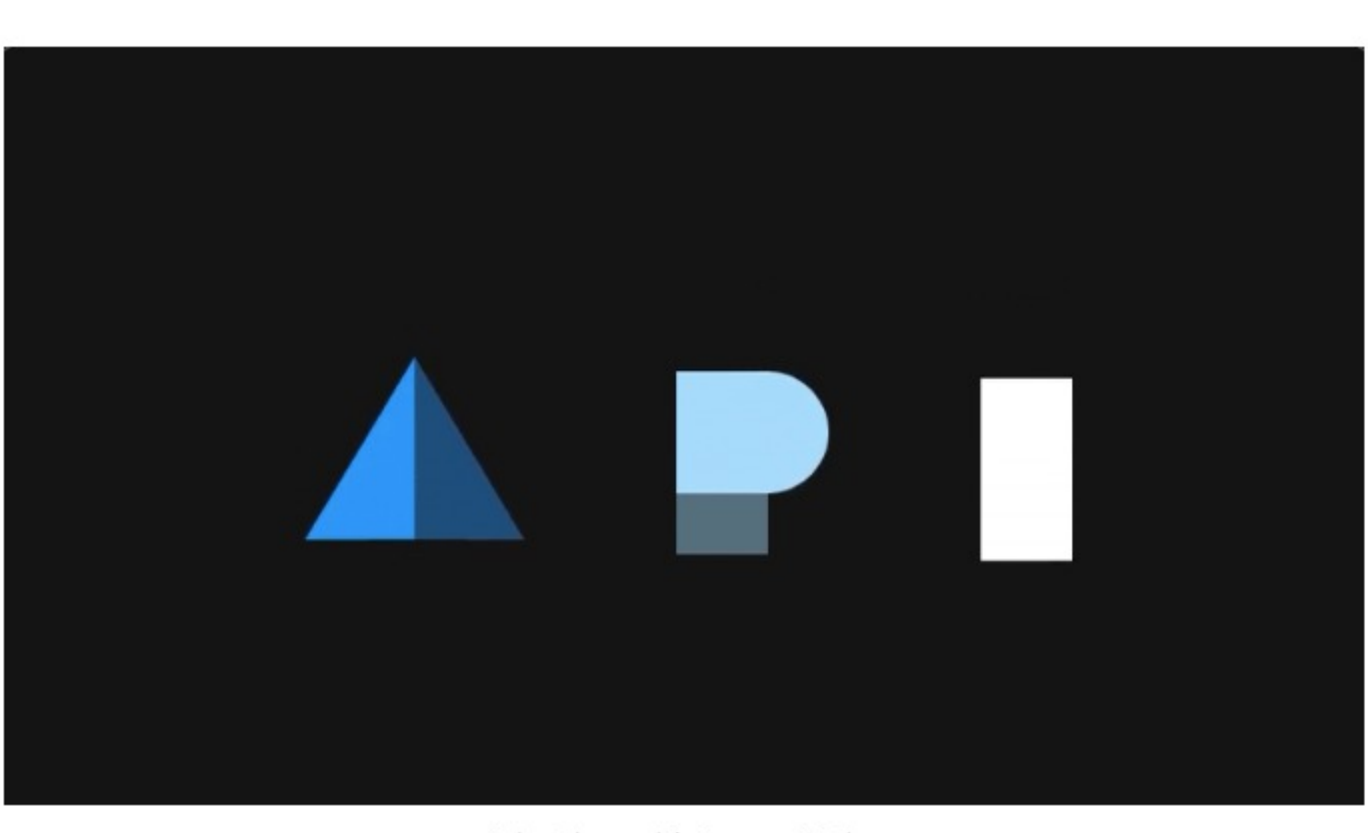
Smooth Animations With React and Framer Motion

Frame Motion is an open source React library to power production-ready animations

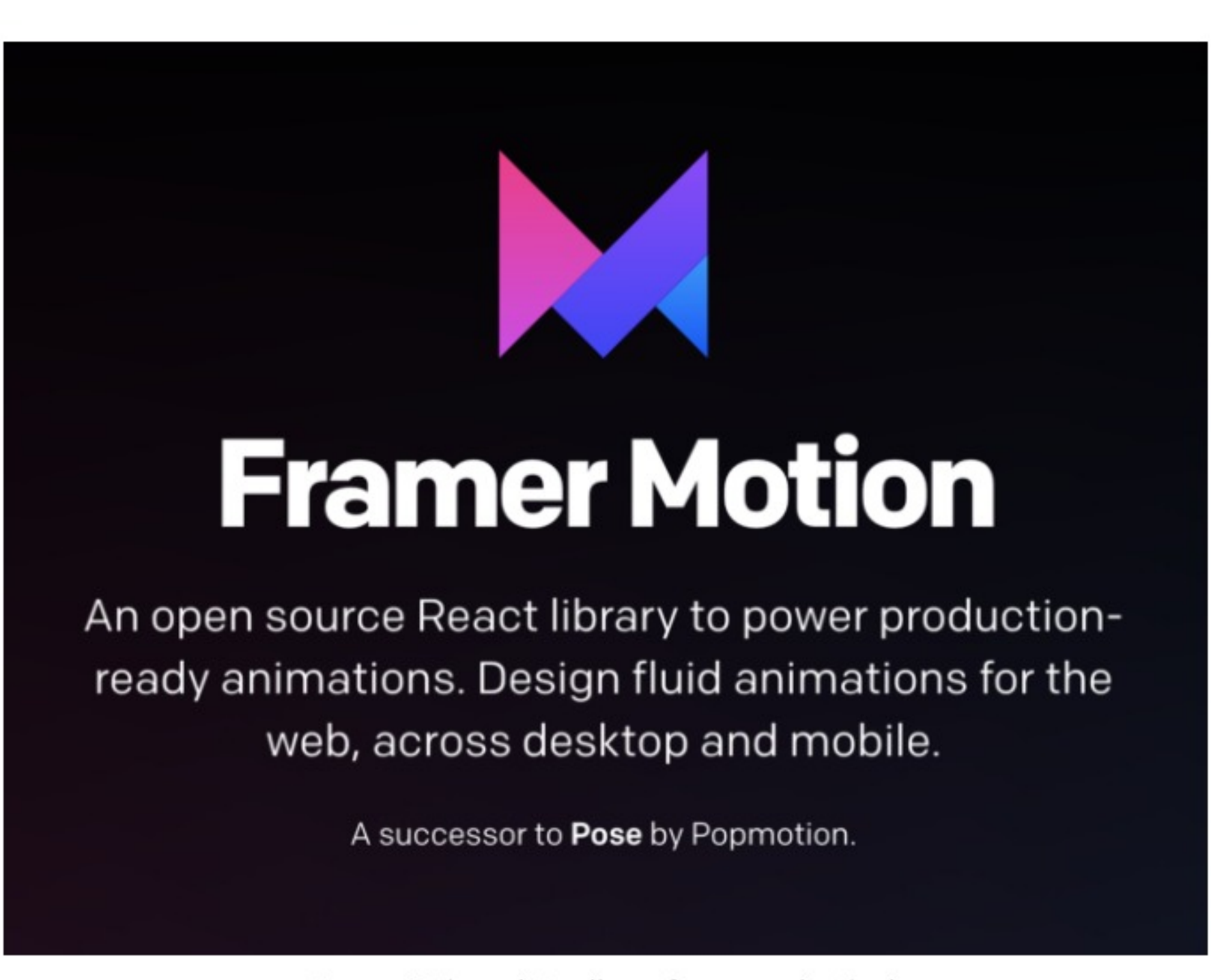
Indrek Lasn in Better Programming Follow
Jul 12 · 4 min read ★



Animations with Framer Motion

What is Framer Motion?

Framer Motion is the successor to the Pose animation library. Like Pose, it provides a declarative API to power animations and gestures in your React app. Motion attempts to make the API both simpler for the simplest cases and more flexible to handle advanced cases.



Framer Motion — https://www.framer.com/motion/

Framer Motion is particularly powerful because it solves another huge problem in design: The handoff.

Designers often spend days perfecting animations down to the last spring, only to find that the finer details get lost during development. By using the same animation library in prototyping and production, you're now guaranteed a truer and more efficient way to ensure your animations are always one-to-one.

. . .

Getting Started

Note: Framer Motion requires ***version React 16.8 or greater****.*

Installation

Install `framer-motion` from npm.

```
npm install framer-motion
```

Once installed, you can import Motion into your components via `framer-motion` package.

```
import { motion } from "framer-motion"
```

frameMotion.js hosted with ❤ by GitHub view raw

We can replace regular divs with the `motion.div` elements. Using the motion elements gives us access to many properties we can use to animate.

. . .

Animating

Let's take this super simple React and Framer Motion component. When the component mounts, it will rotate 360 degrees in two seconds.

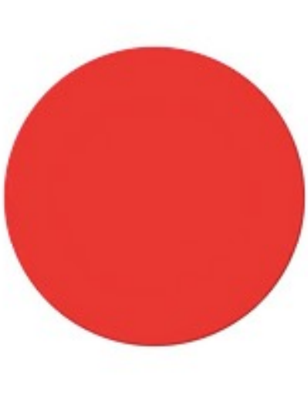
```
import React from "react";
import { motion } from "framer-motion";

const styles = {
  background: "blue",
  borderRadius: 30,
  width: 150,
  height: 150,
  margin: "auto"
};

export const Rotate = () => (
  <motion.div
    style={styles}
    animate={{ rotate: 360 }}
    transition={{ duration: 2 }}
  />
);
```

rotate.js hosted with ❤ by GitHub view raw

Rotating elements

The `animate` property can accept an object of values. When one of them changes, the `motion` component will automatically animate to the new state.

Let's take the pop example, where our element will "pop" in once the component mounts.



"pop"

```
import React from "react";
import { motion } from "framer-motion";

const styles = {
  background: "red",
  borderRadius: 30,
  width: 50,
  height: 50,
  margin: "auto"
};

export const Pop = () => (
  <motion.div
    style={styles}
    animate={{ scale: 3 }}
    transition={{ duration: 0.25 }}
  />
);
```

pop.js hosted with ❤ by GitHub view raw

Notice how we specify what happens with the element inside the `animate` property. This is where we give instructions on how to animate. The `transition` property is where we tell the duration of the animation.

We can even specify multiple values for each `transition`, `animate` properties, and styles by passing the values as an array. This gives us the option to combine the "pop" and rotate animation into one animation.

"rotate-pop"

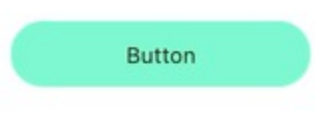
```
import React from "react";
import { motion } from "framer-motion";

const styles = {
  background: "#035ee8",
  borderRadius: 30,
  width: 100,
  height: 100,
  margin: "auto"
};

export const Pop = () => (
  <motion.div
    style={styles}
    animate={{
      scale: [1, 2, 2, 1, 1],
      rotate: [0, 160, 270, 360, 0],
      borderRadius: ["20%", "50%", "20%", "50%", "20%"],
    }}
    transition={{ duration: 2 }}
  />
);
```

rotate-pop.js hosted with ❤ by GitHub view raw

. . .

Gesture Animations

Motion provides `whileHover` and `whileTap` helper props that will temporarily animate a component to a visual state while a gesture is active.



Handling gestures such as hovering and tapping on a button

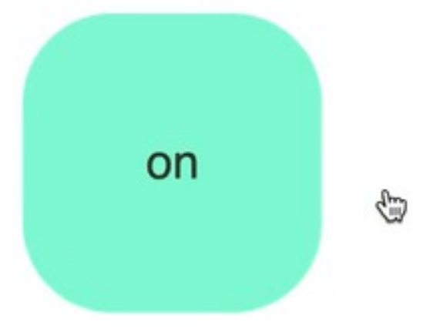
```
import React from "react";
import { motion } from "framer-motion";

const styles = {
  background: "#7fffd4",
  borderRadius: 30,
  width: 150,
  padding: "10px 20px",
  margin: "auto",
  color: "#333",
  outline: "none",
  border: "none",
  cursor: "pointer"
};

export const Gesture = () => (
  <motion.button
    style={styles}
    whileHover={{ scale: 1.1 }}
    whileTap={{ scale: 0.9, x: "-5px", y: "5px" }}
  >
    Button
  </motion.button>
);
```

gestures.js hosted with ❤ by GitHub view raw

Notice how simple it is to animate elements. Now that we have learned the basics of Framer Motion, let's jump into a more complicated example.

. . .

Toggling



Most toggles have two states: either they're on or off, `true` or `false` . Let's create a toggle component that turns off or on, depending on its current state.

```
import React, { useState } from "react";
import { motion } from "framer-motion";

const styles = {
  borderRadius: 30,
  width: 100,
  height: 100,
  margin: "auto",
  display: "flex",
  justifyContent: "center",
  alignItems: "center",
  color: "white",
  cursor: "pointer"
};

const variants = {
  active: {
    opacity: 1,
    background: "#7fffd4",
    x: "-50px",
    scale: 1.5,
    color: "#333"
  },
  inActive: {
    opacity: 1,
    background: "#f95c5c",
    x: "50px",
    scale: 1,
    color: "white"
  }
};

export const Toggle = () => {
  const [isToggled, setToggle] = useState(false);
  return (
    <motion.div
      onClick={() => setToggle(!isToggled)}
      style={styles}
      animate={isToggled ? "active" : "inActive"}
      variants={variants}
    >
      <span>{isToggled ? "on" : "off"}</span>
    </motion.div>
  );
};
```

toggle.js hosted with ❤ by GitHub view raw

toggle.js component

The key point to take away here is the `variant` prop on our motion element. Variants let us define multiple states for our animations.

Variants are pre-defined visual states that a component can be in. By giving a component and its children `variants` with matching names, whole React trees can be animated by changing a single prop.

By using variants, a parent can easily orchestrate the animations of its children with special `transition` props like `staggerChildren`.

Variants can also be dynamic functions that return different props based on data passed to each component's `custom` prop.

Note: Notice how we're using hooks. If hooks are new to you, check out this article.

Demystifying React Hooks
You probably heard about the new concept for React called hooks. Hooks were released in React version 16.8 and they let…
codeburst.io

. . .

Where to go Next

Check out the full list of properties, custom hooks, and cool animations you can do with Framer Motion. It's an animation library well worth learning, and it will boost your app visuals by a huge margin.

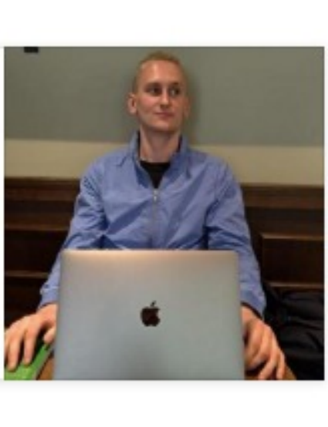
Framer Motion API
An open source, production-ready motion library for React on the web
www.framer.com

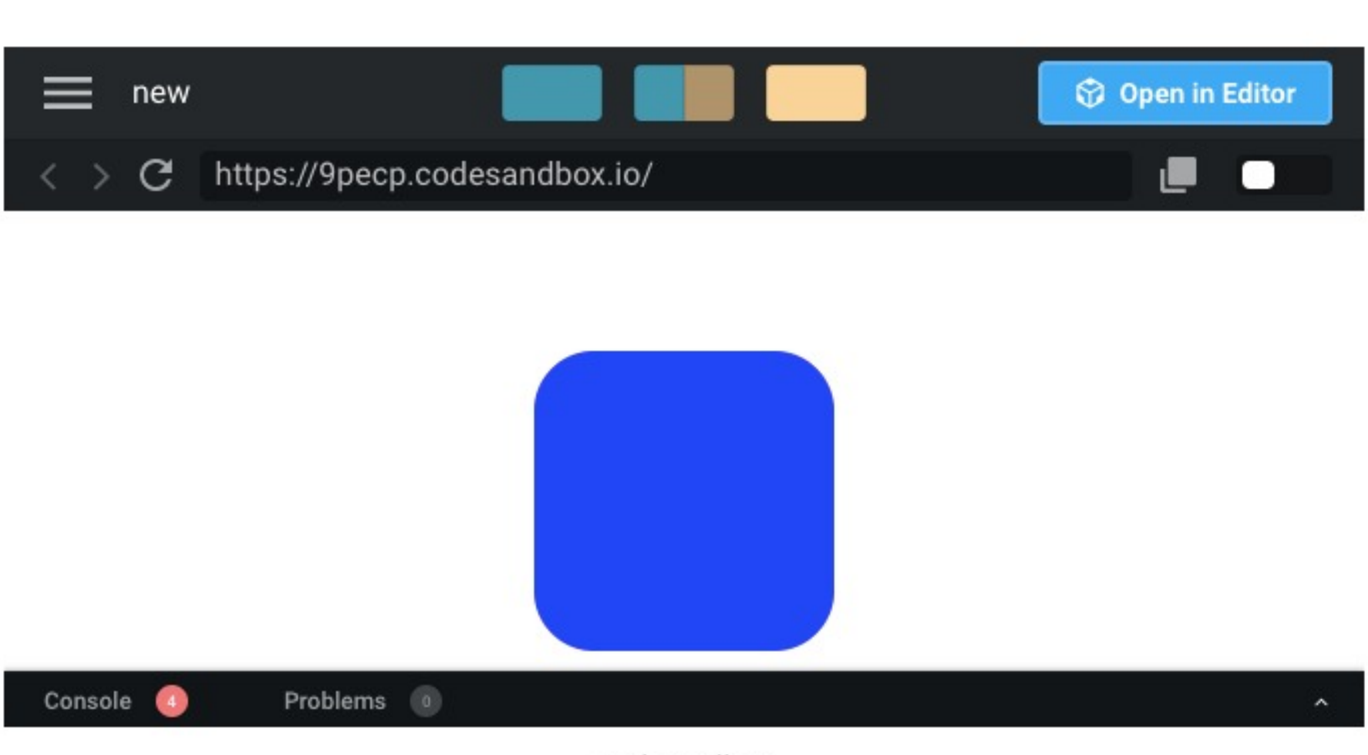
. . .

Source Code

Here's the source code and the code sandbox if you want to try playing around with Framer Motion.

indreklasn/react-framer-motion-examples
Created with CodeSandbox. Contribute to indreklasn/react-framer-motion-examples development by creating an account on…
github.com





Codesandbox

Thanks for reading!

React JavaScript Animation CSS Design

104 claps

WRITTEN BY
Indrek Lasn Follow
Simplicity matters. Follow me on Twitter @ https://twitter.com/lasnindrek

Better Programming Follow
Advice for programmers.