Motion Dev

ACM Spark

Why Motion Dev?

Why Motion Dev specifically?

- Very simple to get started
- Created using React hooks, allowing for optimal performance on React apps
- Can be used dynamically with React for more complex animations
    - Beyond the scope of this program

Setup

Setup

- `npm install motion`
- If you are the first person to integrate, then you need to install motion directly
- Otherwise, just run `npm i` to integrate it into your local packages

Setup

- `“use client”;`
  - Render elements on the client’s side
- `import { motion } from "motion/react"`
- Include this line at the top of your component to import the package
  - Similar how you import useState or useEffect at the top of each file

Application

animate

- Similar to adding `className` to apply Tailwind properties, add a `animate` element to the motion div
- Renders once when the page is initially loaded

```
<motion.div animate={{ rotate: 360}}>

      Some Text

</motion.div>
```

transition

- Take further control by adding durations and delays
- These are elements of the transition property

```
<motion.div animate={{ rotate: 360 }}

            transition={{ duration: 0.5, delay: 0.5 }}>

    Some Text

</motion.div>
```

initial

- Starts off with the state in `initial` applied
- Ends off with the state in `animate` applied

```
<motion.div initial={{scale: 0}} animate={{ scale: 1}}>

      Some Text

</motion.div>
```

whileInView

- `whileInView` replaces the `animate` property
- Waits till the element is visible on your browser to play the animation

```
<motion.div initial={{scale: 0}} whileInView={{ scale: 1}}>

      Some Text

</motion.div>
```

viewport

- Ignore subsequent enter/leave events

```
<motion.div initial={{scale: 0}} whileInView={{ scale: 1}}
            viewport={{ once: true}}>
      Some Text

</motion.div>
```

Putting it all Together

```
{values.map(({ name }, index) => (
    <motion.div
        key={index} initial={{ opacity: 0, y: -20 }}
        whileInView={{ opacity: 1, y: 0 }}
        transition={{ duration: 0.5,
        delay: index * 0.1 }} > // each animates sequentially
        {name}
    </motion.div>
))}
```

whileHover

- Useful to emphasize that something can be clicked or is interactive

```
<motion.div whileHover={{ scale: 1.05}}>

      Some Text

</motion.div>
```

Reuse Animations

- Declare animation above the component
- Notice that `transition` is a part of `whileInView`

```
const sampleAnimation = {

  initial: { scale: 0 },

  whileInView: { scale: 1, transition: { duration: 4.0 } }

};
```

Reuse Animations

```
<motion.div variants={sampleAnimation} initial="initial"
            whileInView="whileInView">
      Some Text
</motion.div>
```

Properties you can Animate

- Opacity
- X, Y, Z
- scaleX, scaleY, scaleZ
- rotateX, rotateY, rotateZ
- skewX, skewY, skewZ
- transformPerspective
- Many many more! Check out the motion docs:
    - https://motion.dev/docs/react-animation

Be Careful!

- It is very easy to apply lots of animations
- Make sure to always keep the average user experience in mind
    - Do not be overwhelm them

THANK YOU!

Questions, comments, concerns?