React Native View Props

accessible

Indicates whether the view is an accessibility element. When true, the view is an accessibility element, which means it can be focused, and it should announce itself when the user navigates through the elements.

accessibilityLabel

Provides a custom text for screen readers to announce. This is useful if the content inside the view is not something that makes sense when read aloud or when the view is a touchable.

accessibilityRole

Tells accessibility services to treat the view as a specific role, such as a button, header, or link, which can provide better context to screen readers.

accessibilityHint

Provides additional context for accessibility services, explaining what happens when the view is interacted with.

accessibilityState

Describes the current state of the view, such as whether it is selected, disabled, or checked. This helps accessibility services convey the state of the view to the user.

accessibilityValue

Specifies the current value of the view, useful for components like sliders or progress bars.

accessibilityActions

Defines custom actions that can be triggered through accessibility services.

onAccessibilityAction

Callback that is invoked when an accessibility action is performed on the view.

onAccessibilityTap

Callback that is invoked when the view is tapped by an accessibility service, like VoiceOver on iOS.

onMagicTap

Callback that is invoked when a magic tap is performed, a gesture used by accessibility services.

onAccessibilityEscape

Callback that is invoked when an escape gesture is performed by an accessibility service.

importantForAccessibility

Determines whether the view should be considered important for accessibility. Options include 'auto', 'yes', 'no', and 'no-hide-descendants'.

hitSlop

Expands the touchable area of the view by adding additional space around the view's bounds.

pointerEvents

Controls how the view should respond to touch events. Options include 'box-none', 'none', 'box-only', and 'auto'.

removeClippedSubviews

Optimizes performance by removing subviews that are outside of the view's bounds from the native rendering tree.

collapsable

Determines whether the view is collapsible, which can be used to optimize layout by collapsing views that are not visible.

needsOffscreenAlphaCompositing

Determines whether the view should be rendered off-screen to preserve alpha transparency.

renderToHardwareTextureAndroid

Forces the view to be rendered to a hardware texture on Android. This can improve rendering performance in some cases.

shouldRasterizeIOS

Enables rasterization for views on iOS, which can improve performance by rendering the view as an image.

style

Allows you to apply custom styles to the view, such as layout, color, and borders.

testID

A unique identifier used in testing to locate the view in automated test scripts.

nativeID

Similar to 'testID', this is a unique identifier for the view used in native code.

onLayout

Callback that is invoked when the view's layout is calculated, providing details like width, height, and position.

accessible

Indicates whether the view is an accessibility element.

accessibilityIgnoresInvertColors

Indicates whether the view should ignore color inversion, which can be used to preserve the original colors in certain accessibility modes.

accessibilityLanguage

Specifies the language of the content within the view, useful for screen readers.