Version: 3.x

scrollTo

> INFO
>
> This page was ported from an old version of the documentation.
>
> As we're rewriting the documentation some of the pages might be a little outdated.

Provides synchronous scroll on the UI thread to a given offset using an animated ref to a scroll view. This allows performing smooth scrolling without lags (which might have otherwise occurred when it was asynchronous and based on lots of events).

Arguments

`animatedRef`

The product of `useAnimatedRef` which is Reanimated's extension of a standard React ref (delivers the view tag on the UI thread).

`x` **[Float]**

Corresponds to the pixel along the horizontal axis of the element that you want displayed in the upper left.

`y` **[Float]**

Corresponds to the pixel along the vertical axis of the element that you want displayed in the upper left.

`animated` **[Boolean]**

Indicates whether the scroll should be smooth (`true`) or instant (`false`).

Returns

`void`

Example

```
import React from 'react';
import { TouchableOpacity, View, Text, ScrollView } from 'react-native';
import {
  useAnimatedRef,
  useDerivedValue,
  useSharedValue,
  scrollTo,
} from 'react-native-reanimated';

const ITEM_COUNT = 10;
const ITEM_SIZE = 100;
const ITEM_MARGIN = 10;

export const Comp = () => {
  const aref = useAnimatedRef();
  const scroll = useSharedValue(0);

  useDerivedValue(() => {
    scrollTo(aref, 0, scroll.value * (ITEM_SIZE + 2 * ITEM_MARGIN), true);
  });

  const items = Array.from(Array(ITEM_COUNT).keys());

  const Incrementor = ({ increment }) => (
    <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
      <TouchableOpacity
        onPress={() => {
          scroll.value =
            scroll.value + increment > 0
              ? scroll.value + increment
              : ITEM_COUNT - 1 + increment;

          if (scroll.value >= ITEM_COUNT - 2) scroll.value = 0;
        }}>
        <Text>{`Scroll ${Math.abs(increment)} ${
          increment > 0 ? 'down' : 'up'
        }`}</Text>
      </TouchableOpacity>
```

```
    </View>
  );

  return (
    <View style={{ flex: 1, flexDirection: 'column' }}>
      <Incrementor increment={1} />
      <View
        style={{ width: '100%', height: (ITEM_SIZE + 2 * ITEM_MARGIN) * 2 }}>
        <ScrollView ref={aref} style={{ backgroundColor: 'orange' }}>
          {items.map((_, i) => (
            <View
              key={i}
              style={{
                backgroundColor: 'white',
                aspectRatio: 1,
                width: ITEM_SIZE,
                margin: ITEM_MARGIN,
                justifyContent: 'center',
                alignContent: 'center',
              }}>
              <Text style={{ textAlign: 'center' }}>{i}</Text>
            </View>
          ))}
        </ScrollView>
      </View>

      <Incrementor increment={-1} />
    </View>
  );
};
```

Edit this page