedString?
    open var accessibilityAttributedHint: NSAttributedString?
}
let attributedLabel = NSAttributedString(string: "Bonjour", attributes: [
    UIAccessibilitySpeechAttributeLanguage : "fr-FR"
])
imageView.accessibilityAttributedLabel = attributedLabel

let newValue = NSAttributedString(string: "50\%", attributes: [
    UIAccessibilitySpeechAttributeQueueAnnouncement : true
])
UIAccessibilityPostNotification(UIAccessibilityAnnouncementNotification, newValue)
let attributedLabel = NSAttributedString(string: "Bonjour", attributes: [UIAccessibilitySpeechAttributeLanguage: "fr-FR"]
imageView.accessibilityAttributedLabel = attributedLabel

let newValue = NSAttributedString(string: "50\%", attributes: [UIAccessibilitySpeechAttributeQueueAnnouncement: true])
UIAccessibilityPostNotification(UIAccessibilityAnnouncementNotification, newValue)
let attributedLabel = NSAttributedString(string: "Bonjour", attributes:
[ UIAccessibilitySpeechAttributeLanguage : "fr-FR" ]
imageView.accessibilityAttributedLabel = attributedLabel

let newValue = NSAttributedString(string: "50\%", attributes:
[ UIAccessibilitySpeechAttributeQueueAnnouncement : true ]
UIAccessibilityPostNotification(UIAccessibilityAnnouncementNotification, newValue)
# Budget

**MONEY IN**

- Paycheck: $4,000
- Additional income: $0
  
**TOTAL INCOME**: $4,000

**MONEY OUT**

- Housing (Rent, mortgage, taxes, insurance): $1,500
- Transportation: $300
- Utilities: $200
- Groceries: $500
- Medical: $30
- Dining, travel, entertainment: $120
- Debt payments: $50
- Savings: $0
- Education: $100
- Miscellaneous: $125

**TOTAL EXPENSES**: $2,925
extension NSObject {
    open var accessibilityContainerType: UIAccessibilityContainerType
}

public enum UIAccessibilityContainerType : Int {
    case none
    case dataTable // requires conformance to UIAccessibilityContainerDataTable protocol
    case list
    case landmark
}
extension NSObject {
    open var accessibilityContainerType: UIAccessibilityContainerType
}

public enum UIAccessibilityContainerType : Int {
    case none
    case dataTable // requires conformance to UIAccessibilityContainerDataTable protocol
    case list
    case landmark
}

// Containers

extension NSObject {
    open var accessibilityContainerType: UIAccessibilityContainerType
}

public enum UIAccessibilityContainerType : Int {
    case none
    case dataTable // requires conformance to UIAccessibilityContainerDataTable protocol
    case list
    case landmark
}
Non-View Elements
open class UIAccessibilityElement: NSObject {
    public init(accessibilityContainer container: AnyObject)
}

extension NSObject {
    open var accessibilityElements: [AnyObject]?
}
open class UIAccessibilityElement : NSObject {
    public init(accessibilityContainer: AnyObject)
}

extension NSObject {
    open var accessibilityElements: [AnyObject]?
}
open class UIAccessibilityElement : NSObject {
    public init(accessibilityContainer container: AnyObject)
}

extension NSObject {
    open var accessibilityElements: [AnyObject]?
}
Lexi Torres
Just finished my second day of my first WWDC ever! My brain hurts from information overload 😞

3 Comments

Greg Apodaca
We have fun at home

8 Comments
Demo

Using UIAccessibility basics
Beyond the Basics
Common problems and solutions
Custom Actions
// Custom Actions

extension NSObject {
    open var accessibilityCustomActions: [UIAccessibilityCustomAction]?
}

open class UIAccessibilityCustomAction : NSObject {
    open var name: String
    open var target: AnyObject?
    open var selector: Selector

    public init(name: String, target: Any?, selector: Selector)
}
// Custom Actions

extension NSObject {
    open var accessibilityCustomActions: [UIAccessibilityCustomAction]?
}

open class UIAccessibilityCustomAction : NSObject {
    open var name: String
    open var target: AnyObject?
    open var selector: Selector

    public init(name: String, target: Any?, selector: Selector)
}
// Custom Actions

extension NSObject {
    open var accessibilityCustomActions: [UIAccessibilityCustomAction]?
}

open class UIAccessibilityCustomAction : NSObject {
    open var name: String
    open var target: AnyObject?
    open var selector: Selector

    public init(name: String, target: Any?, selector: Selector)
}

class StandardPostCollectionViewCell : UICollectionViewCell {

    func commonInit() {
        // Setup Code...
        let customAction = UIAccessibilityCustomAction(name: "Delete", target: self, selector: #selector(deleteCellAction))

        accessibilityCustomActions = [customAction]
    }

    func deleteCellAction() -> Bool {
        // Code for initiating cell delete
    }
}
// UIAccessibilityCustomAction Example

class StandardPostCollectionViewCell: UICollectionViewCell {

    func commonInit() {
        // Setup Code...
        let customAction = UIAccessibilityCustomAction(name: "Delete", target: self, selector: #selector(deleteCellAction))
        accessibilityCustomActions = [customAction]
    }

    func deleteCellAction() -> Bool {
        // Code for initiating cell delete
    }
}

// UIAccessibilityCustomAction Example

class StandardPostCollectionViewCell : UICollectionViewCell {

    func commonInit() {
        // Setup Code...
        let customAction = UIAccessibilityCustomAction(name: "Delete", target: self, selector: #selector(deleteCellAction))
        accessibilityCustomActions = [customAction]
    }

    func deleteCellAction() -> Bool {
        // Code for initiating cell delete
    }
}
Default Activation
extension NSObject {
    open func accessibilityActivate() -> Bool // Boolean indicates success of activation
}
extension NSObject {
    open func accessibilityActivate() -> Bool // Boolean indicates success of activation
}

// Accessibility Activate
// Accessibility Activate Example

class SwitchTableViewCell : UITableViewCell {
    let switchView: UISwitch

    override func accessibilityActivate() -> Bool {
        switchView.setOn(!switchView.isOn, animated: true)
        return true
    }
}
// Accessibility Activate Example

class SwitchTableViewCell : UITableViewCell {
    let switchView: UISwitch

    override func accessibilityActivate() -> Bool {
        switchView.setOn(!switchView.isOn, animated: true)
        return true
    }
}

Adjustable

Tom McNeil
2 Open Posts

Euna Kwon
44m Remaining

Tom McNeil
27m Remaining

Feeling like the pressure is off now that I can talk about everything I've been working on!

2 Comments

Chella Sapkarov
6m Remaining

So excited to show you all what we've been working on! 💫
Adjustable
// Views That Increment and Decrement

extension NSObject {
    // Must have the UIAccessibilityTraitAdjustable
    open func accessibilityIncrement()
    open func accessibilityDecrement()
}

// Views That Increment and Decrement

extension NSObject {
    // Must have the UIAccessibilityTraitAdjustable
    open func accessibilityIncrement()
    open func accessibilityDecrement()
}

// Increment and Decrement Example

class StarsView : UIView {
    var numStars: Int = 0 {
        didSet {
            // Code for adjusting the star rating
        }
    }

    override var accessibilityTraits {
        get { return super.accessibilityTraits | UIAccessibilityTraitAdjustable } 
        set { } 
    }

    override func accessibilityIncrement() {
        numStars += 1
    }

    override func accessibilityDecrement() {
        numStars -= 1
    }
}

// Increment and Decrement Example

class StarsView : UIView {

    var numStars: Int = 0 {
        didSet {
            // Code for adjusting the star rating
        }
    }

    override var accessibilityTraits {
        get { return super.accessibilityTraits | UIAccessibilityTraitAdjustable } 
        set { }
    }

    override func accessibilityIncrement() {
        numStars += 1
    }

    override func accessibilityDecrement() {
        numStars -= 1
    }
}

// Increment and Decrement Example

class StarsView : UIView {
    var numStars: Int = 0 {
didSet {
        // Code for adjusting the star rating
    }
}

    override var accessibilityTraits {
        get { return super.accessibilityTraits | UIAccessibilityTraitAdjustable }  
        set { } 
    }

    override func accessibilityIncrement() {
        numStars += 1
    }

    override func accessibilityDecrement() {
        numStars -= 1
    }
}

// Increment and Decrement Example

class StarsView : UIView {
    var numStars: Int = 0 {
        didSet {
            // Code for adjusting the star rating
        }
    }

    override var accessibilityTraits {
        get { return super.accessibilityTraits | UIAccessibilityTraitAdjustable } 
        set { } 
    }

    override func accessibilityIncrement() {
        numStars += 1
    }

    override func accessibilityDecrement() {
        numStars -= 1
    }
}
// Increment and Decrement Example
class StarsView : UIView {
    var numStars: Int = 0 {

        didSet {
            // Code for adjusting the star rating
        }
    }

    override var accessibilityTraits {
        get { return super.accessibilityTraits | UIAccessibilityTraitAdjustable } 
        set { } 
    }

    override func accessibilityIncrement() {
        numStars += 1
    }

    override func accessibilityDecrement() {
        numStars -= 1
    }
}
Pass Through
// Activation Point

extension NSObject {
    open var accessibilityActivationPoint: CGPoint
}

// Activation Point

extension NSObject {
    open var accessibilityActivationPoint: CGPoint
}

// Activation Point Example

class SliderView: UIView {

    let sliderNub: NubView

    override var accessibilityActivationPoint: CGPoint {
        get {
            return sliderNub.center
        }
        set { }
    }
}

// Activation Point Example

class SliderView: UIView {

    let sliderNub: NubView

    override var accessibilityActivationPoint: CGPoint {
        get {
            return sliderNub.center
        }
        set { }
    }

}
// Activation Point Example

class SliderView: UIView {

    let sliderNub: NubView

    override var accessibilityActivationPoint: CGPoint {
        get {
            return sliderNub.center
        }
        set {
        }
    }
}

Custom Scrolling
// Accessibility Scroll

extension NSObject {
    // Initiated with a 3-finger swipe
    open func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool
}

public enum UIAccessibilityScrollDirection : Int {
    case right
    case left
    case up
    case down
    case next
    case previous
}
// Accessibility Scroll

extension NSObject {
    // Initiated with a 3-finger swipe
    open func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool
}

public enum UIAccessibilityScrollDirection : Int {
    case right
    case left
    case up
    case down
    case next
    case previous
}
/ Accessibility Scroll

extension NSObject {
    // Initiated with a 3-finger swipe
    open func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool
}

public enum UIAccessibilityScrollDirection : Int {
    case right
    case left
    case up
    case down
    case next
    case previous
}
// Accessibility Scroll Example

class MyImageView : UIImageView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && scrollDirection == .down) {
            showDetails()
            return true
        }
        return false
    }
}

// Accessibility Scroll Example

class MyImageView : UIImageView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && scrollDirection == .down) {
            showDetails()
            return true
        }
        return false
    }
}

// Accessibility Scroll Example

class MyImageView : UIImageView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && scrollDirection == .down) {
            showDetails()
            return true
        }
        return false
    }
}

// Accessibility Scroll Example

class MyImageView : UIImageView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && scrollDirection == .down) {
            showDetails()
            return true
        }
        return false
    }
}
// Accessibility Scroll Example

class MyImageView: UIImageView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && scrollDirection == .down) {
            showDetails()
            return true
        }
        return false
    }
}


// Accessibility Scroll Example

class MyImageView : UIView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && direction == .down) {
            showDetails()
            return true
        }
        return false
    }
}
// Accessibility Scroll Example

class MyImageView : UIImageView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && scrollDirection == .down) {
            showDetails()
            return true
        }
        return false
    }
}

// Accessibility Scroll Example
class MyImageView : UIImageView {

    var canShowDetails: Bool = false

    func showDetails() {
        // Code for showing details
    }

    override func accessibilityScroll(_ direction: UIAccessibilityScrollDirection) -> Bool {
        if (canShowDetails && scrollDirection == .down) {
            showDetails()
            return true
        }
        return false
    }

    return false
}
}
More Information

Drag and Drop Accessibility
Thailand

I love this photo of him touching into the pond!
Drag Sources
extension NSObject {
    open var accessibilityDragSourceDescriptors: [UIAccessibilityLocationDescriptor]?
    open var accessibilityDropPointDescriptors: [UIAccessibilityLocationDescriptor]?
}

open class UIAccessibilityLocationDescriptor : NSObject {
    open var view: UIView
    open var point: CGPoint
    open var name: String
    open var attributedName: NSAttributedString

    public convenience init(name: String, view: UIView)
    public convenience init(name: String, point: CGPoint, in view: UIView)
    public init(attributedName: NSAttributedString, point: CGPoint, in view: UIView)
}
extension NSObject {
    open var accessibilityDragSourceDescriptors: [UIAccessibilityLocationDescriptor]?
    open var accessibilityDropPointDescriptors: [UIAccessibilityLocationDescriptor]?
}

open class UIAccessibilityLocationDescriptor : NSObject {
    open var view: UIView
    open var point: CGPoint
    open var name: String
    open var attributedName: NSAttributedString

    public convenience init(name: String, view: UIView)
    public convenience init(name: String, point: CGPoint, in view: UIView)
    public init(attributedName: NSAttributedString, point: CGPoint, in view: UIView)
}
extension NSObject {
    open var accessibilityDragSourceDescriptors: [UIAccessibilityLocationDescriptor]?
    open var accessibilityDropPointDescriptors: [UIAccessibilityLocationDescriptor]?
}

open class UIAccessibilityLocationDescriptor : NSObject {
    open var view: UIView
    open var point: CGPoint
    open var name: String
    open var attributedName: NSAttributedString

    public convenience init(name: String, view: UIView)
    public convenience init(name: String, point: CGPoint, in view: UIView)
    public init(attributedName: NSAttributedString, point: CGPoint, in view: UIView)
}
// Drag and Drop Example

class HomeView: UIView {

    func appView(at IndexPath: IndexPath) -> AppView {

        let appView = AppView()

        // Initialize app view

        let dragDescriptor = UIAccessibilityLocationDescriptor(name: "Drag \(appView.name)", point: appView.center, view: self)
        appView.accessibilityDragSourceDescriptors = [dragDescriptor]

        // Continued on next slide...
    }
}
// Drag and Drop Example

class HomeView: UIView {

    func appView(at IndexPath: IndexPath) -> AppView {

        let appView = AppView()

        // Initialize app view

        let dragDescriptor = UIAccessibilityLocationDescriptor(name: "Drag \(appView.name)", point: appView.center, view: self)
        appView.accessibilityDragSourceDescriptors = [dragDescriptor]

        // Continued on next slide...
    }
}
// Drag and Drop Example

class HomeView: UIView {

    func appView(at IndexPath: IndexPath) -> AppView {

        let appView = AppView()

        // Initialize app view

        let dragDescriptor = UIAccessibilityLocationDescriptor(name: "Drag \(appView.name)",
                                                                  point: appView.center, view: self)
        appView.accessibilityDragSourceDescriptors = [dragDescriptor]

        // Continued on next slide...
    }
}
class HomeView: UIView {

    func appView(at IndexPath: IndexPath) -> AppView {

        let appView = AppView()

        // Initialize app view

        let dragDescriptor = UIAccessibilityLocationDescriptor(name: "Drag \(appView.name)", point: appView.center, view: self)

        appView.accessibilityDragSourceDescriptors = [dragDescriptor]

        // Continued on next slide...
// Drag and Drop Example

class HomeView: UIView {

    func appView(at IndexPath: IndexPath) -> AppView {

        let appView = AppView()

        // Initialize app view

        let dragDescriptor = UIAccessibilityLocationDescriptor(name: "Drag \(appView.name)",
                                                                 point: appView.center, view: self)

        appView.accessibilityDragSourceDescriptors = [dragDescriptor]

        // Continued on next slide...
let leftPoint = CGPoint(x: appView.frame.origin.x, y: appView.center.y)
let rightPoint = CGPoint(x: appView.frame.origin.x + appView.frame.size.width,
    y: appView.center.y)

let leftDescriptor = UIAccessibilityLocationDescriptor(name: "Drop left of 
\(appView.name)" , point: leftPoint, view: self)
let folderDescriptor = UIAccessibilityLocationDescriptor(name: “Create folder with 
\(appView.name)” , point: appView.center, view: self)
let rightDescriptor = UIAccessibilityLocationDescriptor(name: “Drop right of 
\(appView.name)” , point, rightPoint, view: self)

appView.accessibilityDropPointDescriptors = [leftDescriptor, folderDescriptor, 
    rightDescriptor]

return appView
let leftPoint = CGPoint(x: appView.frame.origin.x, y: appView.center.y)
let rightPoint = CGPoint(x: appView.frame.origin.x + appView.frame.size.width, y: appView.center.y)

let leftDescriptor = UIAccessibilityLocationDescriptor(name: "Drop left of \(appView.name)", point: leftPoint, view: self)
let folderDescriptor = UIAccessibilityLocationDescriptor(name: "Create folder with \(appView.name)", point: appView.center, view: self)
let rightDescriptor = UIAccessibilityLocationDescriptor(name: "Drop right of \(appView.name)", point: rightPoint, view: self)

appView.accessibilityDropPointDescriptors = [leftDescriptor, folderDescriptor, rightDescriptor]

return appView
let leftPoint = CGPoint(x: appView.frame.origin.x, y: appView.center.y)
let rightPoint = CGPoint(x: appView.frame.origin.x + appView.frame.size.width, y: appView.center.y)

let leftDescriptor = UIAccessibilityLocationDescriptor(name: "Drop left of \(appView.name)\", point: leftPoint, view: self)
let folderDescriptor = UIAccessibilityLocationDescriptor(name: "Create folder with \(appView.name)\", point: appView.center, view: self)
let rightDescriptor = UIAccessibilityLocationDescriptor(name: "Drop right of \(appView.name)\", point: rightPoint, view: self)

appView.accessibilityDropPointDescriptors = [leftDescriptor, folderDescriptor, rightDescriptor]

return appView
}
// Drag and Drop Example, continued

let leftPoint = CGPoint(x: appView.frame.origin.x, y: appView.center.y)
let rightPoint = CGPoint(x: appView.frame.origin.x + appView.frame.size.width,
                        y: appView.center.y)

let leftDescriptor = UIAccessibilityLocationDescriptor(name: "Drop left of 
(appView.name)", point: leftPoint, view: self)

let folderDescriptor = UIAccessibilityLocationDescriptor(name: "Create folder with 
(appView.name)", point: appView.center, view: self)

let rightDescriptor = UIAccessibilityLocationDescriptor(name: "Drop right of 
(appView.name)", point: rightPoint, view: self)

appView.accessibilityDropPointDescriptors = [leftDescriptor, folderDescriptor, 
rightDescriptor]

return appView

}
// Drag and Drop Example, continued

let leftPoint = CGPoint(x: appView.frame.origin.x, y: appView.center.y)
let rightPoint = CGPoint(x: appView.frame.origin.x + appView.frame.size.width, y: appView.center.y)

let leftDescriptor = UIAccessibilityLocationDescriptor(name: "Drop left of \(appView.name)", point: leftPoint, view: self)
let folderDescriptor = UIAccessibilityLocationDescriptor(name: "Create folder with \(appView.name)", point: appView.center, view: self)
let rightDescriptor = UIAccessibilityLocationDescriptor(name: "Drop right of \(appView.name)", point: rightPoint, view: self)

appView.accessibilityDropPointDescriptors = [leftDescriptor, folderDescriptor, rightDescriptor]

return appView
let leftPoint = CGPoint(x: appView.frame.origin.x, y: appView.center.y)
let rightPoint = CGPoint(x: appView.frame.origin.x + appView.frame.size.width, y: appView.center.y)

let leftDescriptor = UIAccessibilityLocationDescriptor(name: "Drop left of \(appView.name)", point: leftPoint, view: self)
let folderDescriptor = UIAccessibilityLocationDescriptor(name: "Create folder with \(appView.name)", point: appView.center, view: self)
let rightDescriptor = UIAccessibilityLocationDescriptor(name: "Drop right of \(appView.name)", point: rightPoint, view: self)

appView.accessibilityDropPointDescriptors = [leftDescriptor, folderDescriptor, rightDescriptor]

return appView
}
let leftPoint = CGPoint(x: appView.frame.origin.x, y: appView.center.y)
let rightPoint = CGPoint(x: appView.frame.origin.x + appView.frame.size.width, y: appView.center.y)

let leftDescriptor = UIAccessibilityLocationDescriptor(name: "Drop left of \(appView.name)", point: leftPoint, view: self)
let folderDescriptor = UIAccessibilityLocationDescriptor(name: "Create folder with \(appView.name)", point: appView.center, view: self)
let rightDescriptor = UIAccessibilityLocationDescriptor(name: "Drop right of \(appView.name)", point: rightPoint, view: self)

appView.accessibilityDropPointDescriptors = [leftDescriptor, folderDescriptor, rightDescriptor]

return appView
}
Summary

New features
Auditing
App accessibility
People
Thank you!
More Information

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Media and Gaming Accessibility</td>
<td>Grand Ballroom A</td>
<td>Wednesday 3:10PM</td>
</tr>
<tr>
<td>Design For Everyone</td>
<td>Hall 3</td>
<td>Thursday 11:00AM</td>
</tr>
<tr>
<td>Auto Layout Techniques in Interface Builder</td>
<td>Hall 3</td>
<td>Friday 9:00AM</td>
</tr>
<tr>
<td>Building Apps with Dynamic Type</td>
<td>Executive Ballroom</td>
<td>Friday 1:50PM</td>
</tr>
<tr>
<td>Lab</td>
<td>Location</td>
<td>Time</td>
</tr>
<tr>
<td>-------------------------------------------------------</td>
<td>-----------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>Accessibility Lab</td>
<td>Technology Lab J</td>
<td>Wed 4:10PM–6:00PM</td>
</tr>
<tr>
<td>Accessibility and Dynamic Type Lab</td>
<td>Technology Lab C</td>
<td>Fri 2:30PM–4:00PM</td>
</tr>
<tr>
<td>Accessibility Design by Appointment Lab</td>
<td>User Interface Design Lab B</td>
<td>Thur 9:00AM–6:00PM</td>
</tr>
<tr>
<td>Accessibility Design by Appointment Lab</td>
<td>User Interface Design Lab B</td>
<td>Fri 9:00AM–1:15PM</td>
</tr>
<tr>
<td>Other Events</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Accessible Technology and Inclusive Design Get-Together</strong></td>
<td>The Hub</td>
<td>Wednesday 6:30PM</td>
</tr>
</tbody>
</table>