

Animations

React Native Skia offers integration with [Reanimated v3 and above](#), enabling the execution of animations on the UI thread.

React Native Skia supports the direct usage of Reanimated's shared and derived values as properties. There is no need for functions like `createAnimatedComponent` or `useAnimatedProps`; simply pass the Reanimated values directly as properties.

```
import {useEffect} from "react";
import {Canvas, Circle, Group} from "@shopify/react-native-skia";
import {
  useDerivedValue,
  useSharedValue,
  withRepeat,
  withTiming,
} from "react-native-reanimated";

export const HelloWorld = () => {
  const size = 256;
  const r = useSharedValue(0);
  const c = useDerivedValue(() => size - r.value);
  useEffect(() => {
    r.value = withRepeat(withTiming(size * 0.33, { duration: 1000 }), -1);
  }, [r, size]);
  return (
    <Canvas style={{ flex: 1 }}>
      <Group blendMode="multiply">
        <Circle cx={r} cy={r} r={r} color="cyan" />
        <Circle cx={c} cy={r} r={r} color="magenta" />
        <Circle
          cx={size/2}
          cy={c}
          r={r}
          color="yellow"
        />
      </Group>
    </Canvas>
  );
}
```

```
);  
};
```

We offer some [Skia specific animation hooks](#), especially for paths.

Colors

For colors, React Native Skia uses a different storage format from Reanimated. This means that `interpolateColor` from Reanimated won't work out of the box. Instead you can use `interpolateColors` from React Native Skia.

```
import {  
  Canvas,  
  LinearGradient,  
  Fill,  
  // Use this function instead of interpolateColor from Reanimated  
  interpolateColors,  
  vec,  
} from "@shopify/react-native-skia";  
import { useEffect } from "react";  
import { useWindowDimensions } from "react-native";  
import {  
  useDerivedValue,  
  useSharedValue,  
  withRepeat,  
  withTiming,  
} from "react-native-reanimated";  
  
const startColors = [  
  "rgba(34, 193, 195, 0.4)",  
  "rgba(34,193,195,0.4)",  
  "rgba(63,94,251,1)",  
  "rgba(253,29,29,0.4)",  
];  
const endColors = [  
  "rgba(0,212,255,0.4)",  
  "rgba(253,187,45,0.4)",  
  "rgba(252,70,107,1)",  
  "rgba(252,176,69,0.4)",  
];  
  
export const AnimatedGradient = () => {  
  const { width, height } = useWindowDimensions();  
}
```

```
const colorsIndex = useSharedValue(0);
useEffect(() => {
  colorsIndex.value = withRepeat(
    withTiming(startColors.length - 1, {
      duration: 4000,
    }),
    -1,
    true
  );
}, []);
const gradientColors = useDerivedValue(() => {
  return [
    interpolateColors(colorsIndex.value, [0, 1, 2, 3], startColors),
    interpolateColors(colorsIndex.value, [0, 1, 2, 3], endColors),
  ];
});
return (
  <Canvas style={{ flex: 1 }}>
    <Fill>
      <LinearGradient
        start={vec(0, 0)}
        end={vec(width, height)}
        colors={gradientColors}
      />
    </Fill>
  </Canvas>
);
};
```

 [Edit this page](#)