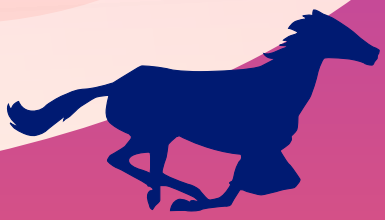
React Native Reanimated

Create smooth animations with an excellent developer experience.



Get started →

Why Reanimated?

DECLARATIVE

Reanimated comes with declarative API for creating animations. Complexity reduced from tens of methods to just a few. Define what the animation should look like and leave Reanimated to animate the styles and properties for you.

PERFORMANT

Reanimated lets you define animations in plain JavaScript which run natively on the UI thread by default. Smooth animations and interactions up to 120 fps and beyond. Reanimated delivers a native experience your users deserve.

FEATURE-RICH

Reanimated's power doesn't end on animating only simple views or images. Hook your animations into device sensors or keyboard. Create amazing experiences using Layout Animations or animate elements between navigation screens with ease.

Learn more about the features in the newest article about Reanimated 4

See blog post →

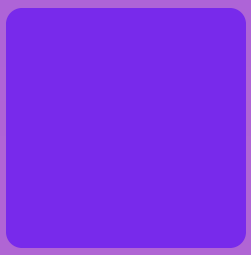


```
function App() {
  const width = useSharedValue(100);
  const handlePress = () => {
    width.value = withSpring(width.value + 50);
  };
  return <Animated.View style={{ ...styles.box, width }} />
}
```

Animations

Animate every React Native prop on iOS, Android and the Web up to 120 fps.

Check docs →



Grab and drag the square

```
import { Gesture, GestureDetector } from ?"react-native-
gesture-handler";

function App() {
  const pan = Gesture.Pan();

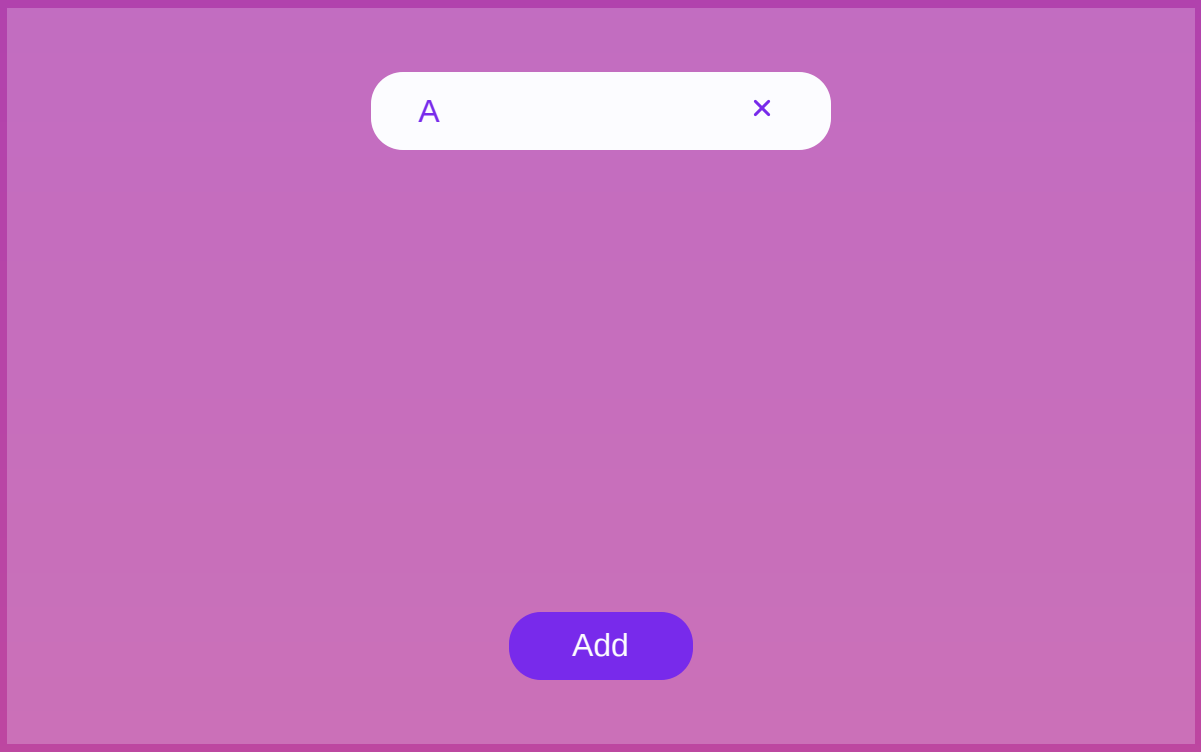
  return (
    <GestureDetector gesture={pan}>
      <Animated.View />
    </GestureDetector>
```

```
    );
  }
```

Gestures

Gesture smoothly thanks to Reanimated's integration with React Native Gesture Handler.

Check docs →

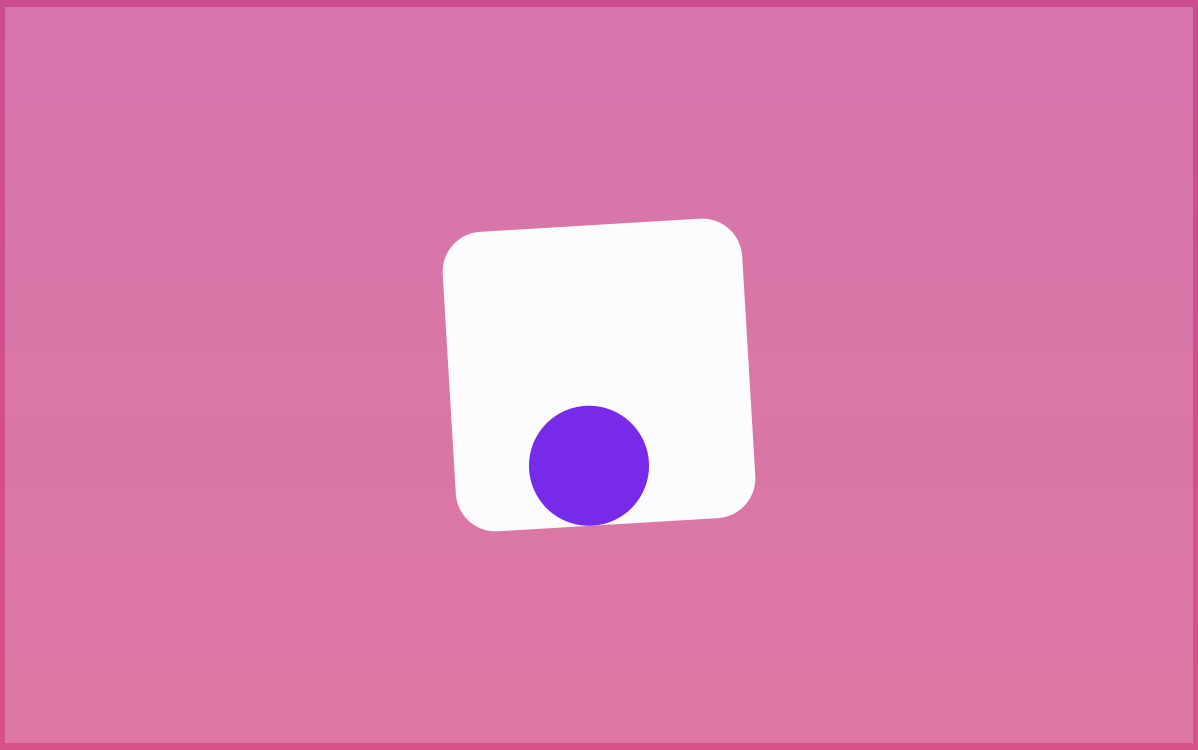



```
function App() {
  return <Animated.View entering={FadeIn} exiting={FadeOut}
/>;
}
```

Layout animations

Animate views when they are added and removed from the view hierarchy. Just like that.

Check docs →



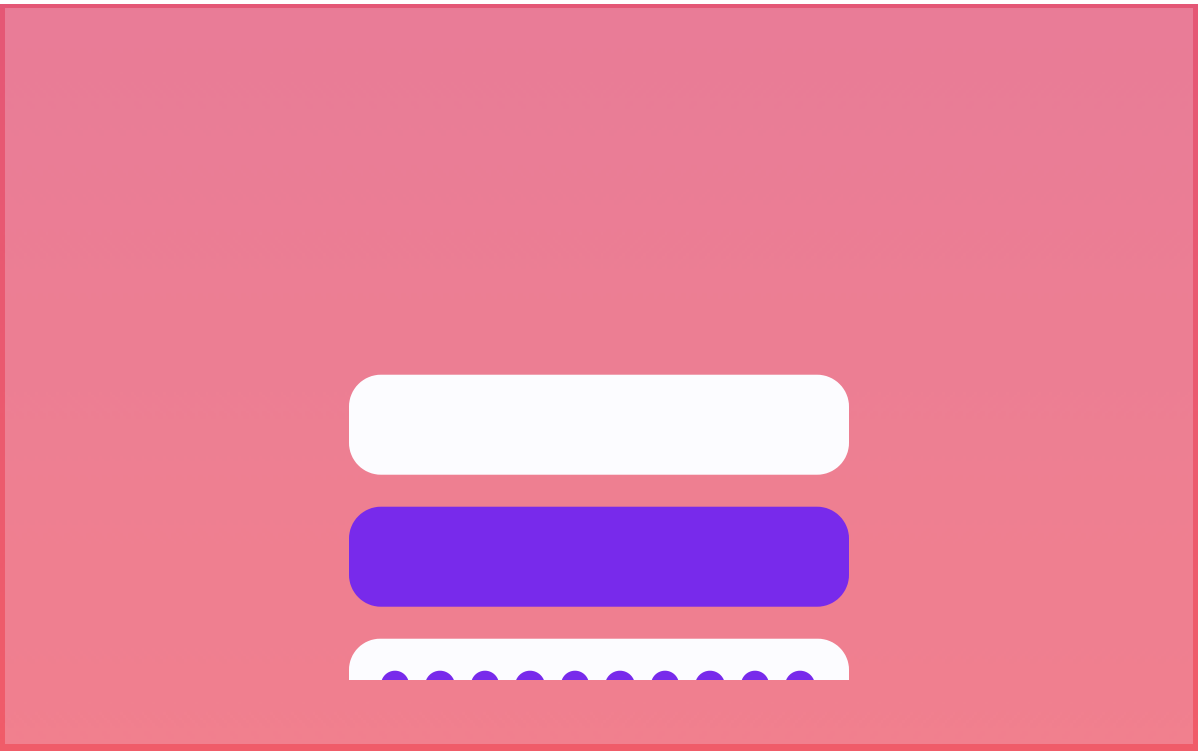
```
const gyroscope = useAnimatedSensor(SensorType.GYROSCOPE);

useDerivedValue(() => {
  const { x, y, z } = gyroscope.sensor.value;
});
```

Sensor-based animations

Connect your animations to a gyroscope or accelerometer with just one hook. It's that easy.

Check docs →



```
function App() {
  const keyboard = useAnimatedKeyboard();
  const animatedStyle = useAnimatedStyle(() => ({
    transform: [{ translateY: -keyboard.height.value }],
  });
  //...
}
```

Keyboard-based animations

Create animations based on the device keyboard state and position.

Check docs →

Testimonials



Marc Rousavy

MARGELO

“We’ve used Reanimated and Gesture Handler for a ton of apps already - it’s amazingly simple yet powerful! 😍”



Andrew Lo

SHOPIFY

“They enable us app devs to make our users feel delight, have fun, and enjoy using our apps more.”

We are Software Mansion

React Native Core Contributors and experts in dealing with all kinds of React Native issues. No matter if you need help with gestures, animations or React Native development we can help.

Hire us →