

Version: 3.x

measure

`measure` lets you synchronously get the dimensions and position of a view on the screen, all on the UI thread.

Reference

```
import { measure } from 'react-native-reanimated';

function App() {
  const animatedRef = useAnimatedRef();

  const handlePress = () => {
    runOnUI(() => {
      const measurement = measure(animatedRef);
      if (measurement === null) {
        return;
      }
      // ...
    })();
  };

  return <Animated.View ref={animatedRef} />;
}
```

▼ Type definitions

Arguments

`animatedRef`

An animated ref connected to the component you'd want to get the measurements from. The animated ref has to be passed either to an Animated component or a React Native built-in component.

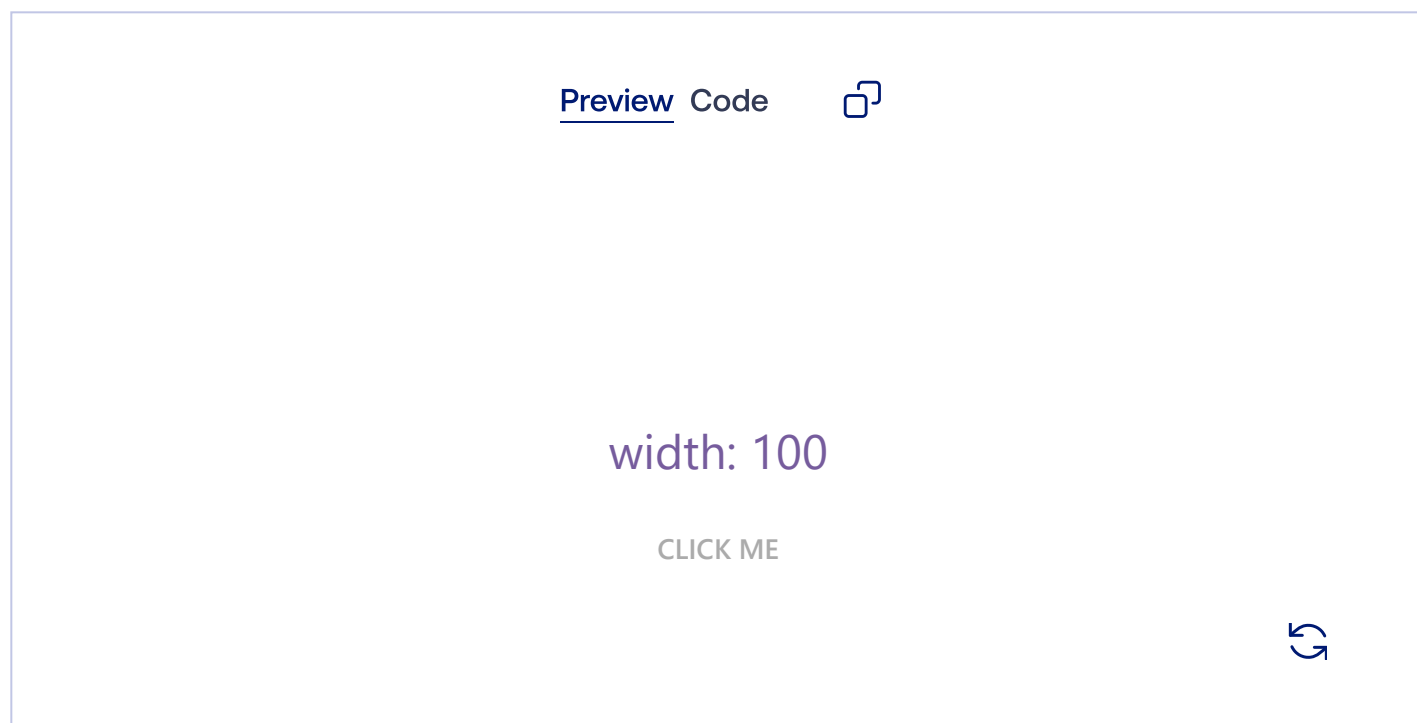
Returns

`measure` returns an object containing these fields:

- `x` a number representing X coordinate relative to the parent component,
- `y` a number representing Y coordinate relative to the parent component,
- `width` a number representing the width of the component,
- `height` a number representing the height of the component,
- `pageX` a number representing X coordinate relative to the screen,
- `pageY` a number representing Y coordinate relative to the screen,

or returns `null` when the measurement couldn't be performed.

Example



Remarks

- `measure` is implemented only on the UI thread. When using `measure` inside event handlers, it has to be wrapped with the `runOnUI` function.

- The `useAnimatedStyle` function is first evaluated on the JavaScript thread just before the views are attached to the native side. For this reason, to safely use the `measure` within `useAnimatedStyle`, a condition similar to the one below must be added to the code:

```
function App() {
  const animatedStyles = useAnimatedStyle(() => {
    if (_WORKLET) {
      // safely use measure
      const measurement = measure(animatedRef);
    }
  });
}
```

Consecutive runs of `useAnimatedStyle` are executed on the UI thread.

- When you only need the dimensions of the component and won't use the measurements during animation, consider using the `onLayout` property instead.
- Sometimes, `measure` returns `null` (e.g., when the `ref` hasn't yet attached to the view). It's best to add a `null` check after the measurement for added safety.




```
const animatedRef = useAnimatedRef();

const handlePress = () => {
  runOnUI(() => {
    const measurement = measure(animatedRef);

    if (measurement === null) {
      return;
    }
    // ...
  })();
};
```

- `measure` can be used only on rendered components. For instance, attempting to `measure` off-screen items in a `FlatList` will return a `null` value.
- `measure` isn't available with the Remote JS Debugger. We highly recommend using Chrome DevTools (also known as `chrome://inspect`) for debugging React Native apps.

Platform compatibility

Android	iOS	Web
		

 [Edit this page](#)