Animations

Animations

Creating an animation

1. import Animated

```
import { Animated, ... } from 'react-native';
```

2. create animated value within the class

```
animatedMargin = new Animated.Value(350);
```

3. Declare Animated component in render

```
<Animated.View />
```

Animations

Creating an animation

4.Add animated value as style

```
<Animated.View style={{ marginTop: this.animatedMargin }} />
```

5.Trigger animation

```
animate = () => {
  Animated.timing(
    this.animatedMargin,
    {
      toValue: 100,
      duration: 1700,
    }
  ).start()
}
```

Animations

Three main Animated methods that you can use to create animations:

- `Animated.timing()` — Maps time range to easing value.
- `Animated.decay()` — starts with an initial velocity and gradually slows to a stop.
- `Animated.spring()` —single-spring physics model (Based on Rebound and Origami).

We will be covering `Animated.timing()` and `Animated.spring()` as they are the most used (almost exclusively).

Animated.timing() is by far the most used.

Animations

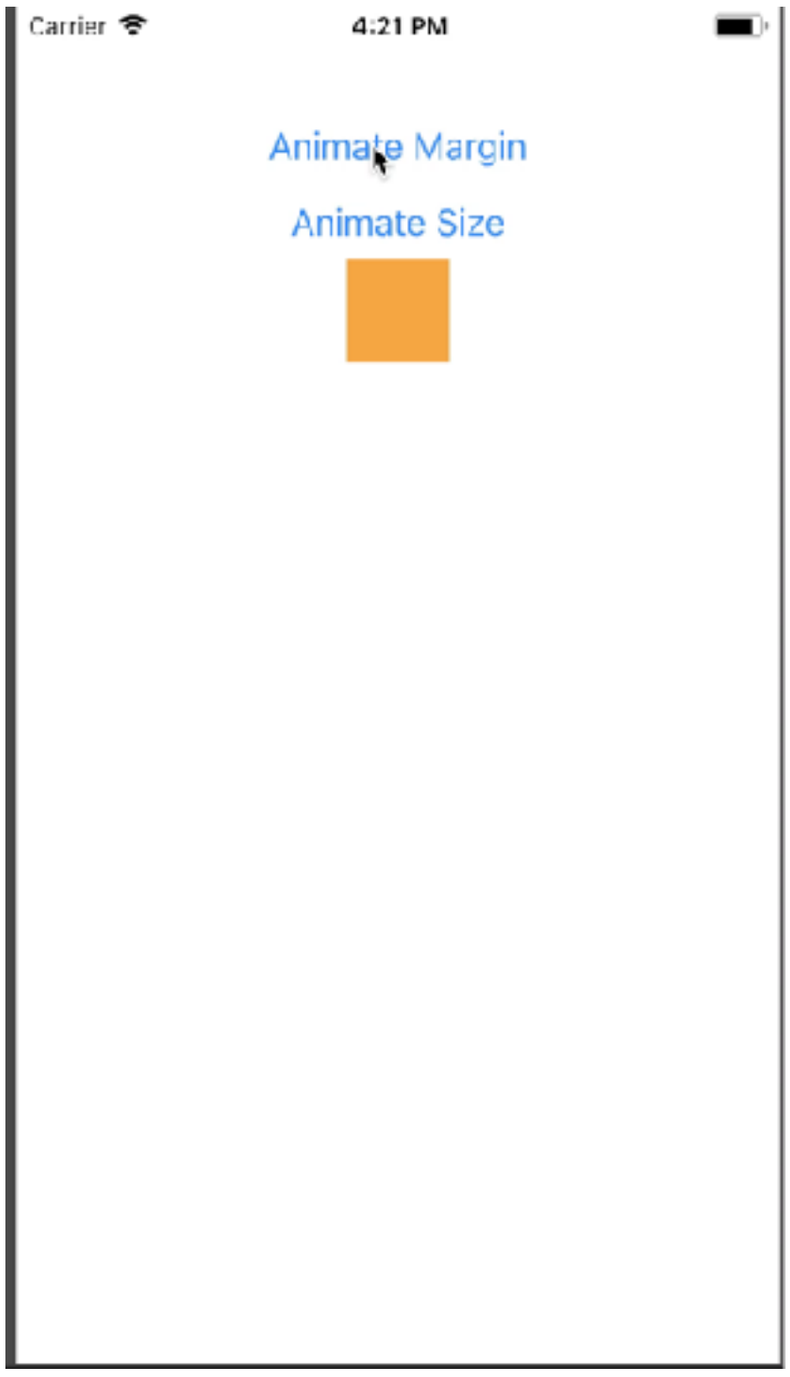
`Animated.timing()`

```
Animated.timing(
  this.animatedButton,
  {
    toValue: 1,
    duration: 1700,
  },
).start()
```

Animations Demo Parallel

https://github.com/hgale/AnimationsDemo

Animations Demo Timing



Animations

Four main Animatable components

`Animated.Image`

`Animated.View`

`Animated.Text`

`Animated.ScrollView`

Any component can be made into an animatable component using the `createAnimatedComponent()` method:

```
const AnimatedButton = Animated.createAnimatedComponent(TouchableHighlight);
```

Animations

Along with these three Animated methods, there are three ways to call these animations along with calling them individually. We will be covering all three of these as well:

- `Animated.parallel()` — Starts an array of animations all at the same time.
- `Animated.sequence()` — Starts an array of animations in order, waiting for each to complete before starting the next. If the current running animation is stopped, no following animations will be started.
- `Animated.stagger()` — Array of animations may run in parallel (overlap), but are started in sequence with successive delays. Very similar to Animated.parallel() but allows you to add delays to the animations.

Animated.parallel()

Takes an array of animations. Starts a number of animations at the same time.

```
Animated.parallel([
  Animated.timing(
    this.animatedWelcome,
    {
      toValue: 1,
      duration: 740,
    },
  ),
  Animated.timing(
    this.animatedButton,
    {
      toValue: 1,
      duration: 1700,
    },
  ),
]).start();
```

Animations Demo Parallel

Animations Demo Parallel

https://github.com/hgale/AnimationsDemo/pull/1

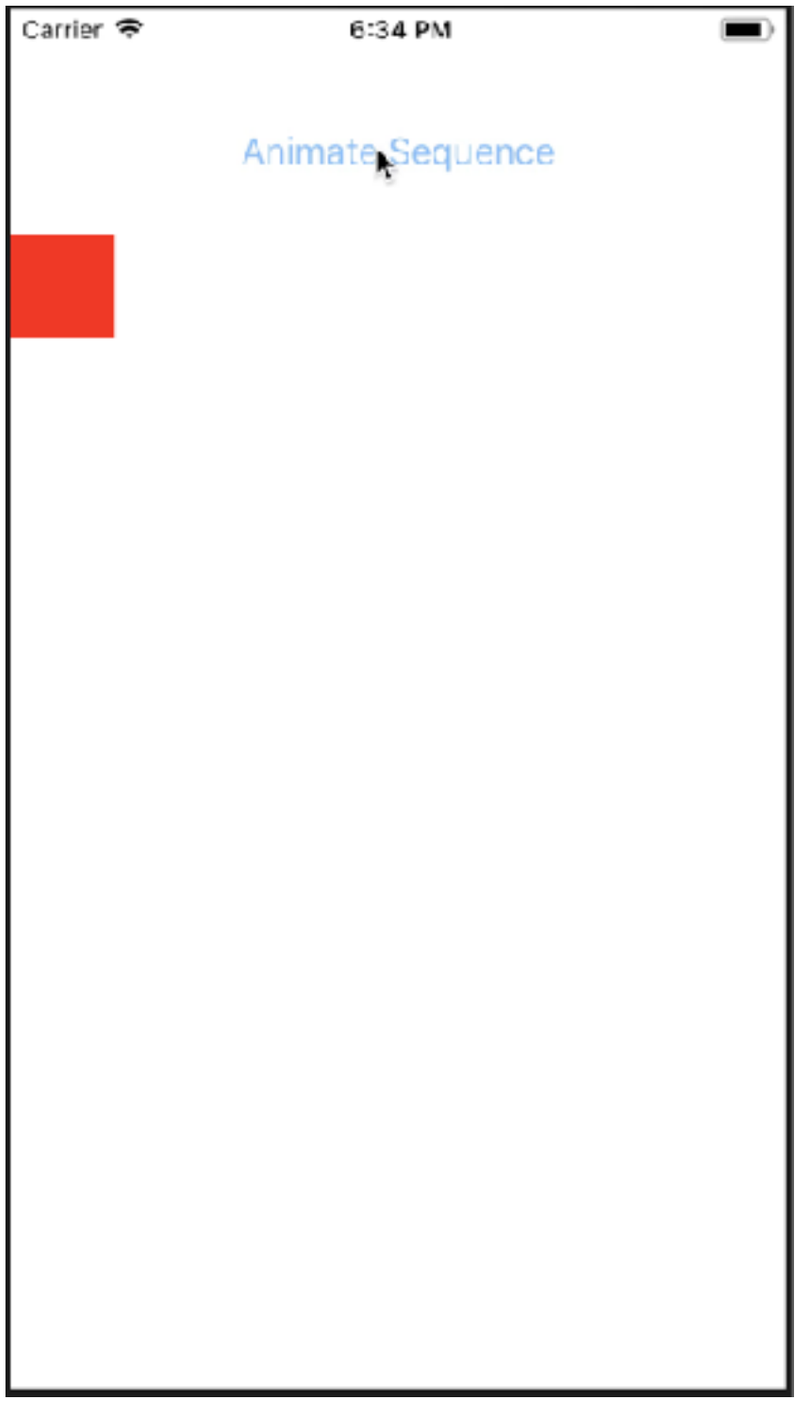
Animations

Animated.sequence()

Takes an array of animations. Starts the animations in order, waiting for each to complete before starting the next.

```
Animated.sequence([
  Animated.timing(
    this.animatedMarginTop,
    {
      toValue: 516,
      duration: 1000,
    },
  ),
  Animated.timing(
    this.animatedMarginLeft,
    {
      toValue: 325,
      duration: 1000,
    },
  ),
]).start()
```

Animations Demo sequence



Animations Demo sequence

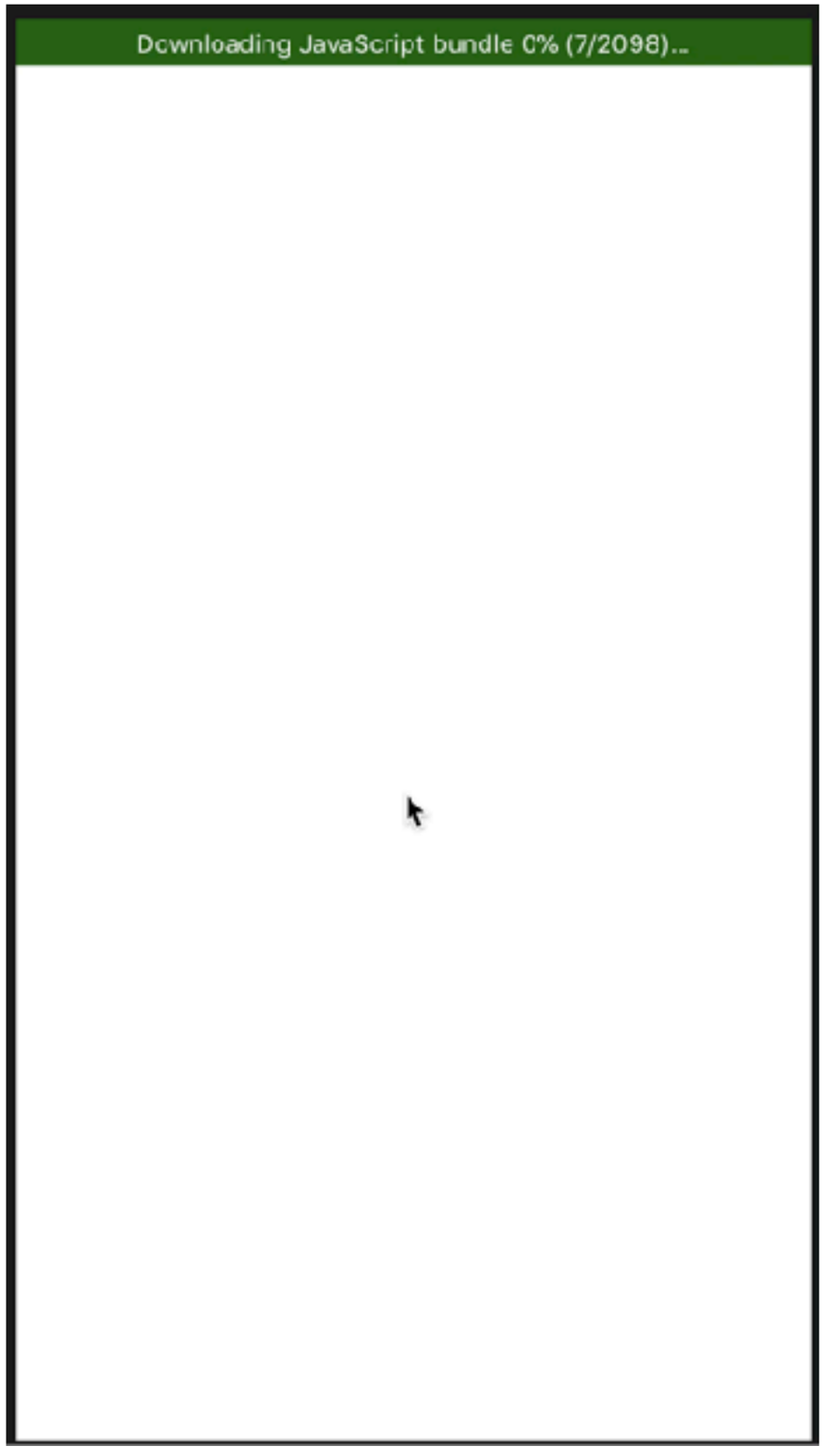
https://github.com/hgale/AnimationsDemo/pull/2

Animated.stagger()

Takes an array of animations. Starts animations in order and in parallel, but with successive delays.

```
Animated.stagger([
  Animated.timing(
    this.animatedMarginTop,
    {
      toValue: 516,
      duration: 1000,
    },
  ),
  Animated.timing(
    this.animatedMarginLeft,
    {
      toValue: 325,
      duration: 1000,
    },
  ),
]).start()
```

Animations Demo Stagger



Animations Demo Stagger

https://github.com/hgale/AnimationsDemo/pull/3

Animations

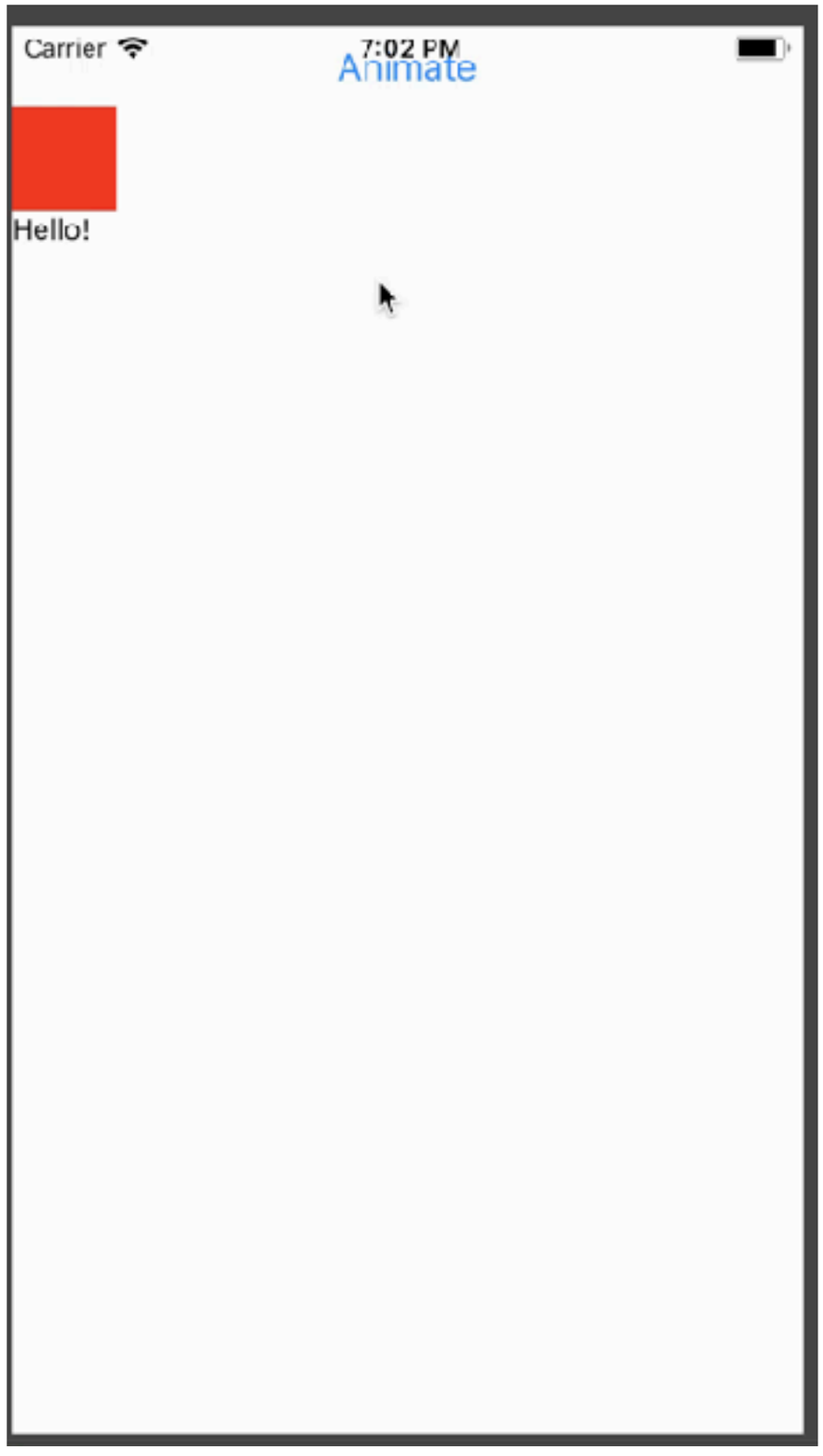
Interpolation

The interpolate() function allows input ranges to map to different output ranges.

```
const width = this.animatedWidthValue.interpolate({
  inputRange: [0, 1],
  outputRange: [50, 500],
});

<View style={{ width: width }} />
```

Animations Demo Interpolation



Animations Demo Interpolation

https://github.com/hgale/AnimationsDemo/pull/4

Animations

Easing

pre-defined animations

- bounce
- cubic
- back(number)
- elastic(number)
- ease
- quad
- sin
- linear
- circle
- bezier(x1, x2, y1, y2)
- exp
- in
- inOut(ease)
- out(ease)

Other resources

React Native Animations Using the Animated API

by Nader Dabit

https://medium.com/react-native-training/react-native-animations-using-the-animated-api-ebe8e0669fae

https://facebook.github.io/react-native/docs/animations.html