

Accessibility with UI Test

접근성 지원 시작하기

2021.8.16 Delma(송다영)

목차

Part 1

- 접근성이란
- 접근성 지원이란
- 필요성

Part 2

- 접근성 측정
 - Accessibility Inspector
- 접근성 개선하기

Part 3

- 접근성과 UITest

접근성이란?

접근하기 쉬운 성질로
교통, 제품, 디자인 등의 환경을
가능한 한 많은 사용자가 불편함 없이 이용할 수 있는 정도를 말한다.

접근성 지원이란?

접근이 가능하도록 지원하는 것으로
장애인을 위해 장애인용 기능을 지원하는 것이 아닌 모든 사람
들이 기능(앱)을 편하게 쓸 수 있도록 하는 것을 말한다.



Vision



Hearing



Mobility



Cognitive

애플의 대표적인 접근성 지원 기능들

시각

- 텍스트 크기를 확대/축소하는 기능
- 문자를 읽어주는 콘텐츠 말하기 기능
- 화면 요소의 움직임을 줄이는 동작 줄이기 기능

애플의 대표적인 접근성 지원 기능들

청각

- 화재 경보, 초인종등의 소리를 감지해 화면에 표시해주는 소리 인식 기능
- 비언어적인 의사소통까지 텍스트로 옮겨놓은 청각 장애인용 자막 지원

애플의 대표적인 접근성 지원 기능들

운동능력

- 받아쓰기 기능
- 자동완성 텍스트
- 뒷면 탭 기능

애플의 대표적인 접근성 지원 기능들

인지능력

- Safari 읽기 도구 - 웹에서 나오는 광고나 애니메이션 등의 방해 요소를 줄이고 콘텐츠에만 집중하도록 도움
- 기기가 한 번에 하나의 앱만 실행하거나 화면의 특정 부분에 대한 터치 입력을 제한하는 사용법 유도 기능

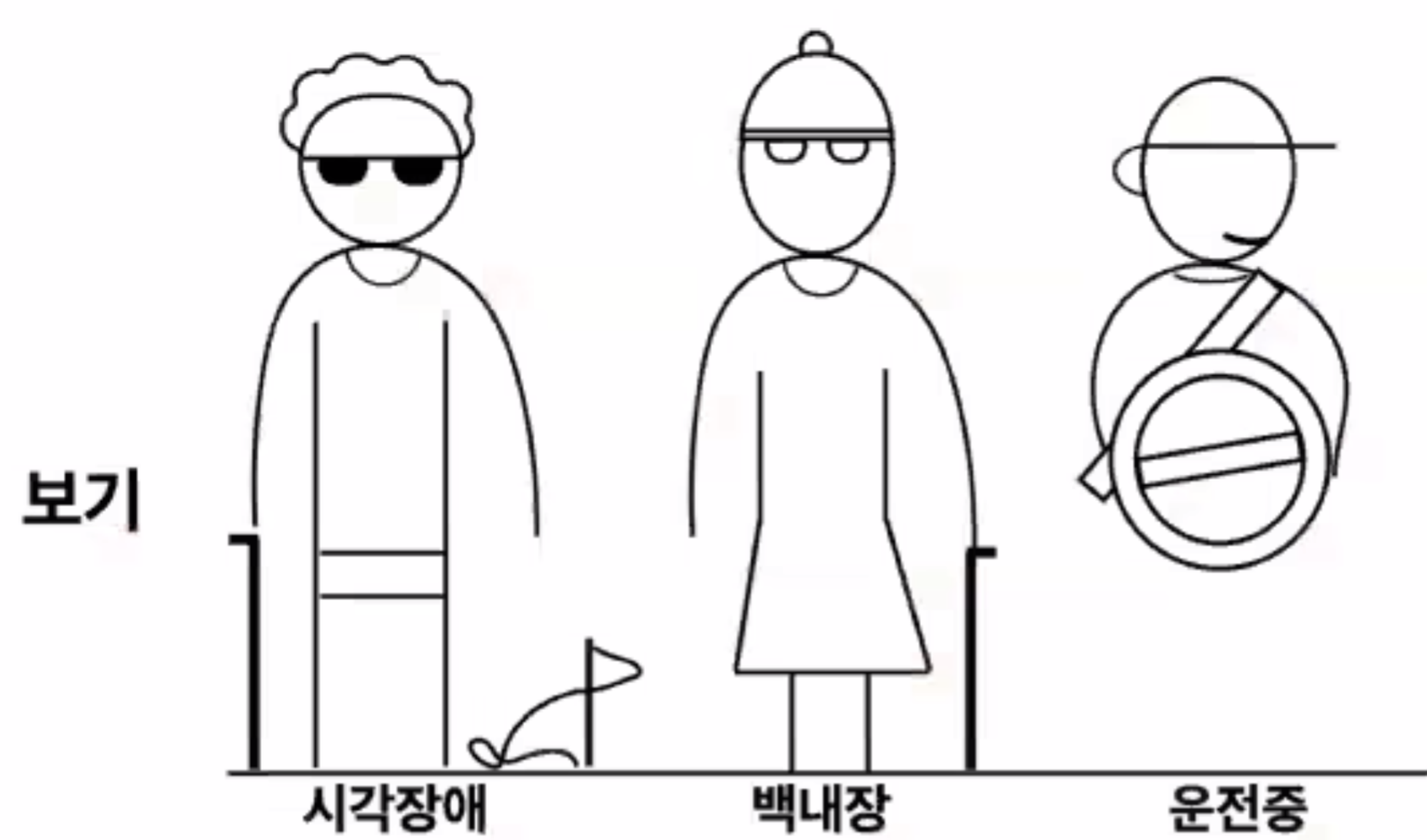
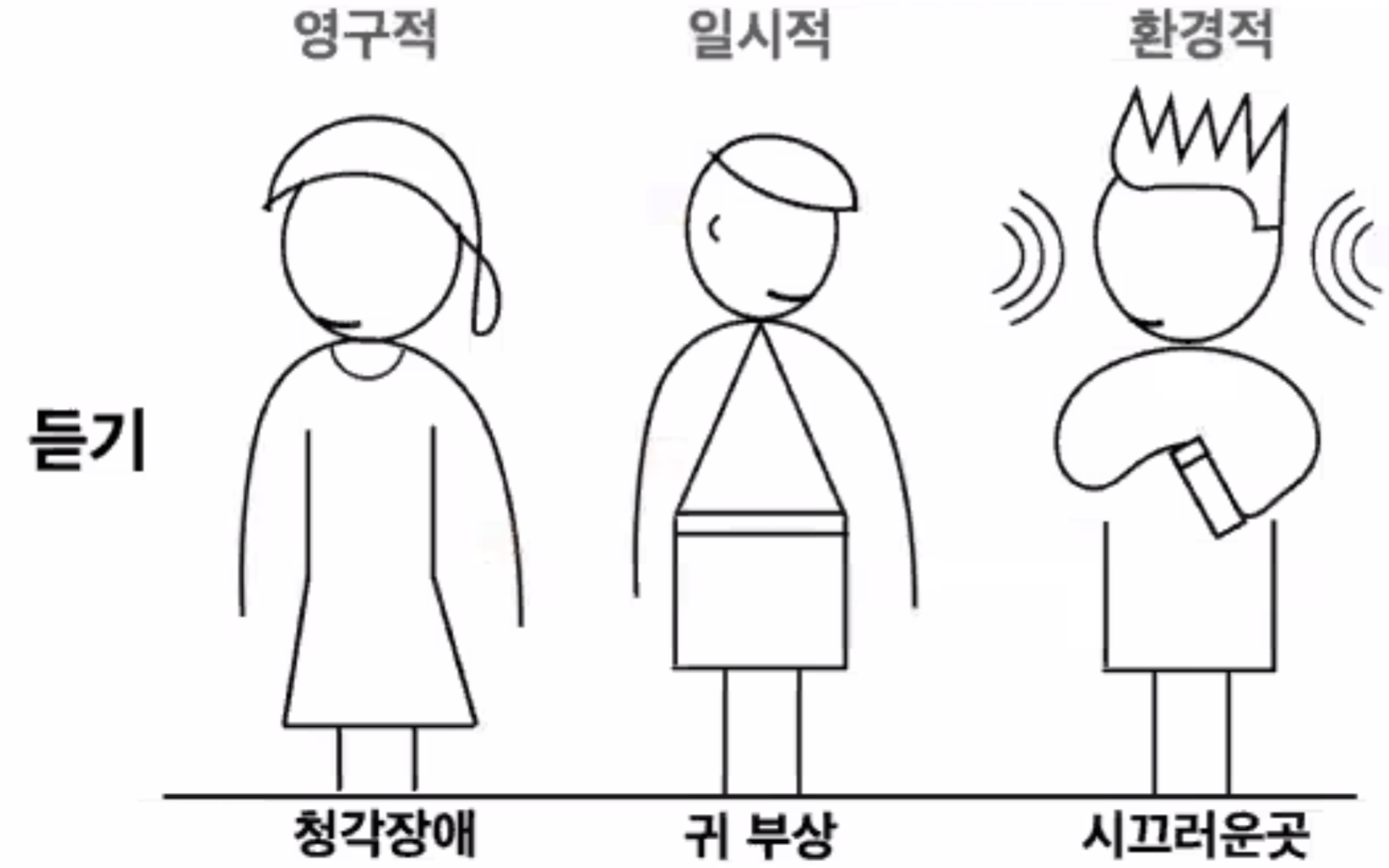
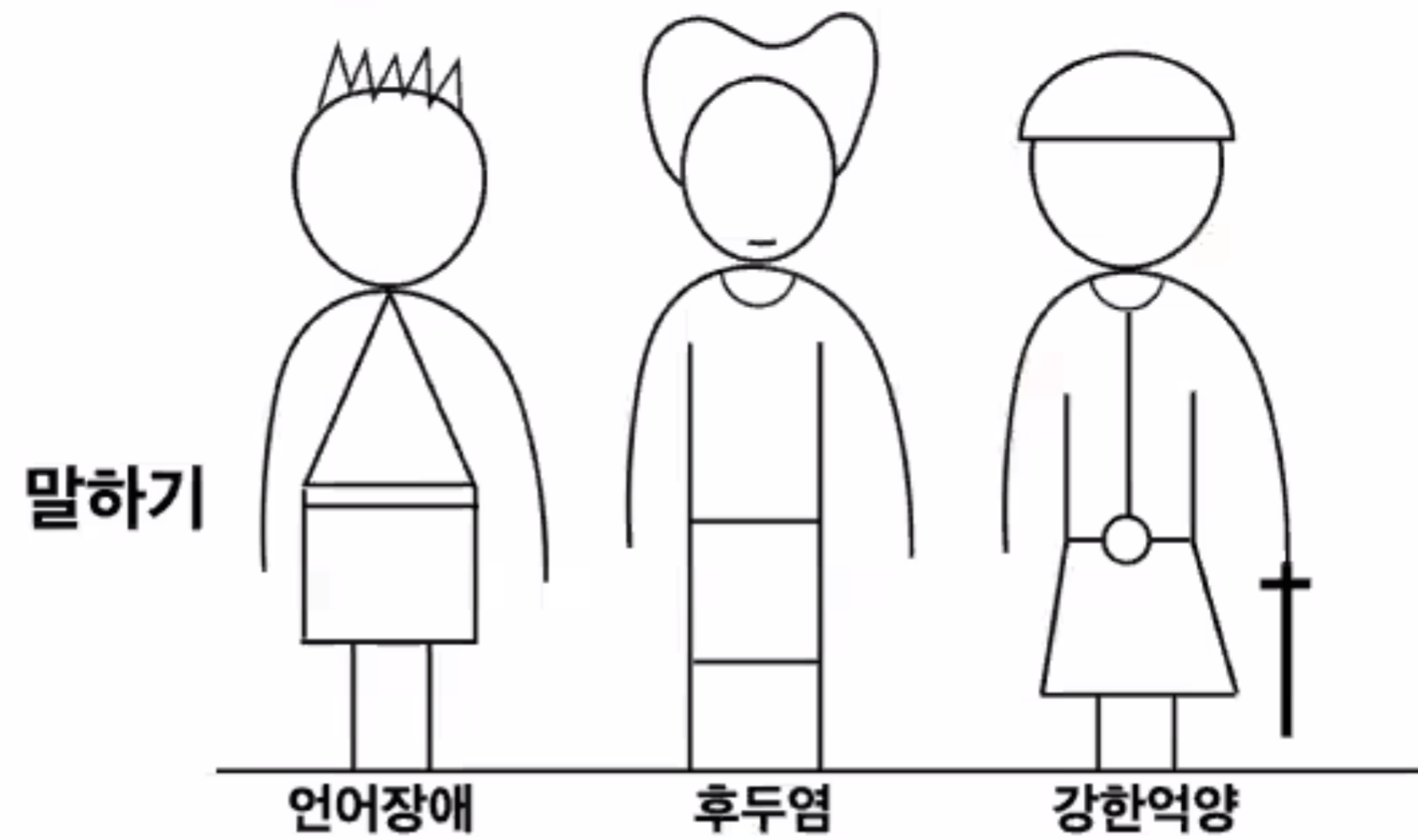
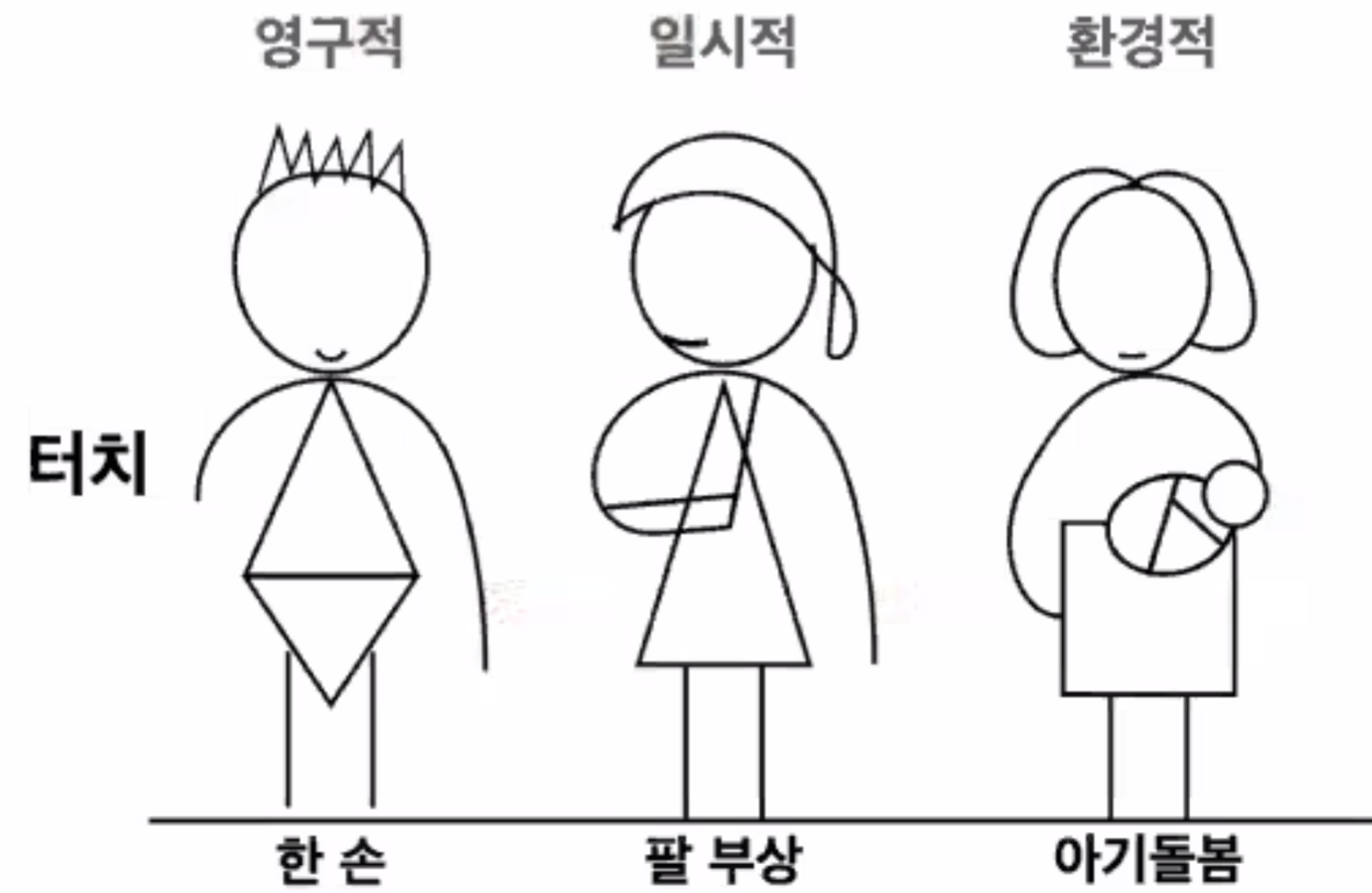
왜 접근성 지원을 해야할까?

애플의 의도대로
API를 사용하도록 도와줍니다

퍼포먼스에 더 신경을 쓰도록 도와줍니다

일관성 있는 UI를 만들 수 있습니다

모든 사람이 사용할 수 있습니다



UI Test가 쉬워지게 합니다

그럼 접근성 측정은 어떻게?

Demo

“알아듣기 어려워..”

Accessibility Label 지정

- 각 요소에 Accessibility Label을 지정합니다
- Label, Button 등의 UI 요소의 타입은 자동으로 읽어주기 때문에 이름에 포함하지 않습니다

✗ `button.accessibilityLabel = "Add button"`

○ `button.accessibilityLabel = "Add"`

Accessibility Label 지정

- 이해하기 쉽고도 구체적이게 이름을 짓습니다
- 장황하거나 반복된 설명을 피합니다

✗ "item from the current folder and add it to the trash"

○ "Delete"

✗ "Previous song", "Play song", "Next song"

○ "Previous", "Play", "Next"

Accessibility Label 지정

- UI가 변경될 때 라벨도 함께 변경합니다
- 의미있는 애니메이션에는 라벨도 추가합니다

```
spinner.accessibilityLabel = "Loading"
```

Voice Over 지원

isAccessibilityElement

- 접근성 요소인지를 지정하는 속성으로 기본값은 false
- true로 설정하면 보이스 오버로 읽을 수 있음
- 주로 불필요한 내용을 읽지 않게 하기 위해 사용

Voice Over 지원

accessibilityHint

- 요소에 대한 추가 설명이나, 동작하기 위한 방법을 안내함

Voice Over 지원

accessibilityValue

- 요소의 현재 값을 나타내는 문자열
- 값이 달라지는 요소(텍스트 필드, 슬라이더 등)에 사용함
- accessibilityLabel은 요소에 대한 설명, accessibilityValue는 값에 대한 설명으로 각각 설정이 필요함

```
let slider = UISlider()  
slider.accessibilityLabel = "볼륨"  
slider.accessibilityValue = "30%"
```

Voice Over 지원

accessibilityTraits

- 접근성 요소를 가장 잘 특성화하는 특징을 지정함
- 요소가 UIKit Control이 아닌 경우 none으로 지정할 수 있음
- Button, Image, Static Text, Link 등이 있음

Voice Over 지원

accessibilityIdentifier

- UI Element를 구분하는 문자열
- UI Recording이나 UI Test시 각각의 요소를 구분하는 데 유용하게 사용됨
- 이 id를 이용하면 accessibilityLabel에 부적절하게 접근하는 것을 방지할 수 있음

Voice Over 지원

Accessibility

Accessibility ☐ Enabled

Label

Hint

Identifier

- Traits
- ☐ Button
 - ☐ Image
 - ☐ Static Text
 - ☐ Search Field
 - ☐ Plays Sound
 - ☐ Keyboard Key
 - ☐ Summary Element
 - ☒ User Interaction Enabled
 - ☐ Updates Frequently
 - ☐ Starts Media Session
 - ☐ Adjustable
 - ☐ Allows Direct Interaction
 - ☐ Causes Page Turn
 - ☐ Header
- ☐ Link
- ☐ Selected

`URLRequest.allowsConstrainedNetworkAccess`

`URLError.networkUnavailableReason == .constrained`

저데이터 모드 지원

- 이미지 요청시 저해상도로 요청하거나 아예 요청하지 않고 플레이스 홀더 이미지를 사용
- 사용자가 명시적으로 요청하지 않은 네트워크를 제한함
- prefetch를 비활성화 함

Color and Shapes - 버튼 모양

- 버튼이 단순한 텍스트 형태가 아닌 배경과 모양을 가지도록 해 버튼으로서 인식이 되도록 함
- 탭바에서 활성화 된 탭에 배경색을 더해 가시성을 높임

• **시각적으로 보완하기**
설정 > 권한 > 쉬운 사용 > 디스플레이 및 텍스트 크기 > 버튼 모양 옵션을 활성화함

```
extension UIAccessibility {  
    @available(iOS 14.0, tvOS 14.0)  
    static var buttonShapesEnabled: Bool { get }  
  
    @available(iOS 14.0, tvOS 14.0)  
    static let buttonShapesEnabledStatusDidChangeNotification: NSNotification.Name  
}
```

최근 재생한 항목

[모두 보기](#)



.... 수요일
323회 - 47은이, 58속이?
송은이 김숙의 비밀보장 323회- 47...



323회 - 47은이, 58속이?



지금 듣기



둘러보기



보관함










검색

시각적으로 보완하기

Color and Shapes - 색상 없는 차별화

- 색맹은 색상만으로 의미 전달하는 것을 구별할 수 없음
- 색상이 아닌 심볼로 의미를 전달하도록 함
- 설정 > 손쉬운 사용 > 디스플레이 및 텍스트 크기 > 색상 사용 없이 구별 옵션을 활성화함

시각 지원

	VoiceOver	끔 >
	확대/축소	끔 >
	확대기	끔 >
	디스플레이 및 텍스트 크기	>
	동작	>
	콘텐츠 말하기	>
	오디오 설명	끔 >

```
extension UIAccessibility {
    @available(iOS 13.0, tvOS 13.0)
    static var shouldDifferentiateWithoutColor: Bool { get }

    @available(iOS 13.0, tvOS 13.0)
    static let shouldDifferentiateWithoutColorDidChangeNotification: NSNotification.Name
}
```

시각적으로 보완하기

Color and Shapes - 대비 증가

- 색약이거나 시력 자체가 낮은 사람은 낮은 대비의 글을 읽기 어려움
- 시스템 컬러의 경우 자동 지원됨
- 커스텀 컬러의 경우 에셋 카탈로그에서 “High Contrast” 옵션을 켜고 대안 컬러들을 지정함
- 설정 > 손쉬운 사용 > 디스플레이 및 텍스트 크기 > 대비 증가 옵션을 활성화함

시각적으로 보완하기

Text Readability - 큰 텍스트를 고려한 디자인

```
// ZodiacConstellationCell.swift

override func traitCollectionDidChange (_
previousTraitCollection: UITraitCollection?) {

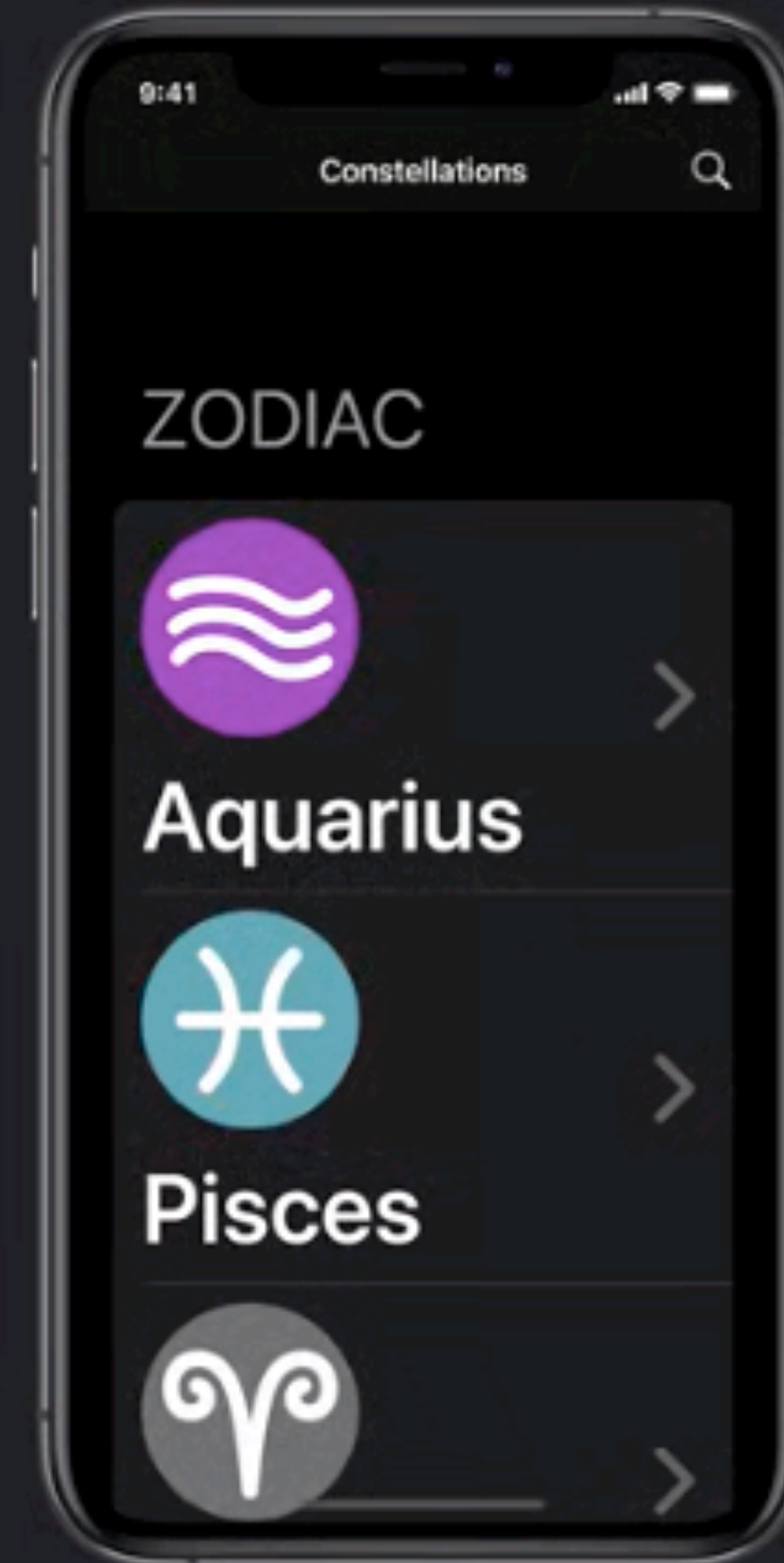
    if (traitCollection.preferredContentSizeCategory
        < .accessibilityMedium) { // Default font sizes

        stackView.axis = .horizontal
        stackView.alignment = .center

    } else { // Accessibility font sizes

        stackView.axis = .vertical
        stackView.alignment = .leading

    }
}
```



- UI Testing Bundle
- UI Elements에 대한 접근성 정보

• **UI Test** OS X 10.11 이상

- iOS 9 이상

UI Test with Accessibility

- UI에 대한 풍부한 semantic(의미)를 제공
- 사용자가 사용하는 것처럼 UI Test를 진행할 수 있음
- API에 대한 미세한 조정이 가능함
- UI Test는 분리된 프로세스 내에서 실행이 가능함

UI Testing API

XCUIApplication

```
graph TD; XCUIApplication[XCUIApplication] --> XCUIElement[XCUIElement]; XCUIApplication --> XCUIElementQuery[XCUIElementQuery];
```

XCUIElement

XCUIElementQuery

XCUIApplication

- 테스트 할 앱을 실행하거나 종료할 수 있는 앱의 프록시
- 새 프로세스를 생성하기 위해 `launch()`를 호출
- 테스트 elements를 찾기위한 시작점

```
let app = XCUIApplication()  
app.launch()  
app.terminate()
```

XCUIElement

- XCUIApplication과 마찬가지로 테스트 할 앱의 UI Element에 대한 프록시
- Button, Cell, Window등의 타입을 가짐
- Accessibility identifier, label, title등의 레이블을 가짐

XCUIElementQuery

- 구체적인 UI elements를 찾기 위한 API
- element의 계층 구조 간의 관계와 필터를 이용해 element를 특정하는 방식으로 동작

Subscripting	<code>table.staticTexts["Groceries"]</code>
Index	<code>table.staticTexts.elementAtIndex(0)</code>
Unique	<code>app.navigationBars.element</code>

XCUElementQuery

```
// 버튼 하나를 반환. 일치하는 요소가 여러개 있는 경우 테스트가 실패함
app.buttons.element

// 일치하는 요소가 여러개 있더라도 첫 번째로 매치되는 버튼 하나를 반환함
app.buttons.firstMatch

// 버튼의 title이 "submit"인 버튼을 반환
app.buttons["submit"]

// 스크롤뷰의 모든 버튼을 반환(직접적인 하위 뷰만 읽음)
app.scrollViews["Main"].children(matching: .button)

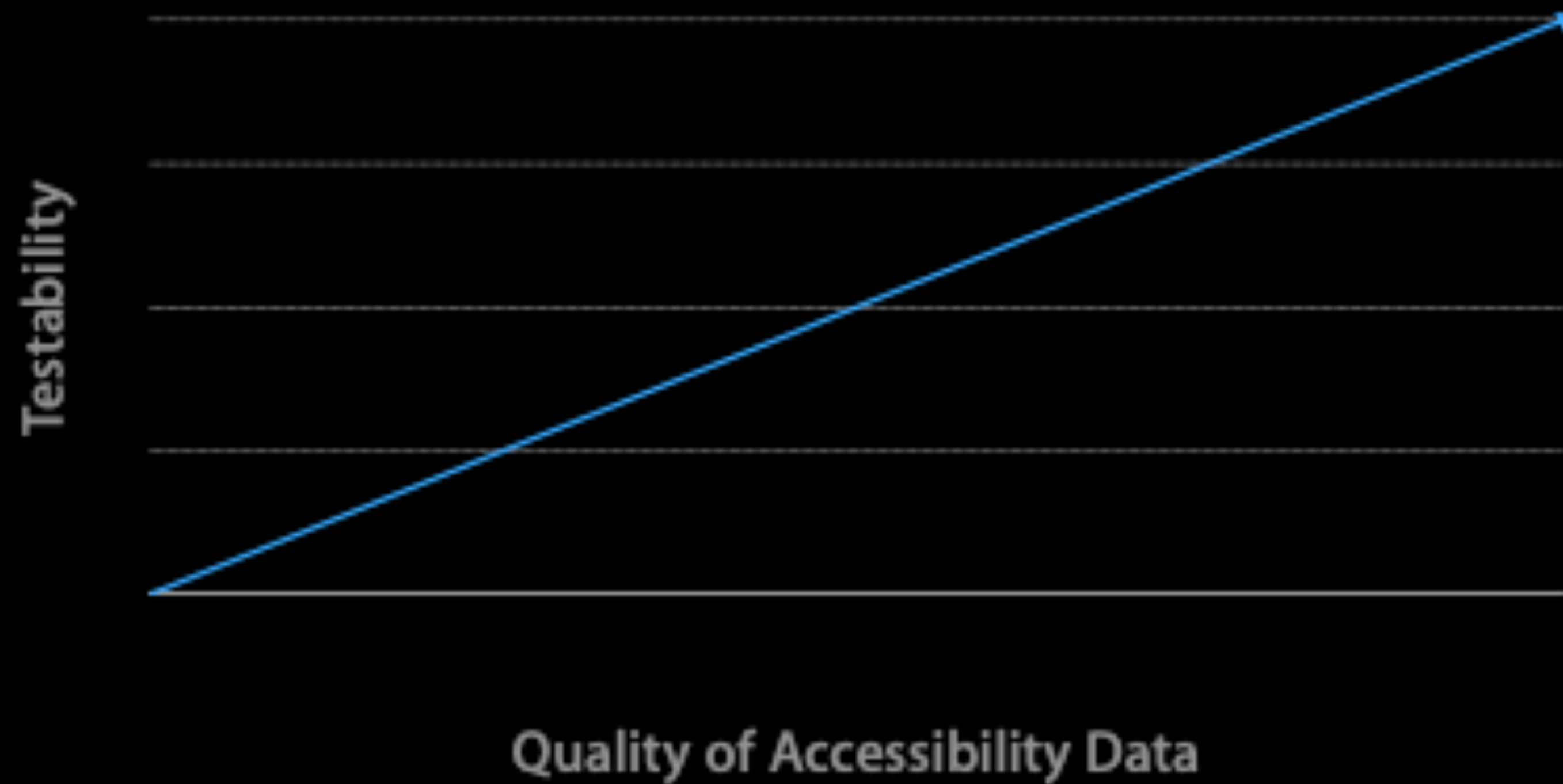
// 스크롤뷰의 모든 버튼을 반환(서브뷰의 서브뷰의 서브뷰에 포함되는)
app.scrollViews["Main"].descendants(matching: .button)

// 테이블의 0번째 셀
app.tables.cells.element(boundBy: 0)
```

```
let table = app.tables
let cell = table.cells.element(boundBy: 1)
cell.buttons.matching(NSPredicate(format: "label BEGINSWITH 'Delete'")).element.tap()
cell.children(matching: .button).matching(identifier: "Delete").element(boundBy: 0)
```


Demo

Accessibility and UI Test



Q & A

References

- <https://developer.apple.com/videos/play/wwdc2019/254/>
- <https://developer.apple.com/videos/play/wwdc2019/257/>
- <https://developer.apple.com/videos/play/wwdc2020/10020/>
- <https://developer.apple.com/videos/play/wwdc2015/406/>
- <https://developer.apple.com/documentation/accessibility>
- <https://sungdoo.dev/programming/accessibility-is-not-about-supporting-blind-people/>
- <https://sungdoo.dev/retrospective-or-psa/how-accessibility-nudges-you-to-be-better-developer/>
- <https://www.hackingwithswift.com/read/39/8/user-interface-testing-with-xctest>
- <https://www.hackingwithswift.com/articles/148/xcode-ui-testing-cheat-sheet>