VoiceOver API の基本

宇佐見公輔

2025-03-21 / YUMEMI.grow Mobile #20

株式会社ゆめみ

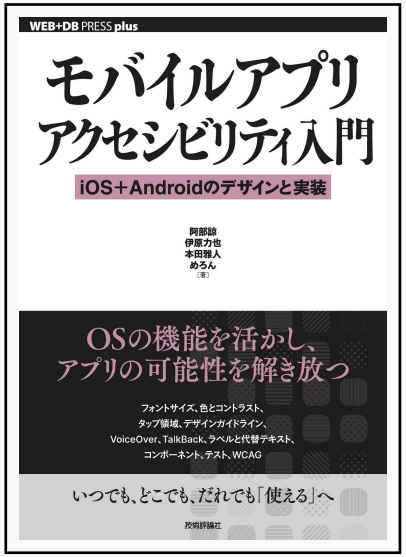
自己紹介

- 宇佐見公輔（うさみこうすけ）
  - 株式会社ゆめみ　iOS テックリード

近況

- 引き続き「モバイルアプリアクセシビリティ入門」読書会に参加中

https://gihyo.jp/book/2024/978-4-297-14602-3



月刊 I/O に記事が掲載されました

- 月刊 I/O 2025 年 4 月号
- 「iOS 18 の視線トラッキング」を書きました

https://www.kohgakusha.co.jp/books/detail/4852

アクセシビリティと VoiceOver

iOS のアクセシビリティ

視覚サポート
- 読み上げ（VoiceOver）
- 文字サイズ変更
- ダークモード　など

音声サポート
- 音声合成　など

身体サポート
- 視線トラッキング
- 音声コントロール
- スイッチコントロール　など

認知サポート
- アシスティブアクセス　など

聴覚サポート
- 字幕　など

VoiceOver

- iOS で UI の要素を読み上げる機能
  - Android の TalkBack に相当
- 全盲など視覚障害があってもアプリを利用できる
  - なお、弱視などある程度見える人も利用している
- アプリ開発時のアクセシビリティ対応で行う機能の代表格
  - 多くの人はこの UI 読み上げ機能を思い浮かべる
  - 本当はアクセシビリティ対応っていろいろあるけど

VoiceOver 対応の基本

デフォルト動作

- デフォルトで、ある程度適切に読み上げられる
  - 標準コンポーネントの強み
  - この点は、Web とモバイルアプリの違いのひとつ
- ただし、いくつかの場合に実装が必要
  - 画像のみのコンポーネント
  - カスタムコンポーネント　など

accessibilityLabel

デフォルトの読み上げでうまくいかない場合に利用する。

```
var body: some View {
    Button(...) {
        // 画像のみ（テキストがない）
        Image(systemName: "heart")
    }
    // 指定しないと画像リソース名が読み上げられてしまう
    .accessibilityLabel("いいね")
}
```

accessibilityTraits

UI 要素の種類や性質を指定する（カスタムコンポーネント向け）。

- 種類
  - `button`（「ボタン」と読み上げてくれる）など
- 状態
  - `selected`（「選択中」と読み上げてくれる）など
- 性質
  - `header`（「見出し」とみなす）など

SwiftUI と UIKit

基本的に、SwiftUI と UIKit のどちらでも同様のことができるが、API は異なる（主に、モディファイアとプロパティの違い）。

```
// SwiftUI
CustomButton(...)
.accessibilityAddTraits(.isButton)

// UIKit
customButton.accessibilityTraits = [.button]
```

他の VoiceOver API

基本 API

前述の 2 つでほとんどの場合は問題ない。

- `accessibilityLabel`
- `accessibilityTraits`

一方で、VoiceOver 関連は API がたくさんある。

知っておくと有益な場合もあるかも。

ヒント情報を追加する

読み上げ内容は基本的に `accessibilityLabel` で与えるが、追加の情報を `accessibilityHint` で与えることができる。

```
Button(...)
.accessibilityLabel("削除")
.accessibilityHint("このアイテムを削除します")
```

- 基本的に、ラベルだけで分かるようにする
- より詳しい説明をヒントで与える
  - ヒントは読み上げられない場合もある

読み上げ順序を制御する（SwiftUI）

読み上げ順序を指定するには、SwiftUI では、
accessibilitySortPriority を使う。

```
// SwiftUI
Text("後に読み上げ（優先度が低い）")
.accessibilitySortPriority(10)

Text("先に読み上げ（優先度が高い）")
.accessibilitySortPriority(20)
```

読み上げ順序を制御する（UIKit）

UIKit では、accessibilityElements を使う。

```
// UIKit
override var accessibilityElements: [Any]? {
    get {
        [firstElement, secondElement]
    }
}
```

能動的に通知する

重要な通知、モーダル表示やレイアウト変更を伝えたい、など。

```
// SwiftUI
AccessibilityNotification
    .Announcement("メッセージ").post()

// UIKit
UIAccessibility.post(
    notification: .announcement,
    argument: "メッセージ")
```

読み上げ音声の調整

読み上げ音声のピッチや言語を調整できる。

- `UIKit : accessibilityAttributedLabel`
- `SwiftUI : accessibilityLabel(Text())`

`AttributedString` にはアクセシビリティ関連の要素がある。

- `accessibilitySpeechPitch`
- `accessibilitySpeechLanguage`
- `accessibilitySpeechSpellOut`

リファレンス

Apple Developer Documentation の次の場所にある。

フレームワーク

- Accessibility

SwiftUI

- Accessibility modifiers

UIKit

- Accessibility for UIKit
- UIAccessibility

リファレンスの特徴

ドキュメントとして充実している。

- 全体的な説明文がわりとちゃんと書かれている
- アクセシビリティ関連でまとめられているので、VoiceOver 以外も含めてたくさんある

ただ、多いので読むのが大変。リファレンスの細かい項目は説明が少ないものもある。

- 関連するデータ構造などが多い
- API の細かい要素になってくると記述がなくなる傾向はある

VoiceOver 対応の整理

VoiceOver 対応の整理

実のところ、一番大変なのは API の問題ではなくて、実際に読み上げてみて、どう読まれるかを確認すること。そこは頑張るしかない。

一方、API はたくさんあるものの、基本 API はふたつ。それ以外を必要に応じて少しずつ追加対応していくとやりやすい。

- `accessibilityLabel`
- `accessibilityTraits`