Animation library
Framer

{CODENATION}

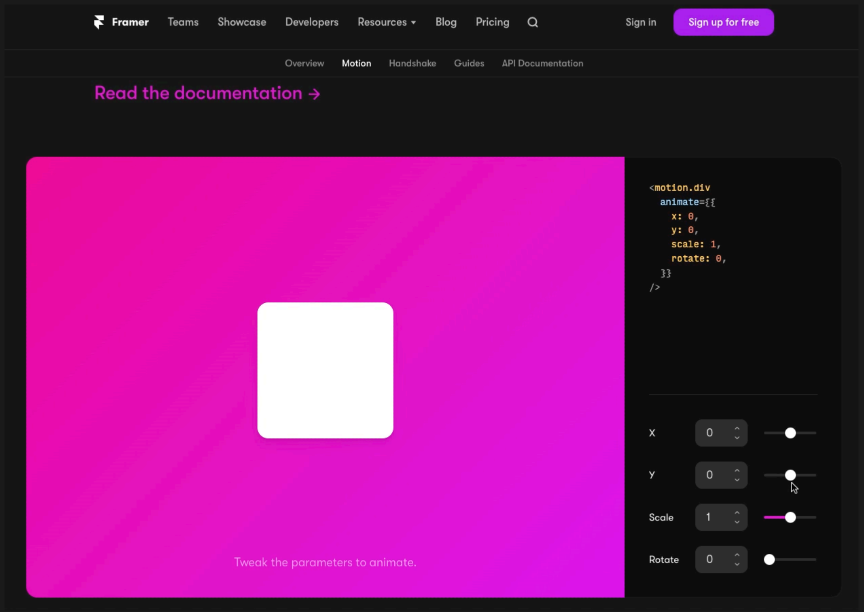
What is it?

Framer Motion is an animation library for react

No need for complex css animations, couple of lines of code and we can be up and running.

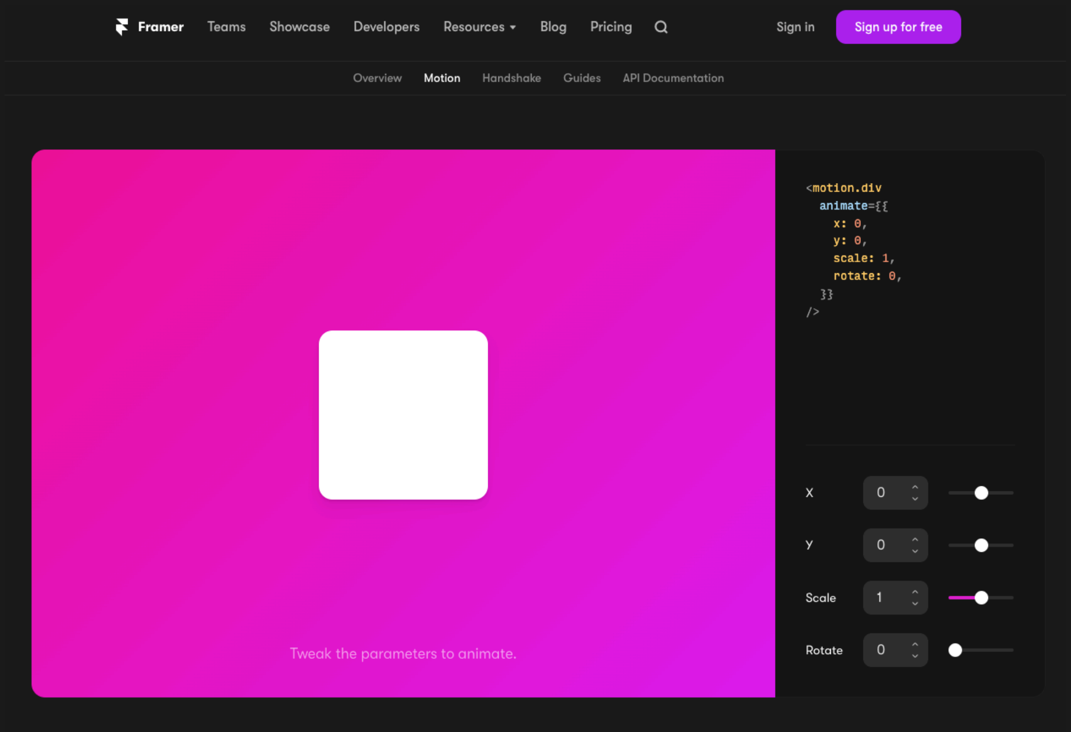
Framer

- If we check the docs we can get an idea of what its all about and how we can use it



```
<motion.div
  animate={{
    x: 0,
    y: 0,
    scale: 1,
    rotate: 0,
  }}
/>
```

X	0
Y	0
Scale	1
Rotate	0



```
<motion.div
  animate={{
    x: 0,
    y: 0,
    scale: 1,
    rotate: 0,
  }}
/>
```

X	0
Y	0
Scale	1
Rotate	0

Framer

{CN}™

- Install framer like any other npm package
- Then import { motion } from the 'framer-motion' library
- This way, when we want to use framer for a specific element, we make it into a motion element
- The element now comes with added abilities to animate it

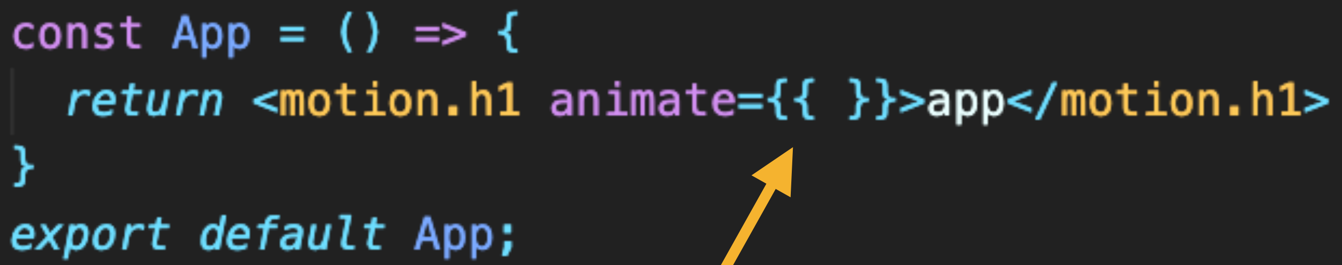
```
import { motion } from 'framer-motion'


const App = () => {
  return <motion.h1>app</motion.h1>
}
export default App;
```

Framer - what do now?

animate

- animate is a property
- For simple animations we can set value directly in this prop

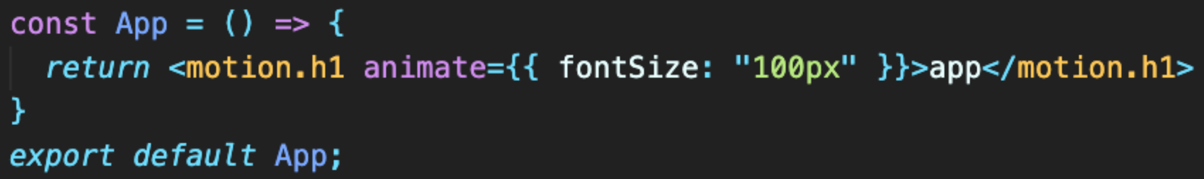


```
const App = () => {
  return <motion.h1 animate={{ }}>app</motion.h1>
}
export default App;
```

Double braces: its an object and we are now in JSX (remember any kind of expression, variable or object in JSX needs a surrounding { })

Framer - what do now? animate

- We can now, add a css property to this object and give it a value
- Framer will animate now from one value (starting) to another

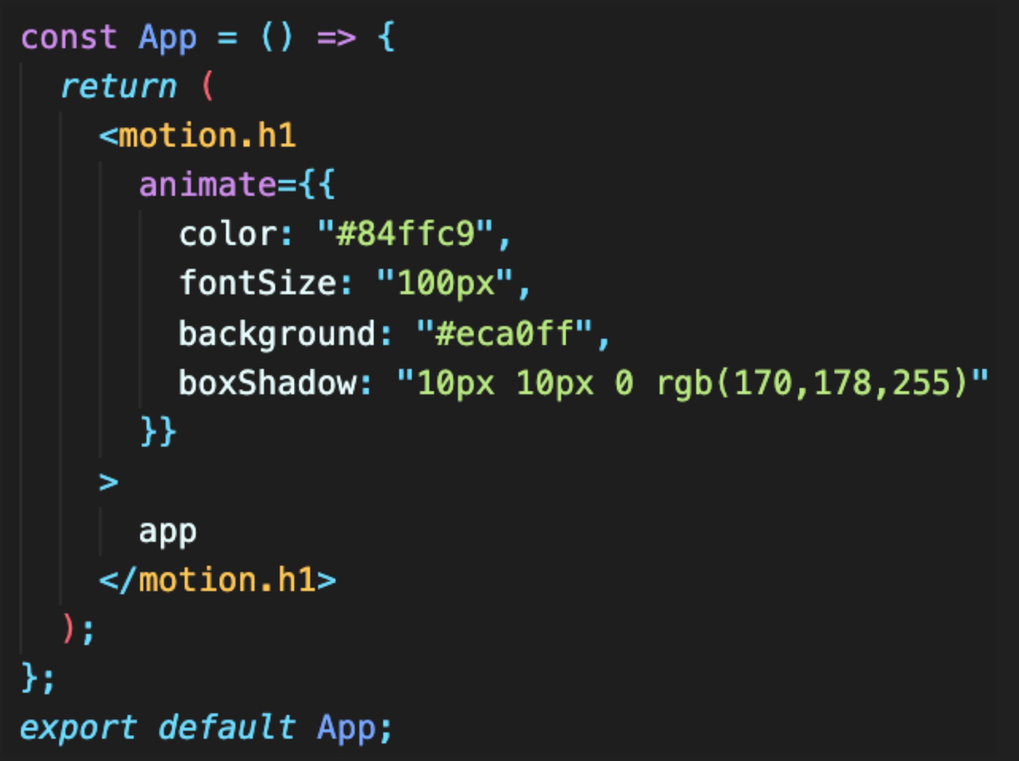


```
const App = () => {
  return <motion.h1 animate={{ fontSize: "100px" }}>app</motion.h1>
}
export default App;
```

app

Framer – what do now? animate

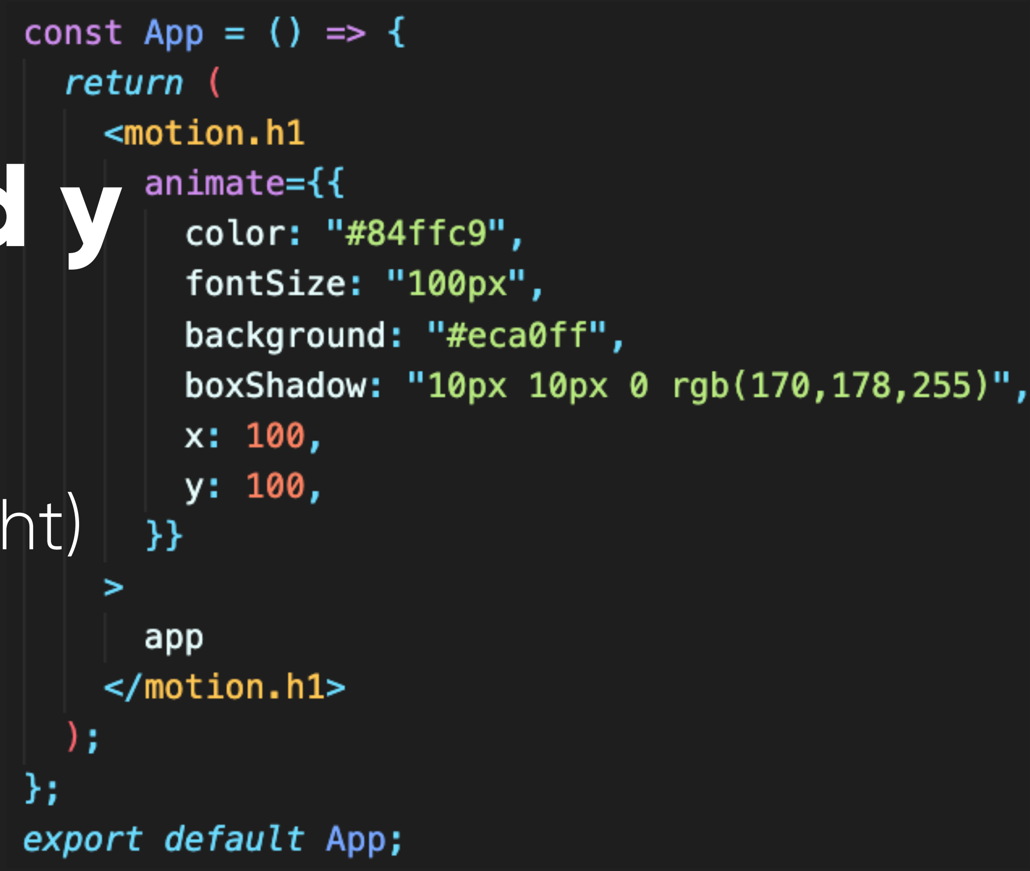
- We can add more if we want
- Things to note:
  - Camel casing
  - No strings for colors (RGB, hex and HSLA)



```
const App = () => {
  return (
    <motion.h1
      animate={{
        color: "#84ffc9",
        fontSize: "100px",
        background: "#eca0ff",
        boxShadow: "10px 10px 0 rgb(170,178,255)"
      }}
    >
      app
    </motion.h1>
  );
};
export default App;
```

Framer - what do now? animate

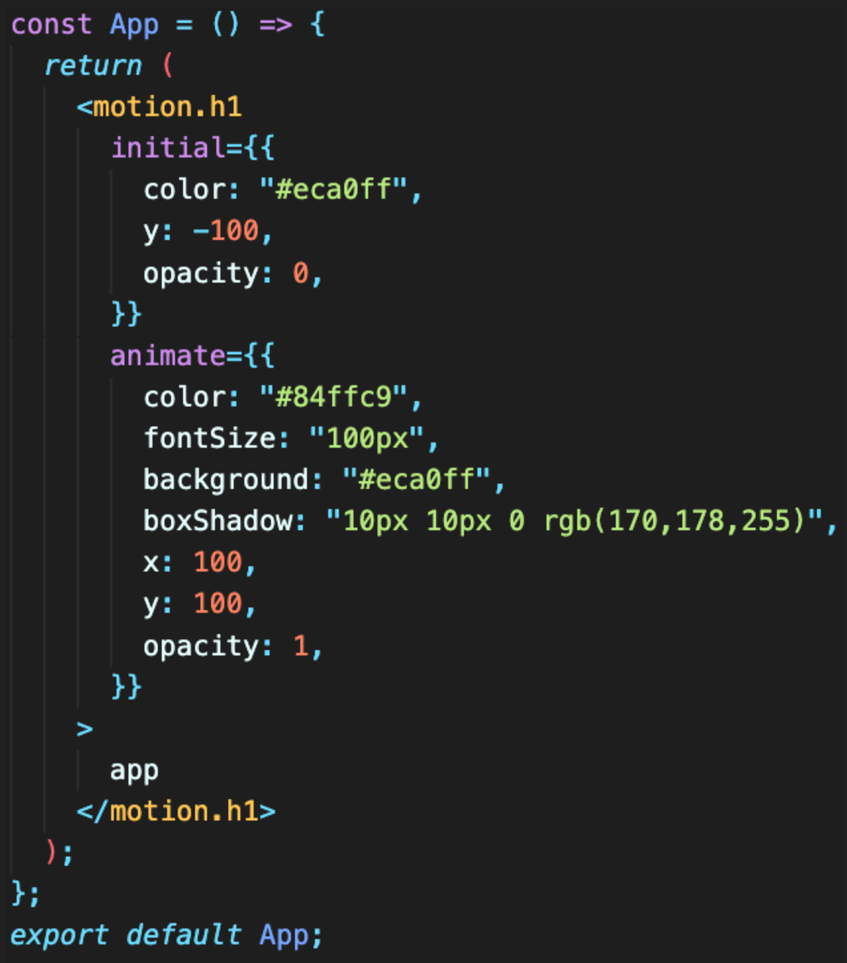
- There is also other cool stuff **x and y**
    - x and y are not css properties
    - x: translate left or right (positive right)
    - y: translate up or down (positive down)



```
const App = () => {
  return (
    <motion.h1
      animate={{
        color: "#84ffc9",
        fontSize: "100px",
        background: "#eca0ff",
        boxShadow: "10px 10px 0 rgb(170,178,255)",
        x: 100,
        y: 100,
      }}
    >
      app
    </motion.h1>
  );
};
export default App;
```

Framer - what do now?
Initial

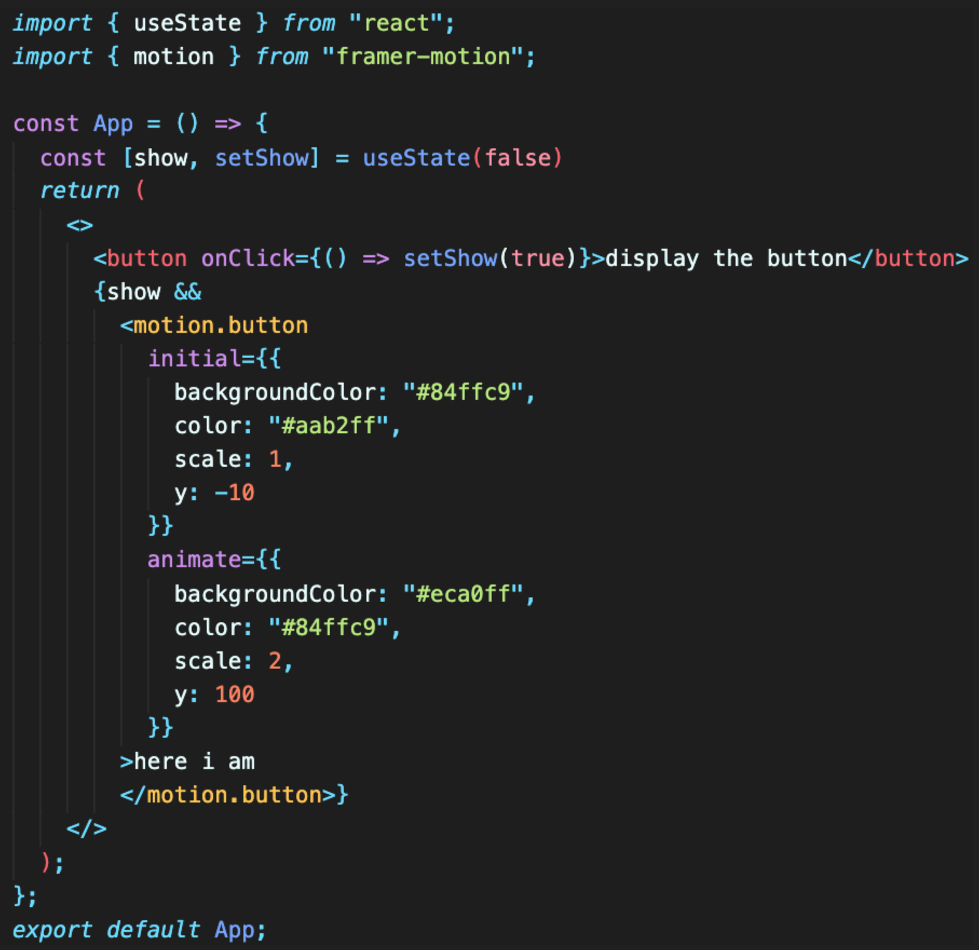
- Initial starting point of the properties we animate from
- Gives us a bit more control



```
const App = () => {
  return (
    <motion.h1
      initial={{
        color: "#eca0ff",
        y: -100,
        opacity: 0,
      }}
      animate={{
        color: "#84ffc9",
        fontSize: "100px",
        background: "#eca0ff",
        boxShadow: "10px 10px 0 rgb(170,178,255)",
        x: 100,
        y: 100,
        opacity: 1,
      }}
    >
      app
    </motion.h1>
  );
};
export default App;
```

Framer Initial – y tho?

- Think about with react what this could be used for
- When a state is true show this…

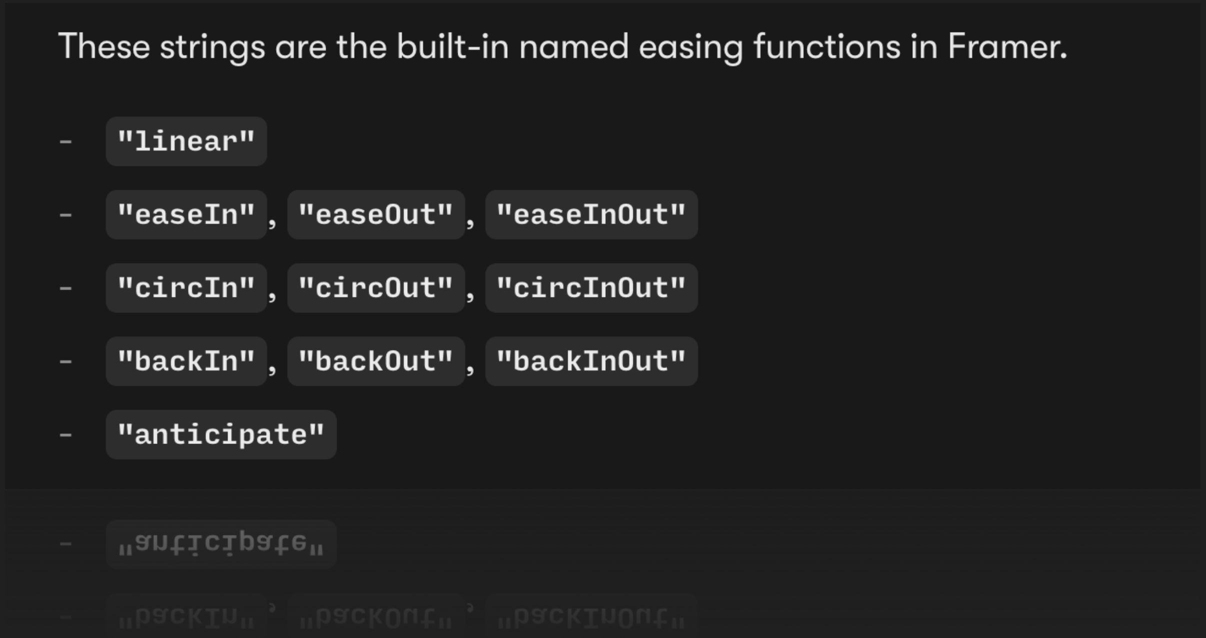


```
import { useState } from "react";
import { motion } from "framer-motion";

const App = () => {
  const [show, setShow] = useState(false)
  return (
    <>
      <button onClick={() => setShow(true)}>display the button</button>
      {show &&
        <motion.button
          initial={{
            backgroundColor: "#84ffc9",
            color: "#aab2ff",
            scale: 1,
            y: -10
          }}
          animate={{
            backgroundColor: "#eca0ff",
            color: "#84ffc9",
            scale: 2,
            y: 100
          }}
        >here i am
        </motion.button>}
    </>
  );
};
export default App;
```

Framer Transition

- How the animation transitions from start to end
- From **initial** to **animate**

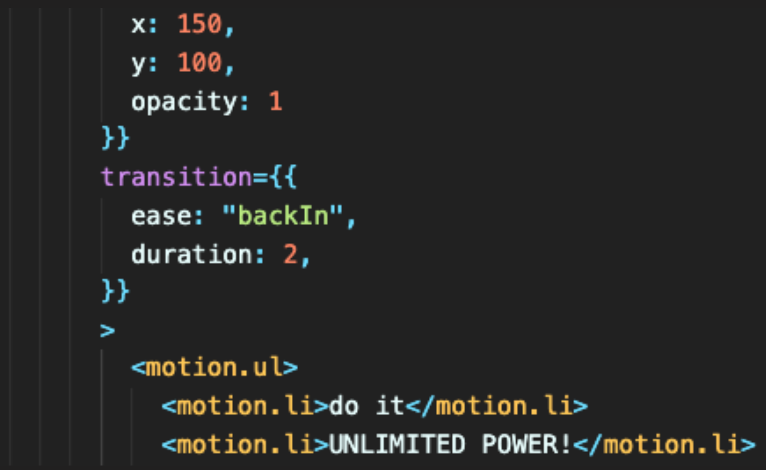


These strings are the built-in named easing functions in Framer.

- `"linear"`
- `"easeIn"`, `"easeOut"`, `"easeInOut"`
- `"circIn"`, `"circOut"`, `"circInOut"`
- `"backIn"`, `"backOut"`, `"backInOut"`
- `"anticipate"`

Framer Transition

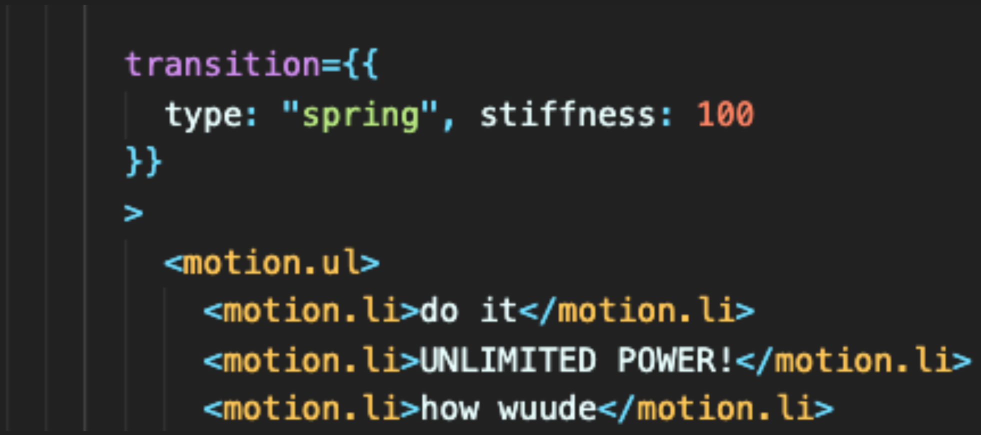
- Change the way the div enters
- Can use either one of the previous string values or have an array of numbers



```
      x: 150,
      y: 100,
      opacity: 1
    }}
    transition={{
      ease: "backIn",
      duration: 2,
    }}
    >
      <motion.ul>
        <motion.li>do it</motion.li>
        <motion.li>UNLIMITED POWER!</motion.li>
```

Framer Transition

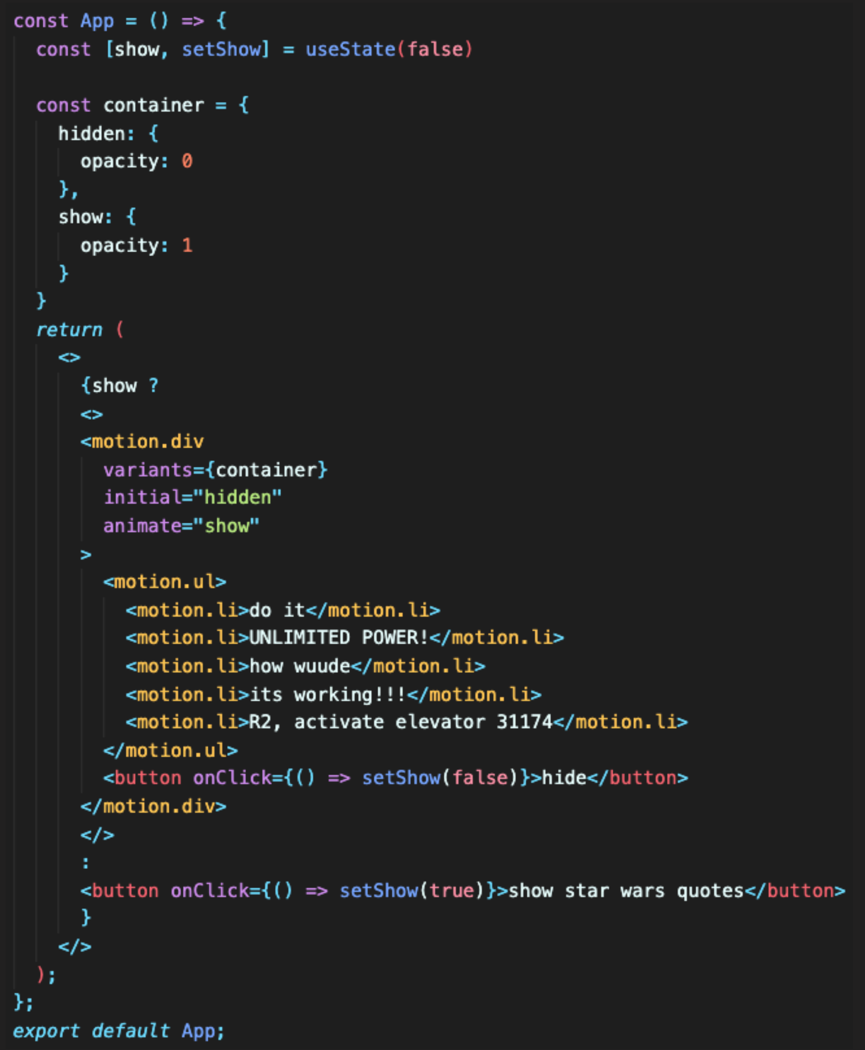
- We also have a types property
  - Tween
  - Spring
  - Inertia



```
transition={{
  type: "spring", stiffness: 100
}}
>
  <motion.ul>
    <motion.li>do it</motion.li>
    <motion.li>UNLIMITED POWER!</motion.li>
    <motion.li>how wuude</motion.li>
```

Framer Variants

- Can use this to group animation, initial and transition together
- Can help us keep our code looking clean

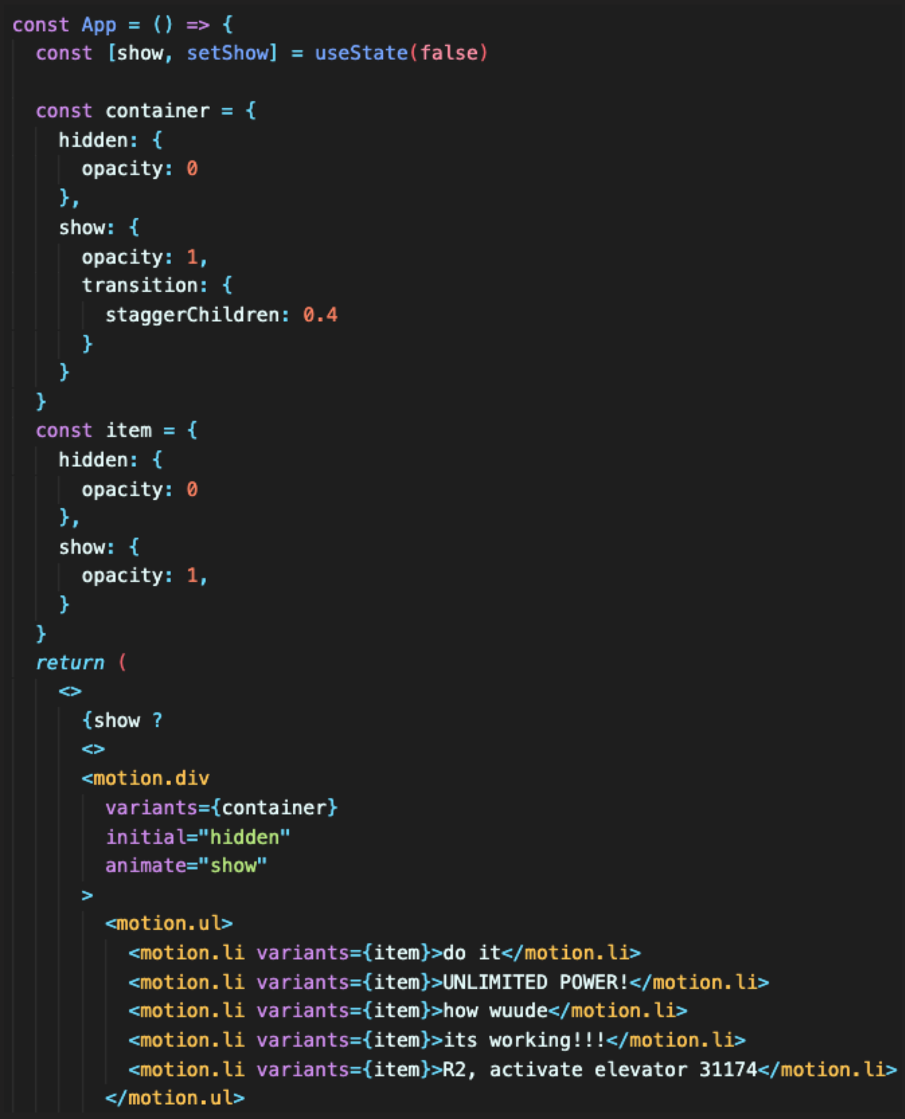


```jsx
const App = () => {
  const [show, setShow] = useState(false)

  const container = {
    hidden: {
      opacity: 0
    },
    show: {
      opacity: 1
    }
  }
  return (
    <>
      {show ?
      <>
      <motion.div
        variants={container}
        initial="hidden"
        animate="show"
      >
        <motion.ul>
          <motion.li>do it</motion.li>
          <motion.li>UNLIMITED POWER!</motion.li>
          <motion.li>how wuude</motion.li>
          <motion.li>its working!!!</motion.li>
          <motion.li>R2, activate elevator 31174</motion.li>
        </motion.ul>
        <button onClick={() => setShow(false)}>hide</button>
      </motion.div>
      </>
      :
      <button onClick={() => setShow(true)}>show star wars quotes</button>
      }
    </>
  );
};
export default App;
```

Framer Variants

- Can also bring in **orchestration**
- Meaning with variants, we can let the parent elements decide when the animation will execute for its children



```
const App = () => {
  const [show, setShow] = useState(false)

  const container = {
    hidden: {
      opacity: 0
    },
    show: {
      opacity: 1,
      transition: {
        staggerChildren: 0.4
      }
    }
  }
  const item = {
    hidden: {
      opacity: 0
    },
    show: {
      opacity: 1,
    }
  }
  return (
    <>
      {show ?
      <>
      <motion.div
        variants={container}
        initial="hidden"
        animate="show"
      >
        <motion.ul>
          <motion.li variants={item}>do it</motion.li>
          <motion.li variants={item}>UNLIMITED POWER!</motion.li>
          <motion.li variants={item}>how wuude</motion.li>
          <motion.li variants={item}>its working!!!</motion.li>
          <motion.li variants={item}>R2, activate elevator 31174</motion.li>
        </motion.ul>
```

FRAMER

Check out the docs for more cool /sweet stuff

https://www.framer.com/docs/

{CODENATION}