SHIFT LEFT ON ACCESSIBILITY

Building Inclusive iOS Apps from Day 1

WHAT’S AHEAD

ABOUT ME

- iOS Engineer at Deque Systems
- From: CA, Currently in: DC
- Hobbies include: Sewing, knitting, pottery, cocktail-making
- Favorite show to rewatch: Brooklyn 99

ABOUT TODAY’S TALK

Topics to be covered:
- Intro to mobile accessibility
- Does the shift-left mentality actually work?

Bonus ✨:
- Bad jokes
- Pottery lore
- AI or not to AI

INTRO TO MOBILE ACCESSIBILITY

1
90% of Americans own a smartphone. Source: Pew Research Center

2
Across the world an estimated **16%** of people experience disability. Source: WHO

Across the U.S., **28%**. Source: CDC

3
5,000 new lawsuits in 2025 in the U.S. Source: UsableNet

MOBILE ACCESSIBILITY STANDARDS

Web Content Accessibility Guidelines (WCAG)

EN 301 549

ACCESSIBLE TECHNOLOGIES IN iOS

Voice Over
An on-device screen reader

Switch Control
Gestures activated using one or more switches

Dynamic Type
Larger text sizes across the device

And more…

EXAMPLE TIME…

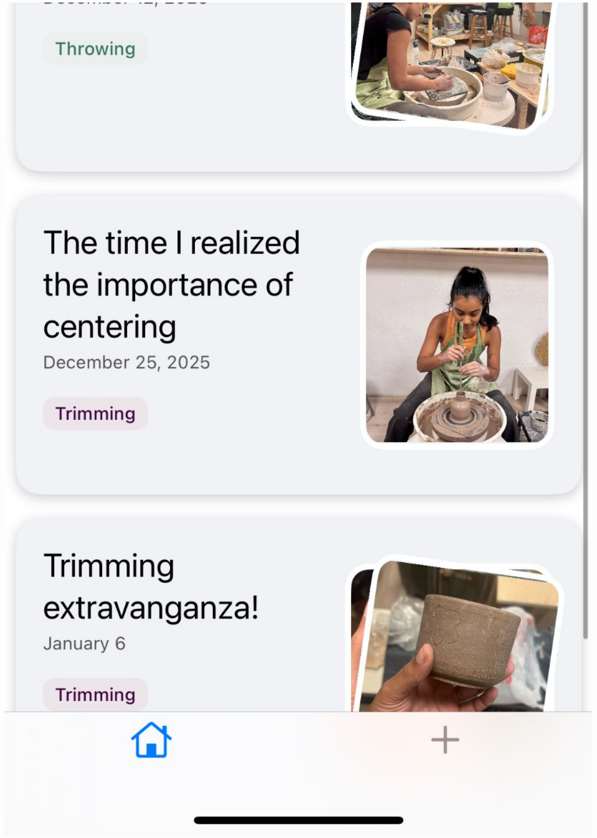
TAB BAR

Improvement:

- Provide a clear name with accessibility labels
- Success Criterion 4.1.2 - Level A - Name, Role, Value

Time to implement:

- 2 line change, ~30 seconds
- Deciding on a label, ~5 minutes


Throwing

The time I realized the importance of centering
December 25, 2025
Trimming

Trimming extravanganza!
January 6
Trimming

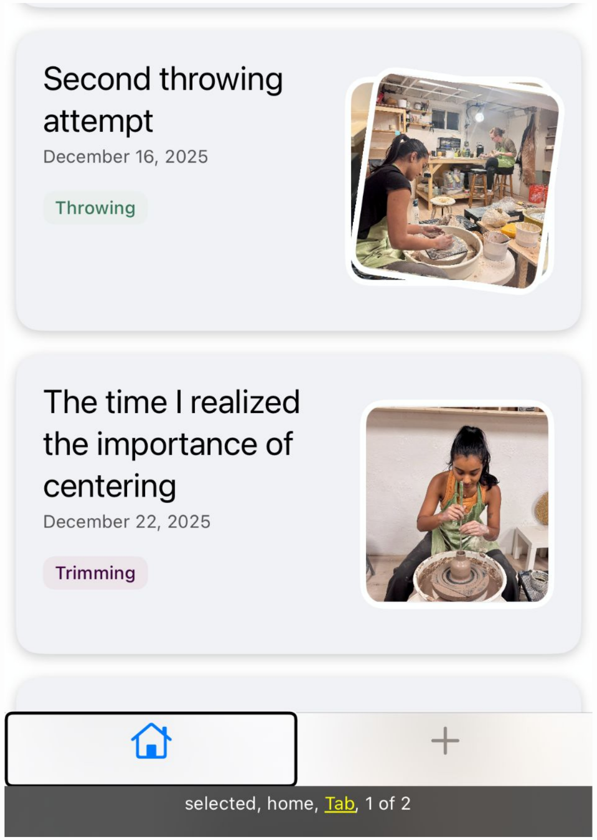
TAB BAR

Improvement:

- Provide a clear name with accessibility labels
- Success Criterion 4.1.2 - Level A - Name, Role, Value

Time to implement:

- 2 line change, ~30 seconds
- Deciding on a label, ~5 minutes


Second throwing attempt
December 16, 2025
Throwing

The time I realized the importance of centering
December 22, 2025
Trimming

selected, home, Tab, 1 of 2

TAB BAR

```
vc1.tabBarItem.image = UIImage(systemName: "house")
vc2.tabBarItem.image = UIImage(systemName: "plus")

// add accessibility label
vc1.tabBarItem.accessibilityLabel = "Feed"
vc2.tabBarItem.accessibilityLabel = "Create"
```

TAB BAR

```
vc1.tabBarItem.image = UIImage(systemName: "house")
vc2.tabBarItem.image = UIImage(systemName: "plus")

// add visible label below tab bar item
// also adds accessibility label
vc1.tabBarItem.title = "Feed"
vc2.tabBarItem.title = "Create"
```

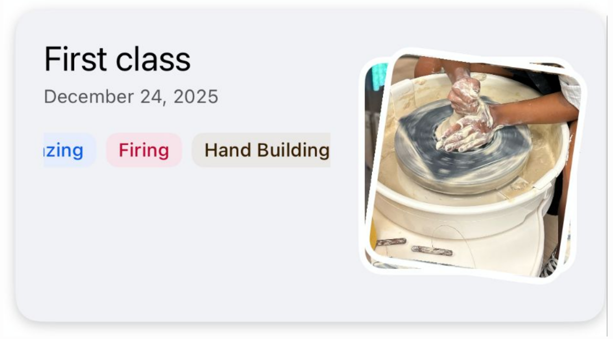
COLLECTION VIEW CELL

Improvement:

- Group elements in the cell together
- SC 1.3.1 - Level A - Info and Relationships

Time to implement:

- 20 line change to group, ~15 minutes
- 20 line change for actions, ~30 minutes


First class
December 24, 2025

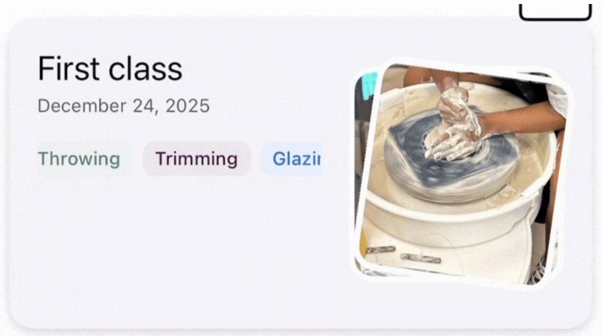
COLLECTION VIEW CELL

Improvement:

- Group elements in the cell together
- SC 1.3.1 - Level A - Info and Relationships

Time to implement:

- 20 line change to group, ~15 minutes
- 20 line change for actions, ~30 minutes



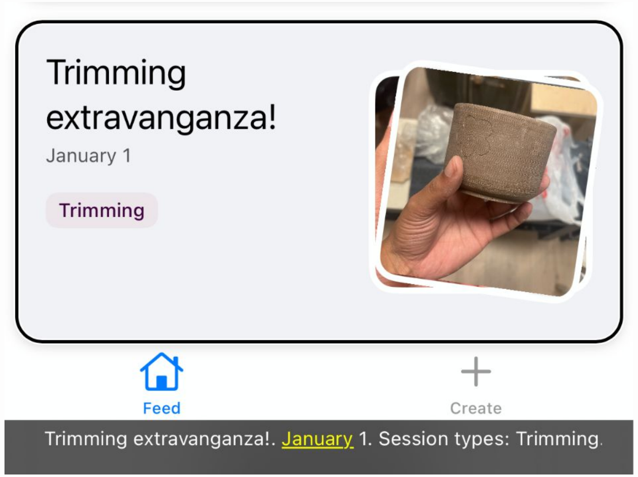
COLLECTION VIEW CELL

Improvement:

- Group elements in the cell together
- SC 1.3.1 - Level A - Info and Relationships

Time to implement:

- 20 line change to group, ~15 minutes
- 20 line change for actions, ~30 minutes


Trimming extravanganza!

January 1

Trimming

Trimming extravanganza!. January 1. Session types: Trimming.

COLLECTION VIEW CELL

```swift
func configureForAccessibility() {
   self.isAccessibilityElement = true
}
```

03

COLLECTION VIEW CELL

```swift
func buildAccessibleLabel() -> String {
  var labelComponents: [String] = []
  labelComponents.append(session.title)
  labelComponents.append(session.date.formattedForDisplay
())
  /* ... */
  return labelComponents.joined(separator: ". ")
}
```

03

COLLECTION VIEW CELL

```swift
func buildCustomActions() -> [UIAccessibilityCustomAction]
{
  let shareAction = UIAccessibilityCustomAction(
      name: "Share session",
      target: self,
      selector: #selector(shareAction)
  )
  return [shareAction]
}
```

COLLECTION VIEW CELL

```
@objc private func shareAction() -> Bool {
   // announce feedback to user
   UIAccessibility.post(notification: .announcement,
argument: "Sharing \(currentSession?.title ?? "session")")
   return true
}
```

03

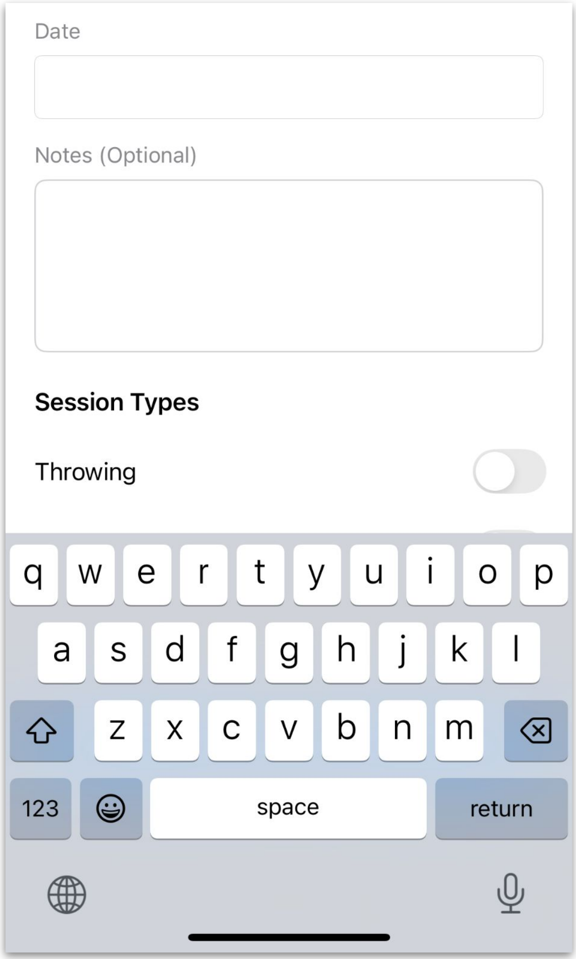
COLLECTION VIEW CELL

```
func configureForAccessibility() {
  self.isAccessibilityElement = true
  self.accessibilityLabel = buildAccessibleLabel()
  self.customActions = buildCustomActions()
}
```

KEYBOARD

Improvement:
- Add a way to dismiss keyboard
- SC 2.1.2 - Level A - No Keyboard Trap

Time to implement:
- 5-8 line change, ~30 minutes



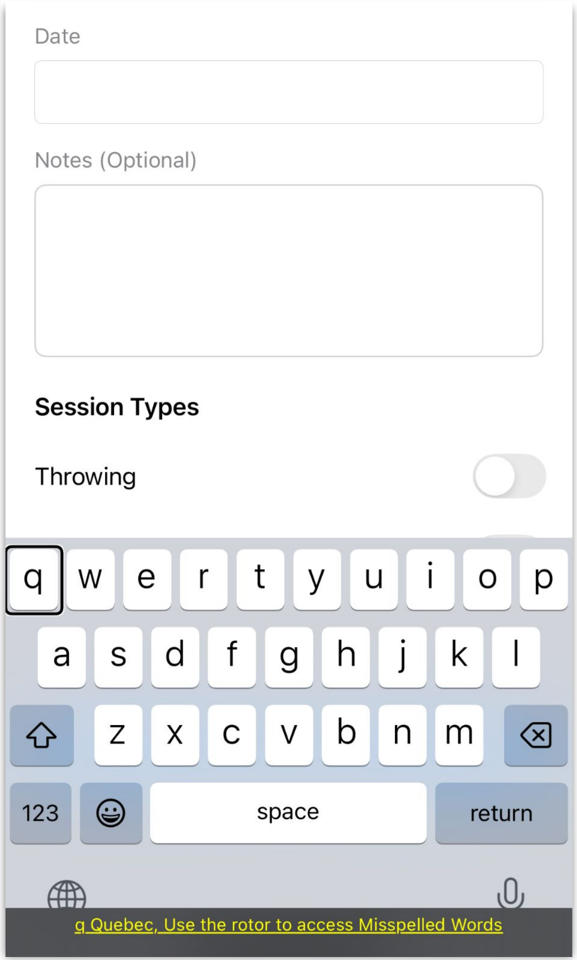
KEYBOARD

Improvement:

- Add a way to dismiss keyboard
- SC 2.1.2 - Level A - No Keyboard Trap

Time to implement:

- 5-8 line change, ~30 minutes


Date

Notes (Optional)

Session Types

Throwing

q Quebec, Use the rotor to access Misspelled Words

KEYBOARD

```swift
extension AccessibleSessionForm: UITextFieldDelegate {
    func textFieldShouldReturn(_ textField: UITextField) -> Bool {
        // Return key dismisses keyboard
        textField.resignFirstResponder()
        return true
    }
}
```

03

KEYBOARD

```swift
 // doesn’t work for multi-line textFields
func setupKeyboardDismissal_ReturnKey() {
    titleTextField.delegate = self
    dateTextField.delegate = self
}
```

03

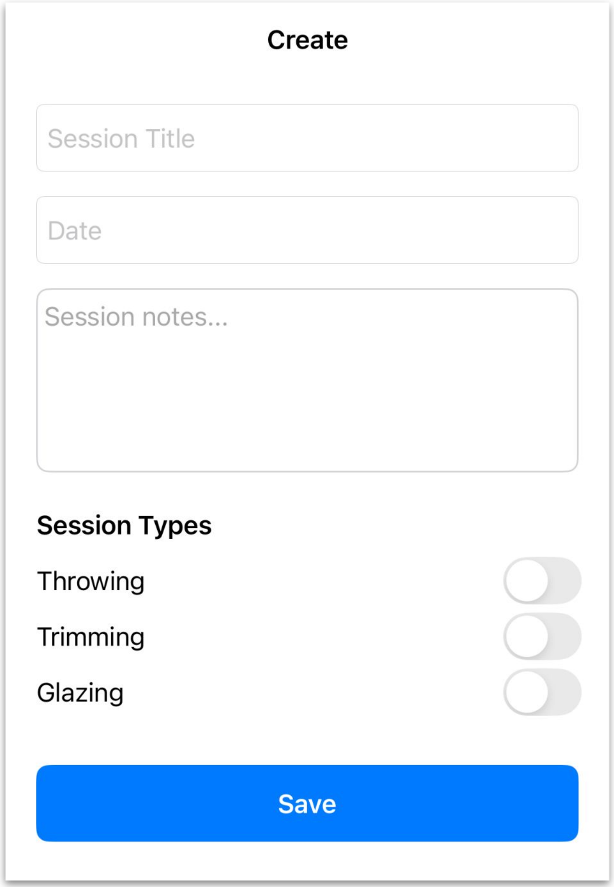
KEYBOARD

```swift
func setupKeyboardDismissal_TapGesture() {
    let tapGesture = UITapGestureRecognizer(target: self, action:
#selector(dismissKeyboard))
    tapGesture.cancelsTouchesInView = false
    addGestureRecognizer(tapGesture) // add a hint to tell users how
they can dismiss the keyboard!

}

@objc private func dismissKeyboard() {
    endEditing(true)
}
```

FORM VIEW

Improvement:
- Don’t rely on placeholder text alone
- SC 1.3.1 - Level A - Info and Relationships
- SC 4.1.2 - Level A - Name, Role, Value
- SC 3.3.2 - Level A - Labels or Instructions

Time to implement:
- 20-40 line change, ~1 hr

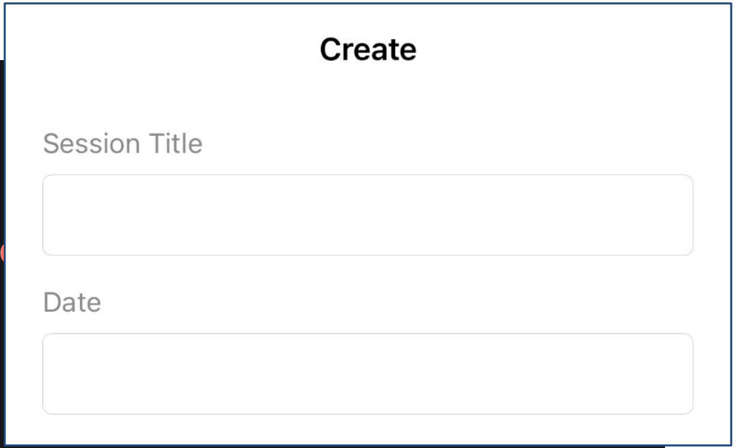


FORM VIEW

```swift
lazy var titleTextField: UITextField = {
    let textField = UITextField()
    textField.accessibilityLabel = "Session Title"
    return textField
}()

lazy var titleLabel: UILabel = {
    let label = UILabel()
    label.text = "Session Title"
    return label
}()
```

FORM VIEW

```
lazy var titleTextField: UITextField = {
    let textField = UITextField()
    textField.accessibilityLabel = "Sessi
    return textField
}()

lazy var titleLabel: UILabel = {
    let label = UILabel()
    label.text = "Session Title"
    return label
}()
```



FORM VIEW

```
class AccessibleTextField: UITextField {
    override var accessibilityPath: UIBezierPath? {
        get {
            guard let parentView = self.superview else { return nil }
            return UIBezierPath(rect: parentView.accessibilityFrame)
        }
        set {}
    }

    override var accessibilityLabel: String? {}
    ...
}
```

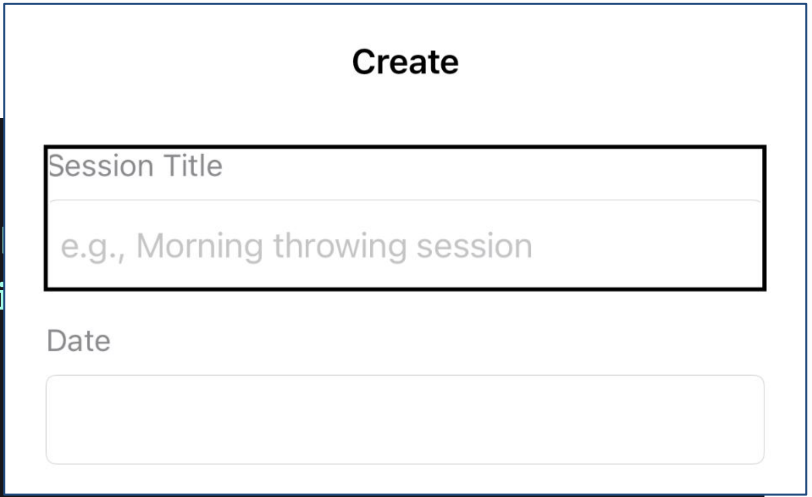
FORM VIEW

```swift
class LabeledTextField: UIView {
    private lazy var textField: AccessibleTextField = {
        let textField = AccessibleTextField()
        return textField
    }()

    private lazy var label: UILabel = {
        let label = UILabel()
        label.isAccessibilityElement = false
        return label
    }()
}
```

03

FORM VIEW

```
class LabeledTextField: UIView {
    private lazy var textField: Accessi
        let textField = AccessibleTextFi
        return textField
    }()

    private lazy var label: UILabel = {
        let label = UILabel()
        label.isAccessibilityElement = false
        return label
    }()
}
```



Create

Session Title

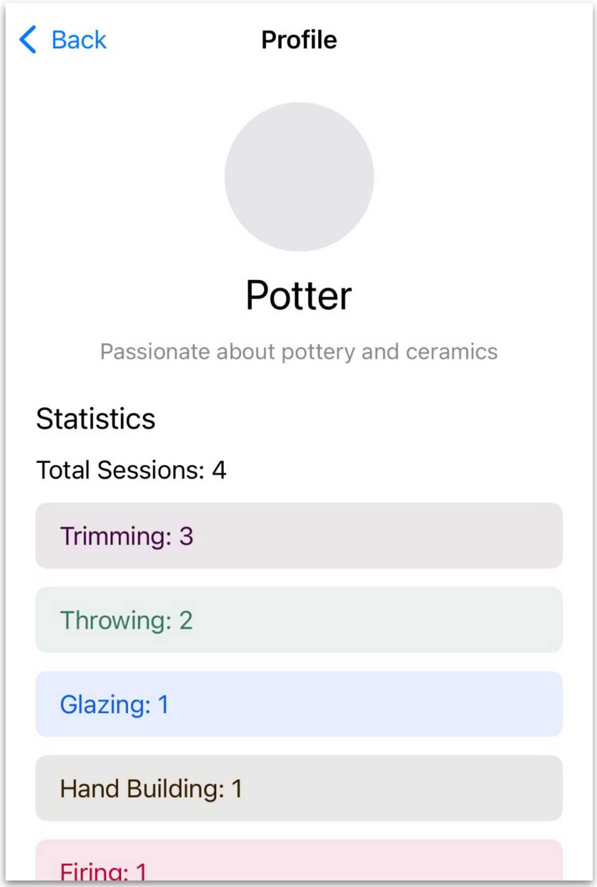
e.g., Morning throwing session

Date

DYNAMIC TYPE

Improvement:
- Allow text to resize itself
- SC 1.4.4 - Level AA - Resize Text

Time to implement:
- 0 line change if you do this from the start*
- More involved afterwards or with custom fonts


Back
Profile

Potter

Passionate about pottery and ceramics

Statistics

Total Sessions: 4

Trimming: 3

Throwing: 2

Glazing: 1

Hand Building: 1

Firing: 1

DYNAMIC TYPE

```
label.font = UIFont.preferredFont(forTextStyle:
.subheadline)
label.adjustsFontForContentSizeCategory = true
```

DYNAMIC TYPE

```
let customFont = UIFont(name: name, size: size)
label.font = UIFontMetrics(forTextStyle: textStyle).scaledFont(for:
customFont)
label.adjustsFontForContentSizeCategory = true
```

DYNAMIC TYPE

```
extension UIFont {
    static func scaledCustomFont() -> UIFont {
        let customFont = UIFont(name: name, size: size)
        let fontMetrics = UIFontMetrics(forTextStyle:
    textStyle)
        return fontMetrics.scaledFont(for: customFont)
    }
}
```

AI OR NOT TO AI

Can AI write accessible code?

Short answer: Yes

Long answer: Not right out of the box, but with some prompting and testing in between, you can get on the right path.

AIMC: AI Model Accessibility Checker

- Put together by the GAAD Foundation and ServiceNow.
- Asked models to create different web pages with NO accessibility guidance.
- Ran axe-core on each web page to catch accessibility issues.
- Created a ranking based on those findings.
- The winner? **Open AI's GPT 5.2 Pro**

FINAL THOUGHTS

- Testing early and often helps catch issues when it’s the cheapest
- Reusable components are your friend!
- Prioritize accessibility in new features (retrofitting everything at once isn’t realistic)
- Two birds one stone: common accessibility pitfalls affect all users

RESOURCES

Code:

- github.com/nbaddam/throw
- Both accessible and inaccessible implementations
- Reusable accessible components

Free tools:

- Apple Accessibility Inspector + Accessibility Audit
- WWDC Accessibility Sessions
- Appt foundation: https://appt.org/en/
- Color Contrast Analyzer (CCA):
  https://github.com/ThePacielloGroup/CCAe

THANK YOU!