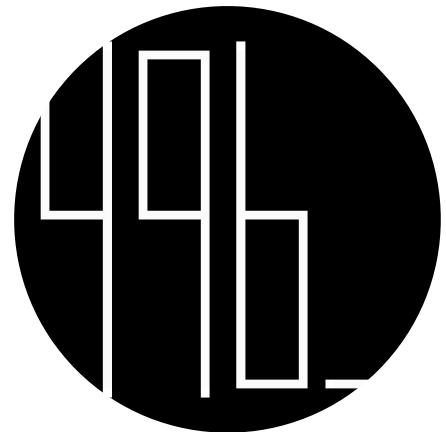


# iPadOSでマウス・キーボード対 応アプリを作る

iOSDC 2020

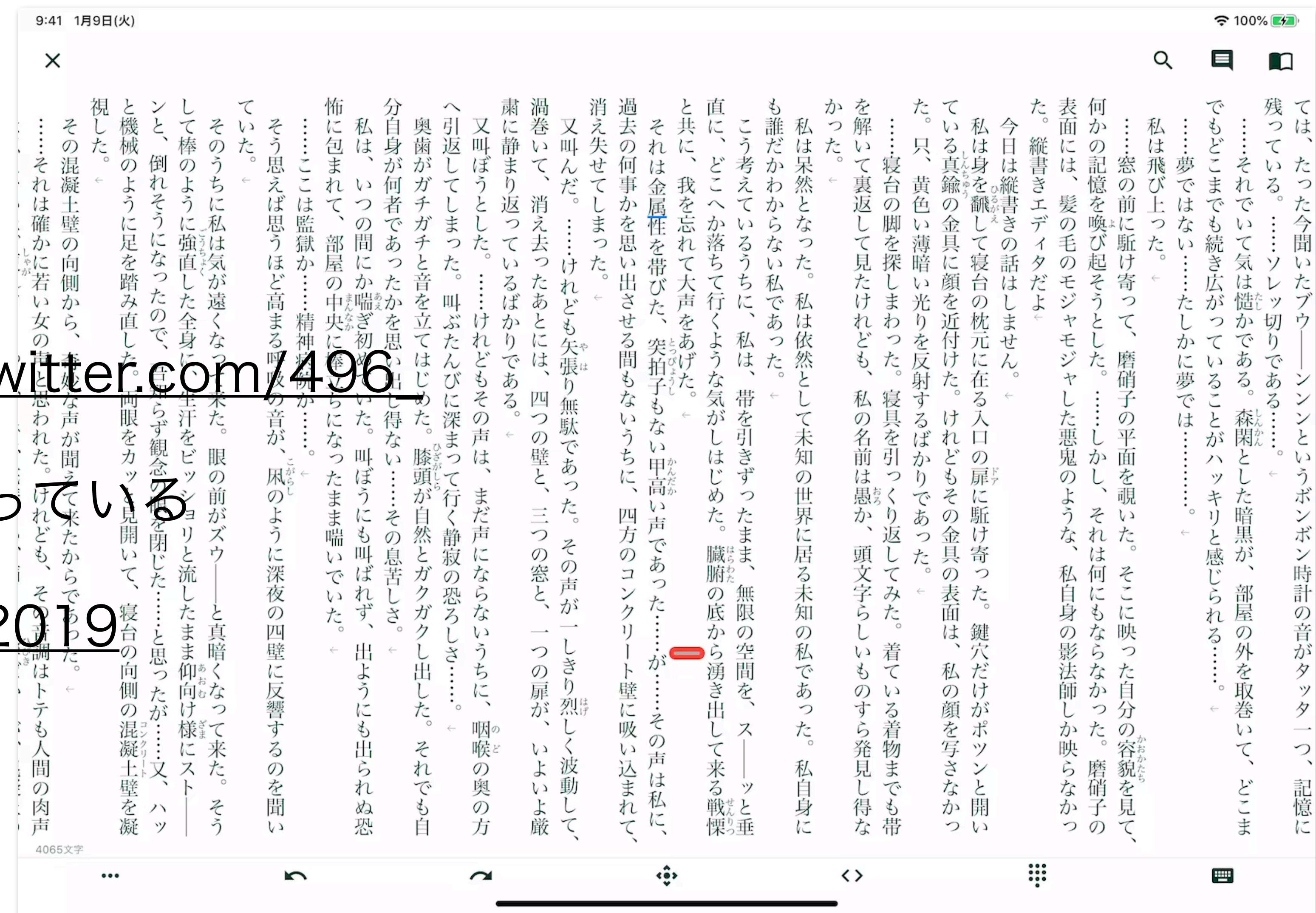


rokuroku

# 自己紹介

## rokuroku

- **rokuroku / 野中涼**
- 所属: ピクシブ株式会社
- Twitter: @496\_ <https://twitter.com/496>
- 趣味で縦書きエディタを作っている
- iOSDC 2018 / iOSDC 2019
- DroidKaigi 2019



# iOS 13.4で変わったこと

## 今日話すマウスの話、キーボードの話

- ・ ポインターデバイスが使えるようになったこと（マウスやトラックパッド）
- ・ キーイベントが拾えるようになったこと（キーボード）

# マウス・キーボード対応

## 何が嬉しいのか

- 新しい作業環境
  - iPad Pro用のMagic Keyboardはトラックパッド+キーボード
- 操作性の向上
  - 操作の精度 - マウスやトラックパッドでは指よりも細かい操作が可能
  - 操作の速度 - コンテキストメニュー キーボードショートカット
  - 操作の分離 - スクロールとドラッグ&ドロップなどの分離

# アウトライン

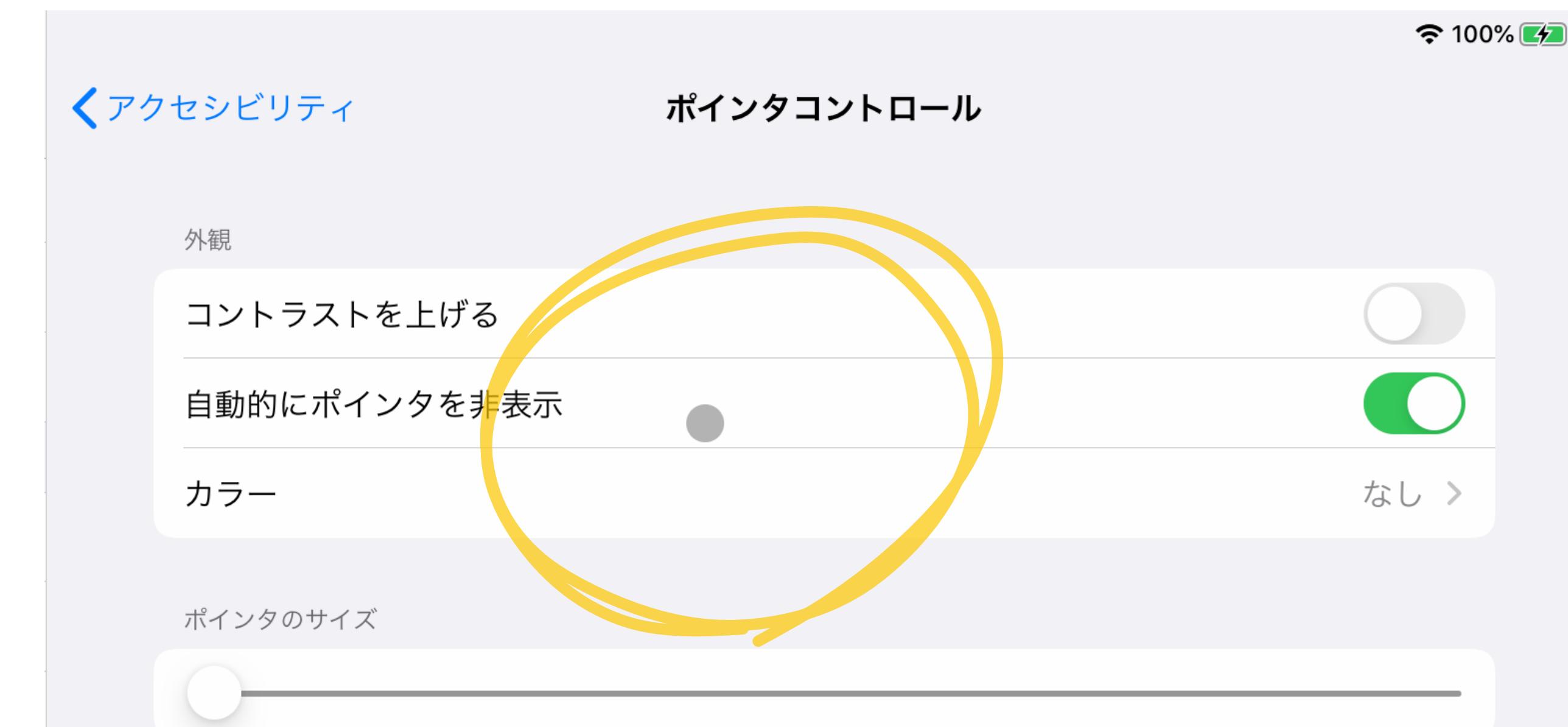
## マウスの話、キーボードの話

- ・ ポインターの話
- ・ コンテキストメニューの話
- ・ ジェスチャーの話
- ・ キーボードの話
- ・ まとめ

ポインター  
UIPointerInteraction

# ポインター 見た目

- iPadOS 13.4以降はマウスやトラックパッドを接続することができる
- デフォルト設定だと時間が経つと消えるグレーの丸



# ポインター 見た目

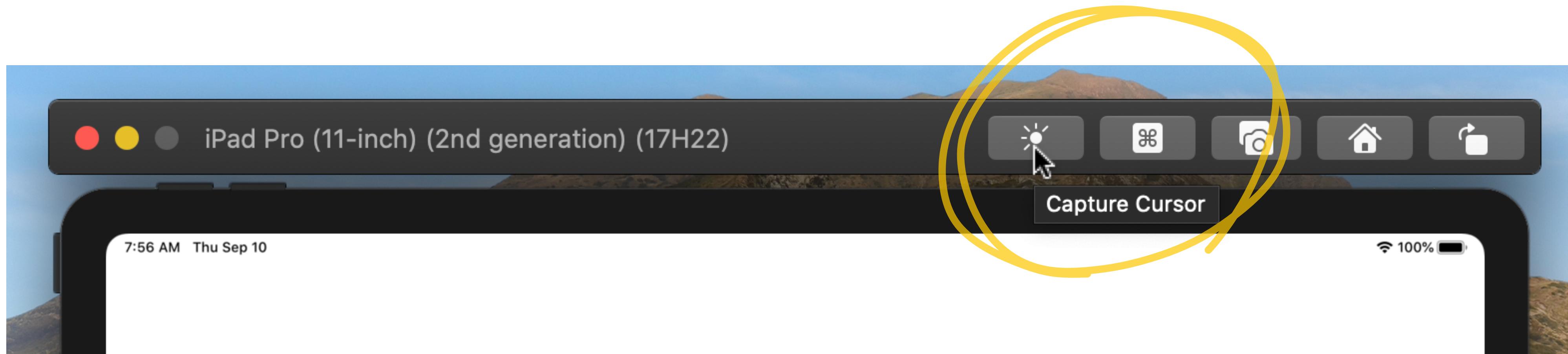
- ・デフォルト設定だと時間が経つと消えるグレーの丸
- ・発表では見やすさのために以下の設定を使っています
  - ・自動的にポインタを非表示: オン→オフ

- ・カラー: なし→レッド



# ポインター シミュレーター

- ・ ポインターデバイス（マウス・トラックパッド）が使えるようになった
- ・ iPadOS 13.4以降のシミュレーター
  - ・ Capture Cursorを押すとマウスとキーボードがキャプチャされる



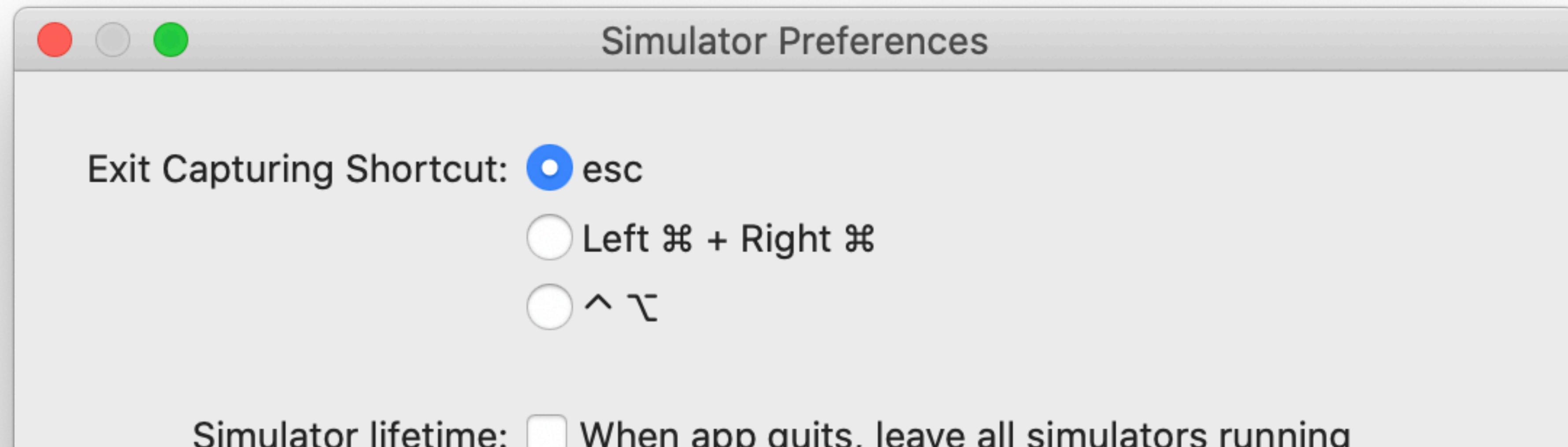
# ポインター シミュレーター

- ・ ポインターデバイス（マウス・トラックパッド）が使えるようになった
- ・ iPadOS 13.4以降のシミュレーター
  - ・ Capture Cursorを押すとマウスとキーボードがキャプチャされる
  - ・ Escを押すとキャプチャーモードから抜ける



# ポインター シミュレーター

- ・ ポインターデバイス（マウス・トラックパッド）が使えるようになった
- ・ iPadOS 13.4以降のシミュレーター
  - ・ Escキーの挙動の確認したいときはPreferencesから設定を変えると良い



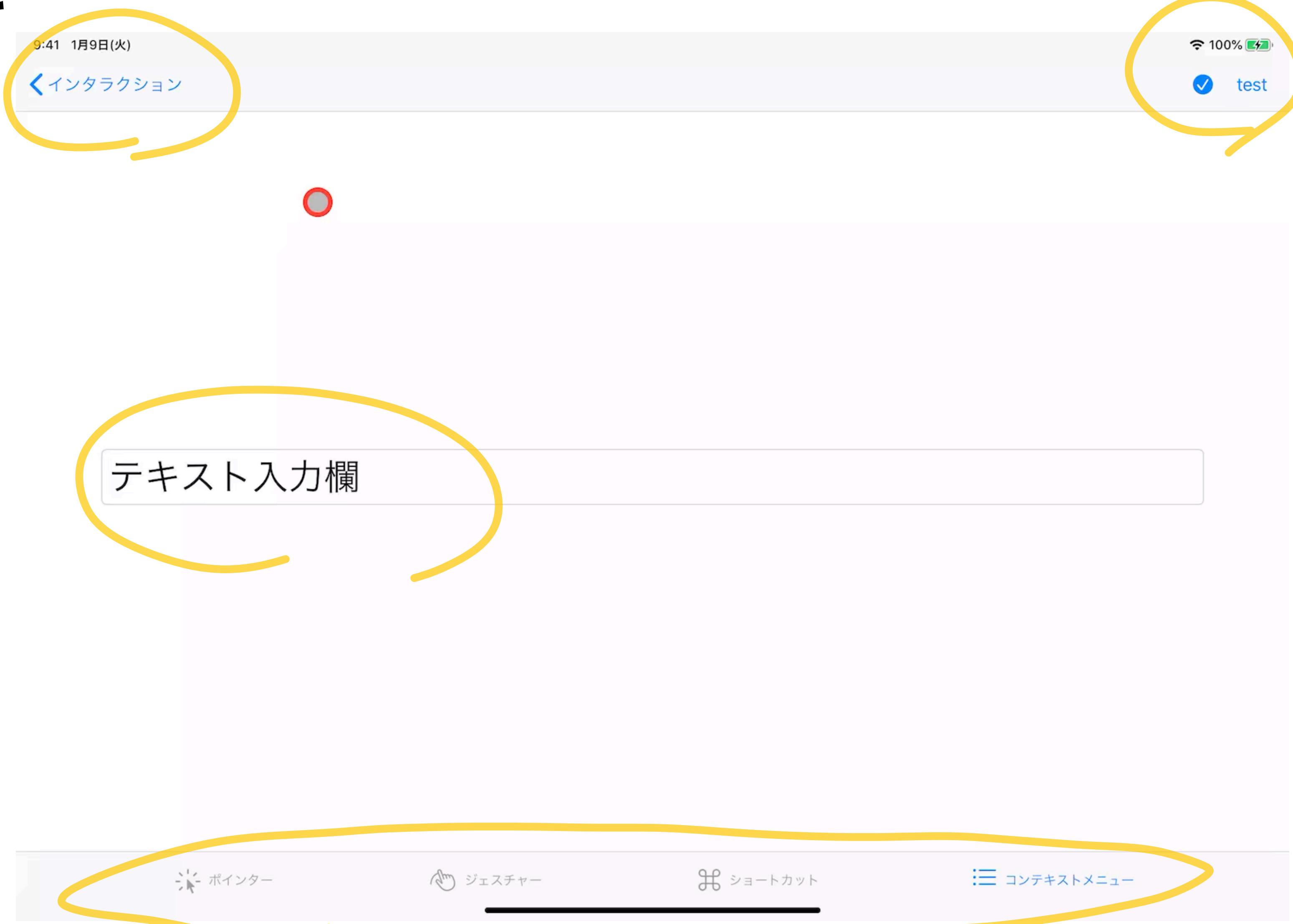
# ポインター

## 3つのこと

- ・自動で対応されていること
- ・簡単に対応できること
- ・よりカスタマイズしたい場合

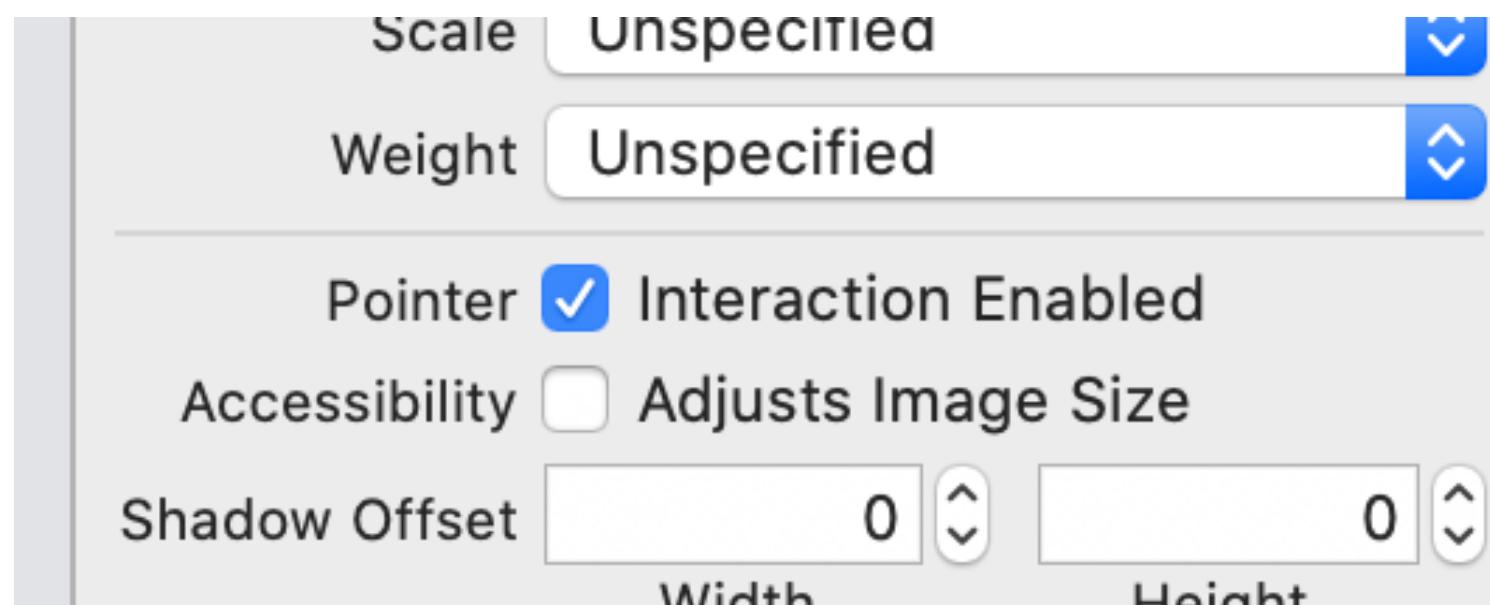
# ポインター 自動に対応されていること

- UIBarButtonItem
  - ナビゲーションバーなど
- UITabBarItem
  - タブバーのアイテム
- UITextField / UITextView
  - テキスト入力欄



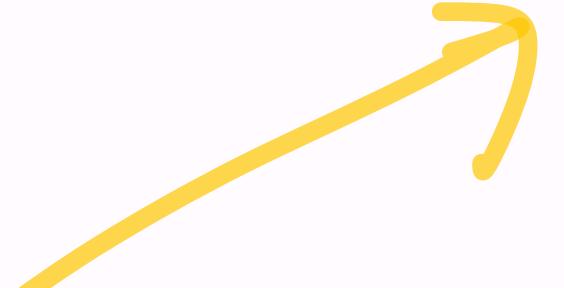
# ポインター 簡単に対応できること

- UIButton
  - isPointerInteractionEnabledを  
trueにするだけ



- ボタン全体ではなく文字のところだけ

	hover	highlight	automatic	lift	default
false	Button	Button	Button	Button	Button
true	Button	Button	Button	Button	Button
true	Button	Button	Button	Button	Button



iOSDC 2020

# ポインター エフェクトの種類

- pointerStyleProviderで返す効果
  - hover : 領域が薄く覆われる
  - highlight : 領域全体を持ち上がる
  - automatic : 自動割り当て
  - lift : ポインターが下に潜り込む

ボタンに背景色を設定

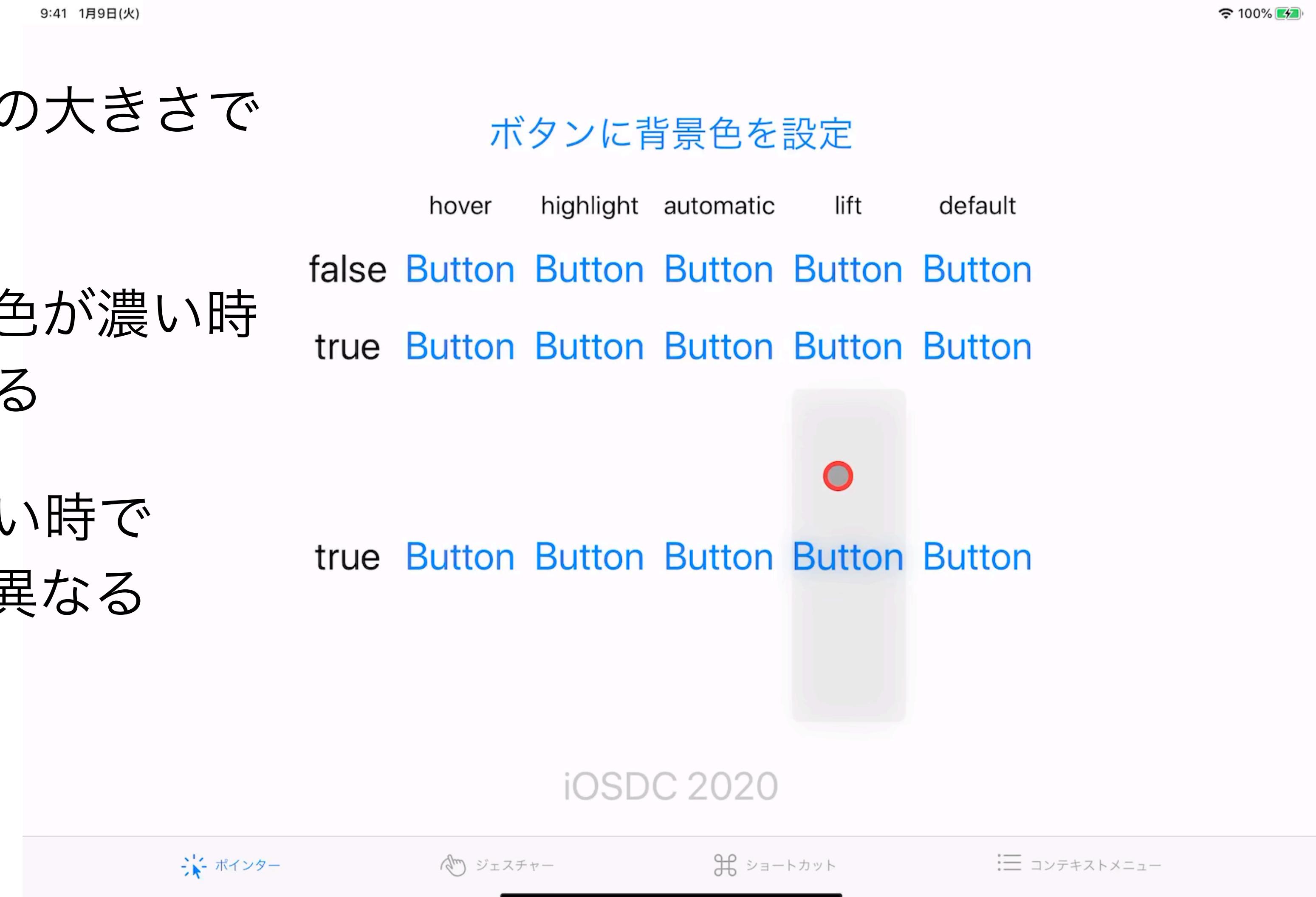
	hover	highlight	automatic	lift	default
false	Button	Button	Button	Button	Button
true	Button	Button	Button	Button	Button

true Button Button Button Button

iOSDC 2020

# ポインター エフェクトの種類

- ・同じエフェクトでもボタンの大きさで挙動が変わる
- ・ポインターのグレーも背景色が濃い時と薄い時で明るさが変化する
- ・ボタンに背景がある時とない時でautomaticのエフェクトが異なる



# ポインター 領域が大きい場合（背景に透明度がある）

背景に透明度がある	通常	領域が大きい場合
デフォルトのまま	ラベル周辺のみをhighlight	ラベル周辺のみをhighlight
hover	ポインターはそのまま 指定した領域を薄く色付け	ポインターはそのまま 指定した領域を薄く色付け
highlight	ポインターは指定した領域の形状 ポインターの領域を薄く色付け	ポインターはそのまま 指定した領域を薄く色付け（角丸）
lift	ポインターは見えなくなる 指定した領域にブラー	ポインターはそのまま 指定した領域にブラー
automatic	highlightと同じ	highlightと同じ

# ポインター

## 領域が大きい場合×背景が不透明の場合

背景が不透明の場合	通常	領域が大きい場合
デフォルトのまま	liftに同じ	liftに同じ
hover	ポインターはそのまま 指定した領域を薄く色付け	ポインターはそのまま 指定した領域を薄く色付け
highlight	ポインターは指定した領域の形状 ポインターの領域を薄く色付け	ポインターはそのまま 指定した領域を薄く色付け
lift	ポインターは見えなくなる 指定した領域が持ち上がる	ポインターはそのまま 指定した領域が持ち上がる
automatic	liftに同じ	liftに同じ

# ポインター ボタン・エフェクトの振る舞い

- automaticは適切なエフェクトになる
  - 背景色が透明度を持つときはhighlight
  - 背景色が不透明な場合はlift
- UIButtonのエフェクトは何もしなくても綺麗に動くようになっているので基本的に変更する必要はない

# ポインター UIButton以外へのポインター対応

- UIButton以外について
  - UIPointerInteractionを使う
  - 任意の形状を指定できる
    - アニメーションも可能
    - その他の効果も対応可能



# ポインター

## iPadOS以外について

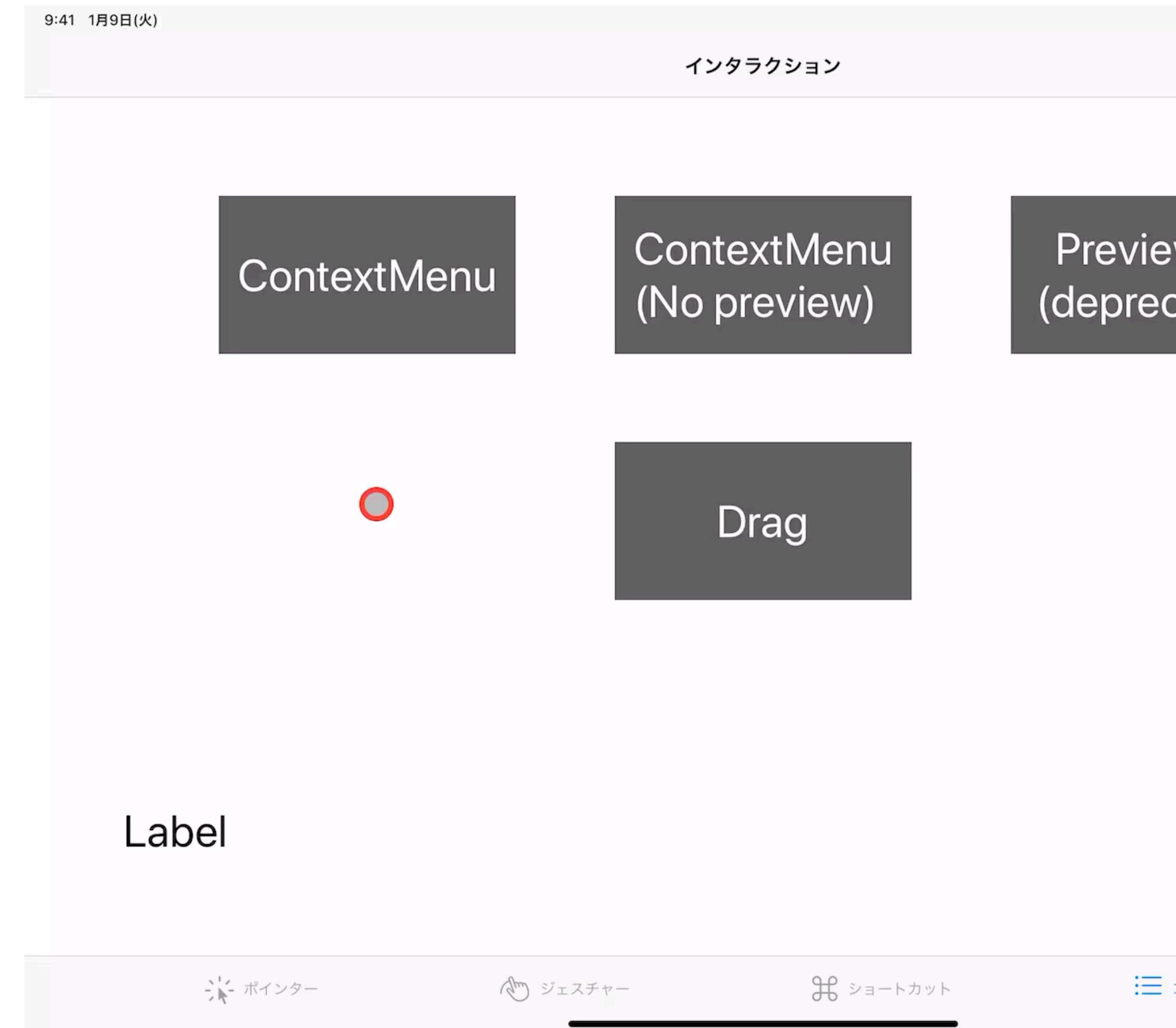
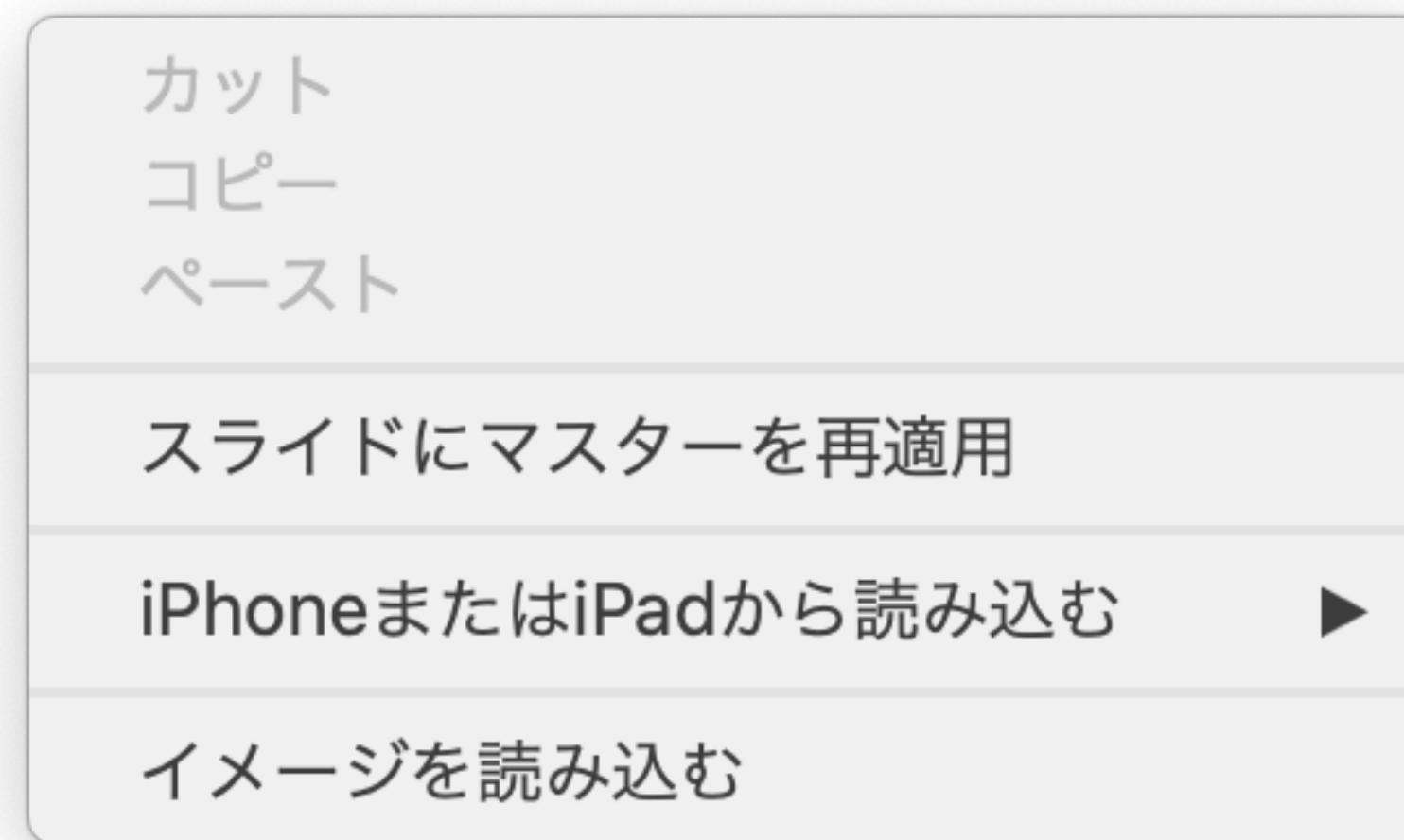
- ・ この発表でiPadOS限定の話はここまでUIPointerInteractionだけ
  - ・ iOSやCatalystではUIPointerInteractionは動作しない
- ・ おそらく……
  - ・ iOSもマウス対応されているがアクセシビリティとしての側面が強い
    - ・ iOSにはマウスは接続できるがトラックパッドは接続できない
  - ・ CatalystはmacOSのマウスカーソルの世界観を尊重した

**コンテキストメニュー**  
UIContextMenuInteraction

# コンテキストメニュー

## macOSの場合

- iOSのコンテキストメニュー→
- macOSのコンテキストメニュー↓

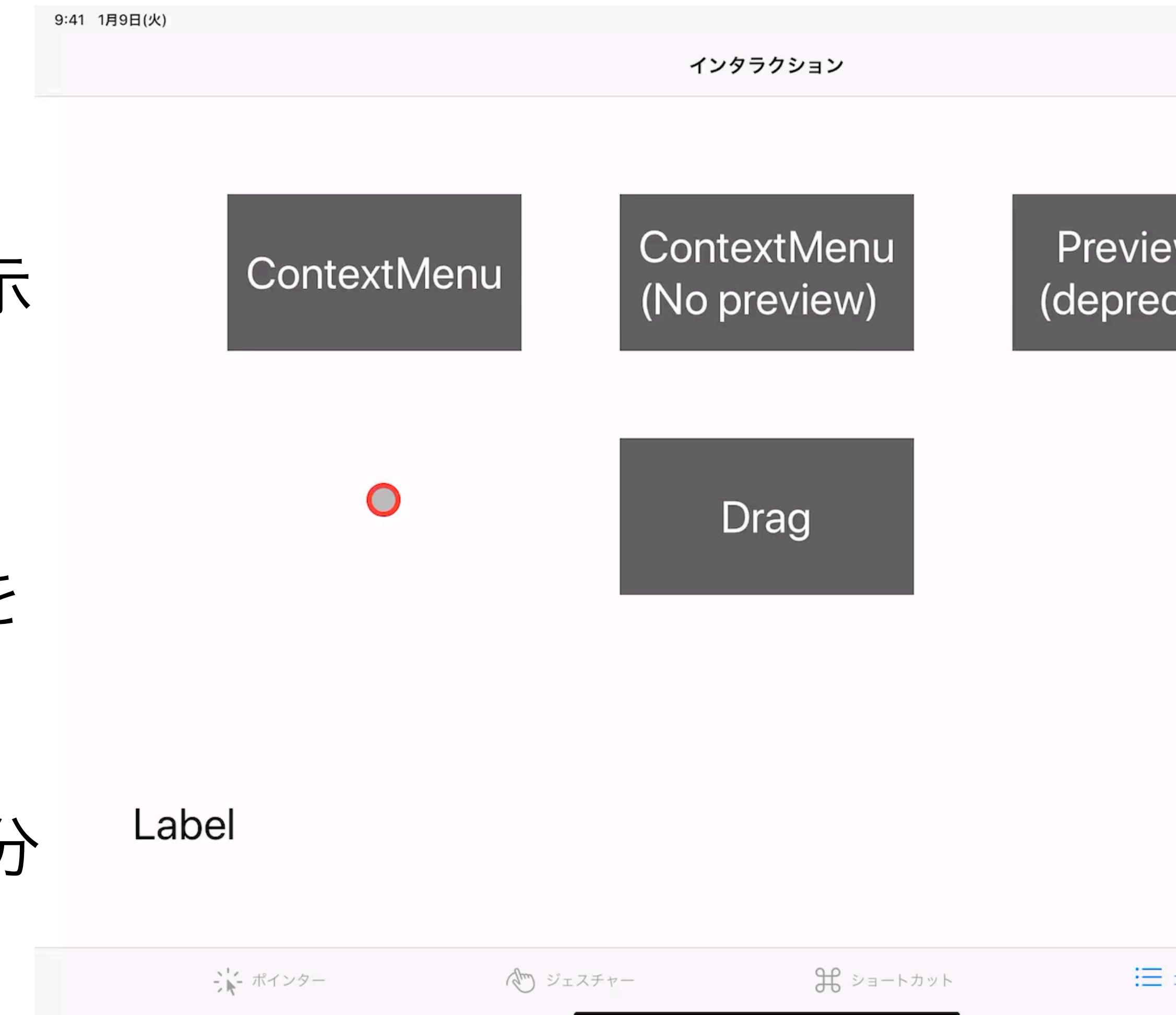


# コンテキストメニュー

iOS 13での導入 (Catalyst登場のタイミング)

- UIContextMenuInteraction

- 右クリックでコンテキストメニューを表示
  - Control + (タップ or クリック) も可
  - ロングタップでもコンテキストメニューを表示 (こっちはあればプレビューも表示)
  - プレビューをタップした際の挙動は自分で決めることができる



# コンテキストメニュー

## 似た機能 (3D Touch登場のタイミング)

- UIViewControllerPreviewingDelegate
  - iOS 10で3D Touchと一緒に追加された
  - 3D Touchするとプレビューが出る
  - 上にスワイプするとアクションメニューが出る
  - さらに強く押すかタップすると遷移する
  - コンテキストメニューの登場でdeprecatedになった

Previewing  
(deprecated)

Label



# おまけ ドラッグ&ドロップ

```
let dragInteraction = UIDragInteraction(delegate: self)
dragInteraction.isEnabled = true // for iOS
dragView.addInteraction(dragInteraction)
```

9:41 1月9日(火)

インタラクション

- iOS 11で導入された

- UIDragInteraction

ContextMenu

ContextMenu  
(No preview)

Preview  
(deprecated)

- タップもペンシルもロングタップする必要があった

- マウスではノータイムでドラッグを開始できる

Drag

- UIDropInteraction

- マウスだとドロップ先を指で隠さずに済む

Label

ジェスチャー  
UIGestureRecognizer

# ジェスチャー (Hover)

iOS 13で新規で追加されたもの

9:41 1月9日(火)

- UIHoverGestureRecognizerの追加

Tap LongPress Pan Rotation Pinch Hover

- Catalyst対応で入ったもの



- ドキュメントには

UIHoverGestureRecognizer has none

no effect when your app runs

in iOS. と書かれているが実際には

機能する

動くこともあるが UIPanGestureRecognizer

にコンフリクトしてスクロールできなくなるたりする

AlphaShift Alternate Command Control NumericPad Shift

# ジェスチャー (Hover)

## 余談：SwiftUI

- SwiftUIにもonHoverやhoverEffectがある
  - onHover(perform:)
  - hoverEffect(\_:)
- これらはiOSやCatalystでは動作しない
  - UIPointerInteractionと同じ実装になっているのか

# 既存のジェスチャー もっとマウス対応

- iOS 13.3まで: スクリーンタッチとPencilによるイベントの2種
- iOS 13.4から: 上記に加えてマウス・トラックパッドによるイベントの3種に
  - direct, pencil, indirectPointer
- [UIApplicationSupportsIndirectInputEvents](#)をInfo.plistでYESにするとマウス・トラックパッドのイベントを区別可能になる

Key	Type	Value
Application supports indirect input events	Boolean	YES
Bundle identifier	String	\$(PRODUCT)

# 既存のジェスチャー もっとマウス対応

<code>UIApplicationSupportsIndirectInputEvents</code>	<code>YES</code>	<code>NO</code>
<code>UITouch</code> の <code>type</code>	<code>indirectPointer</code> が入ってくる	ポインターは <code>direct</code> に集約される
<code>UIRotationGestureRecognizer</code>	<code>numberOfTouches = 0</code>	<code>numberOfTouches = 2</code>
<code>UIPinchGestureRecognizer</code>	<code>numberOfTouches = 0</code>	<code>numberOfTouches = 2</code>
<code>UIGestureRecognizer</code> の <code>allowedTouchTypes</code>	<code>indirectPointer</code> が使える	ポインターは <code>direct</code> に集約される

# 既存のジェスチャー

## スクリーンタッチ、Pencil、ポインター

9:41 1月9日(火)

- マウス・トラックパッドはlocationの精度が非常に高い
- 今まで座標は画面のピクセル解像度だけの離散値だったが連続値になる

Tap LongPress Pan Rotation Pinch Hover

none



AlphaShift Alternate Command Control NumericPad Shift

↑ ポインター

↗ ジェスチャー

☰ ショートカット

≡ コンテキストメニュー

# ジェスチャー

## UITapGestureRecognizer

- buttonMaskRequired
  - クリック種別を指定できる
    - primary: クリック
    - secondary: 右クリック
  - スクリーンタッチからのイベントは拒否しない限り送られてくる
  - allowedTouchTypes をindirectPointerのみにすると拒否できる

# ジェスチャー

## Rotation / Pinch

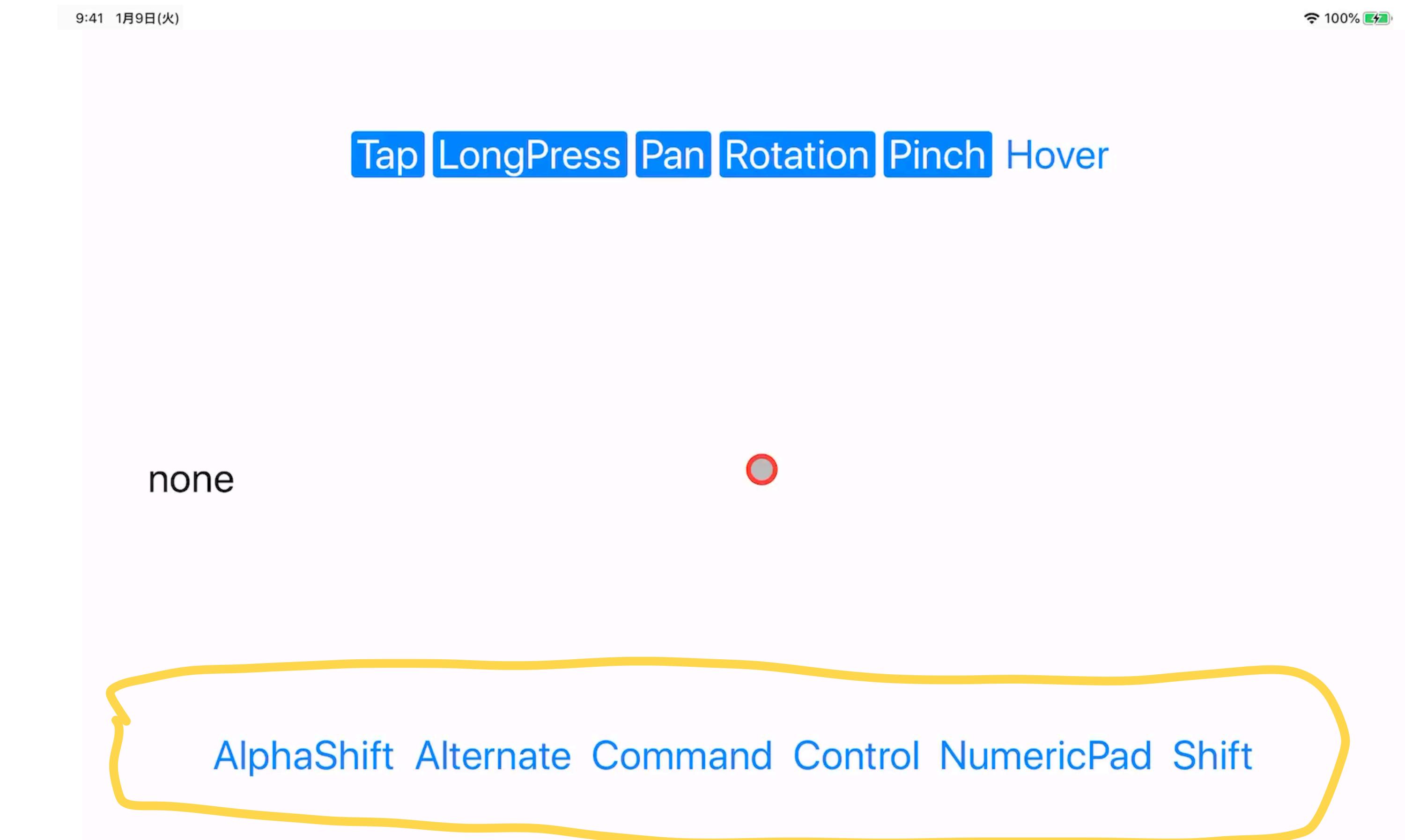
- allowedTouchTypes を空配列にするとトラックパッド由来に限定できる
  - 内部的にUIEvent.typeがtransformなので
  - つまりtouchesイベントではないのでここが空配列になる

# ジェスチャー UIPanGestureRecognizer

- allowedTouchTypes を空配列にするとマウス・トラックパッドに限定できる
  - 内部的にUIEvent.typeがscrollなので
- allowedScrollTypesMask
  - スクロール種別を指定できる (UIEvent.typeがscrollの際のフィルタ)
    - discrete: マウスのホイール
    - continuous: トラックパッド

# ジェスチャー 修飾キー

- UIGestureRecognizerに修飾キーの状態が追加された



# ジェスチャー+キーボード

## 範囲選択

- これまでShiftやCommandでの範囲選択は存在したか
- 物理キーボードでの文字入力はOSのIMEが解釈したものをUITextInput経由での受け取る
- このとき文字列の範囲選択はキーボードで行えるが、あくまで処理しているのはアプリの外だった

キー ボード  
UIKey

# キーボード

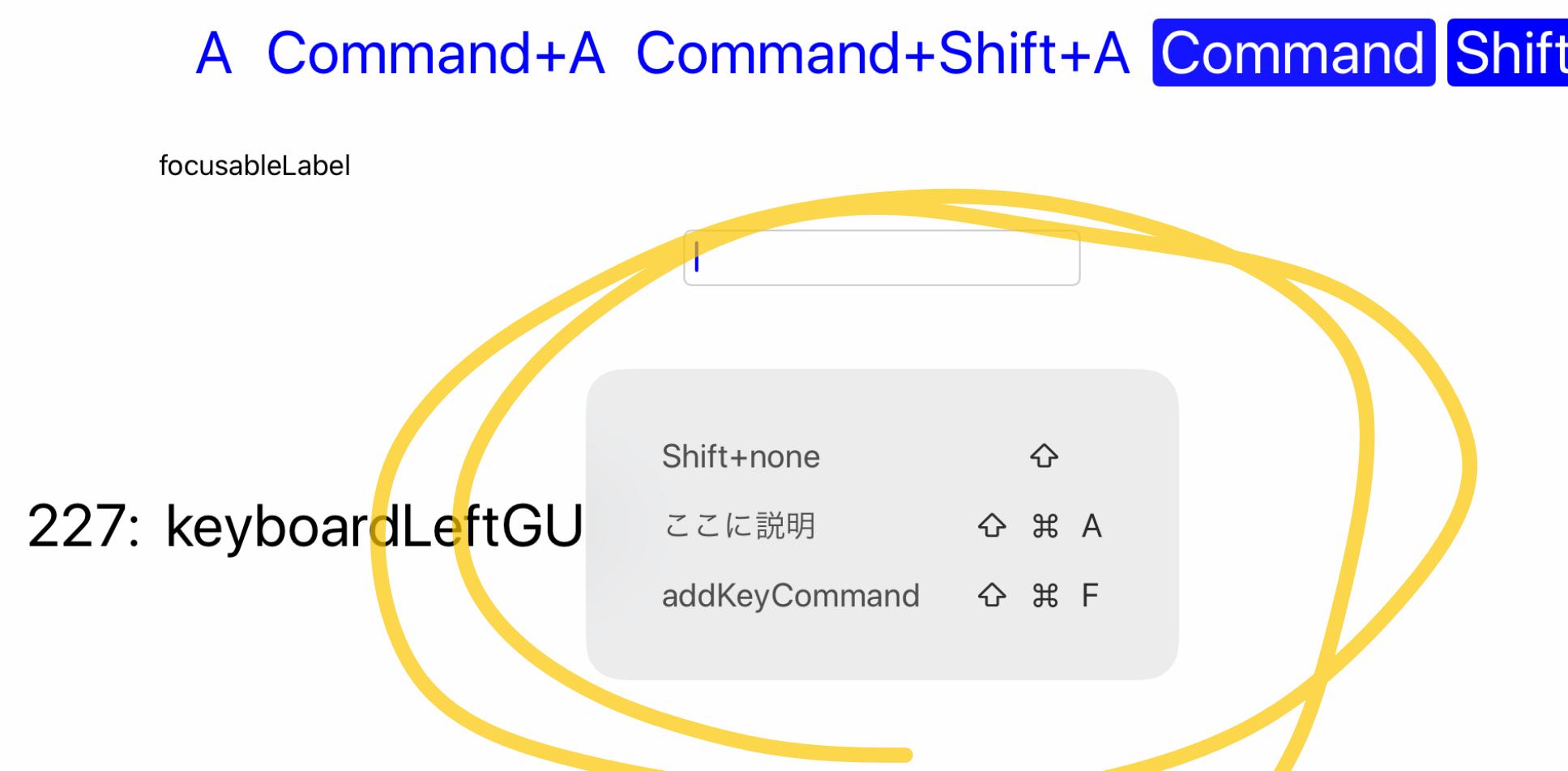
## キーボードについてのトピック

- ・ キーコマンド（キーボードショートカット）
- ・ フルキーボードアクセス（iOS 13.4から）
- ・ キーイベント（iOS 13.4から）

# キーボード

## キーボードショートカット

- UIResponderのkeyCommandsにUIKeyCommandを登録しておください
- 自明ではないショートカットにはUIKeyCommandにdiscoverabilityTitleをセットすると良い
  - Commandキーを長押ししたときにdiscoverabilityTitleのあるショートカット一覧が表示される



# キーボード

## フルキーボードアクセス

- ・ 設定>アクセシビリティ>キーボード>フルキーボードアクセス
  - ・ iOSの操作をすべてキーボードだけができるようになります
- ・ Tab+Gでタッチジェスチャーの再現

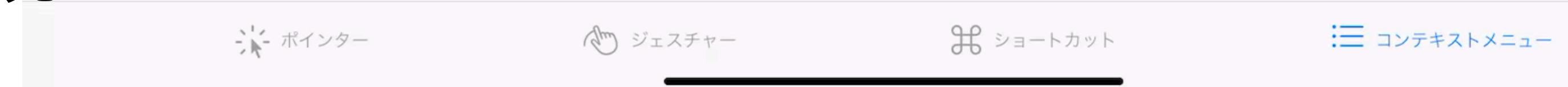
ContextMenu

ContextMenu  
(No preview)

Previewing  
(deprecated)

Drag

Label



# キーボード

## フルキーボードアクセス

ContextMenu

ContextMenu  
(No preview)Previewing  
(deprecated)

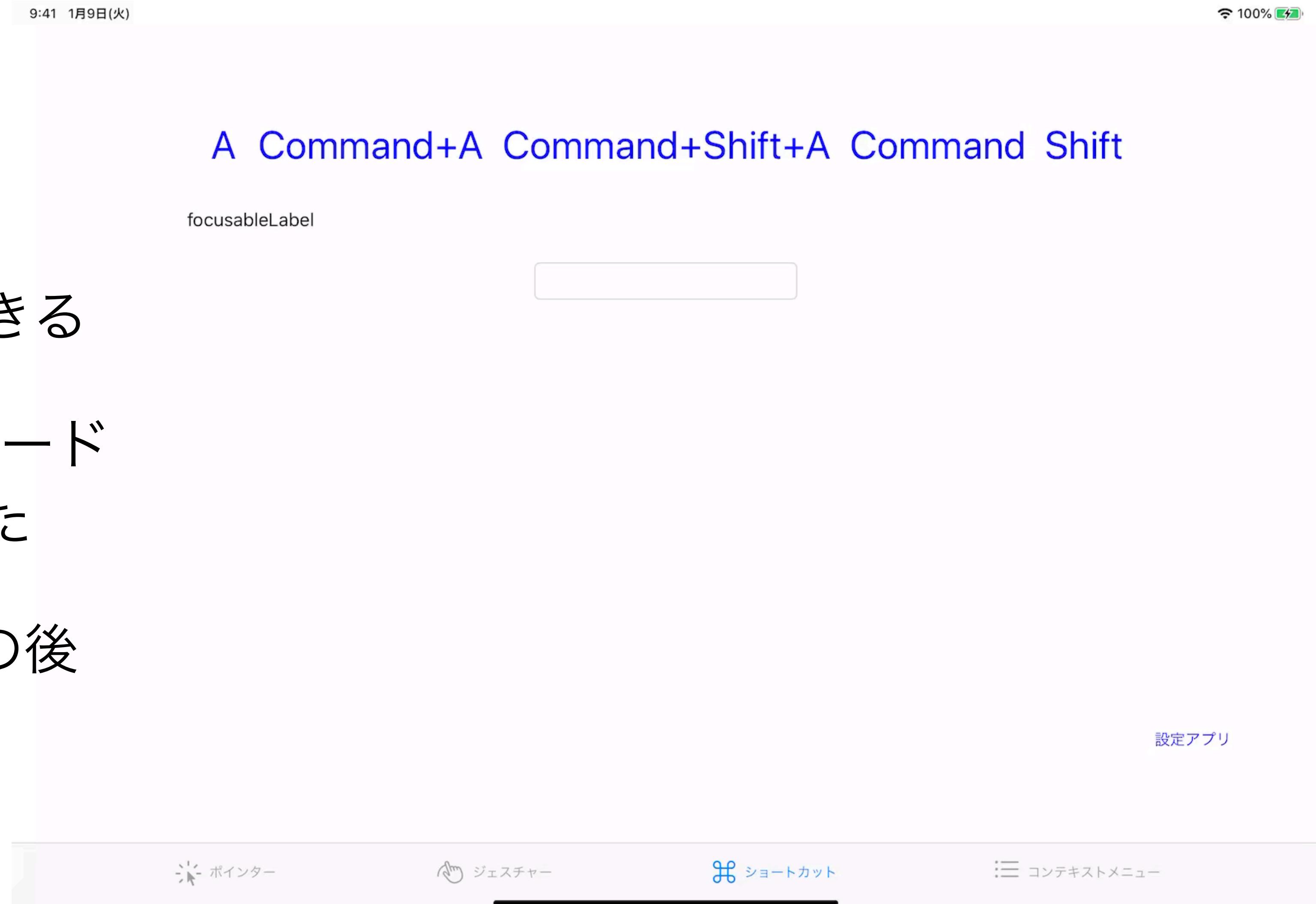
- ・フォーカスの枠線は親のviewのtintColor
- ・たぶん枠線が親のViewに描画されるから
- ・Tabでフォーカスを当てられるのはおそらく以下の2つ
  - Label
  - ・isAccessibilityItemかつaccessibilityRespondsToUserInteraction
- ・isUserInteractionEnabledかつcanBecomeFocused
- ・AccessibilityItemの検索 (Tab+F) でaccessibilityLabelが使われる



# キーボード

## iOS 13.4の新機能

- UIPressにkeyが追加された
  - pressesBegan/pressesEndedで確認できる
  - 押された/離されたキーコードが全てわかるようになった
  - フルキーボードアクセスの後
  - テキスト入力よりも前



# まとめ

## ポインター・キーボード

- ・ボタンなどにポインターインタラクションを適切に設定しよう
- ・コンテキストメニュー や ドラッグ&ドロップなどを活用していこう
- ・ジェスチャーをフル活用することで細やかな機能を提供できる
- ・ジェスチャーでキーボードの修飾キーの状態がわかるようになった
- ・キーコマンドにはdiscoverabilityTitleを設定して一覧できるようにしよう
- ・生のキーイベントが取れるようになりキーボードの活用の幅が広がった

終わり