ReactJS - Animation

Animation is an exciting feature of modern web application. It gives a refreshing feel to the application. React community provides many excellent react based animation library like React Motion, React Reveal, react-animations, etc., React itself provides an animation library, *React Transition Group* as an add-on option earlier. It is an independent library enhancing the earlier version of the library. Let us learn React Transition Group animation library in this chapter.

React Transition Group

React Transition Group library is a simple implementation of animation. It does not do any animation out of the box. Instead, it exposes the core animation related information. Every animation is basically transition of an element from one state to another. The library exposes minimum possible state of every element and they are given below −

- Entering
- Entered
- Exiting
- Exited

The library provides options to set CSS style for each state and animate the element based on the style when the element moves from one state to another. The library provides in props to set the current state of the element. If *in* props value is true, then it means the element is moving from *entering* state to *exiting* state. If in props value is false, then it means the element is moving from exiting to exited.

Transition

Transition is the basic component provided by the *React Transition Group* to animate an element. Let us create a simple application and try to fade in / fade out an element using *Transition element*.

First, create a new react application, *react-animation-app* using *Create React App* or *Rollup* bundler by following instruction in *Creating a React application* chapter.

Next, install *React Transition Group* library.

```
cd /go/to/project
npm install react-transition-group --save
```

Next, open the application in your favorite editor.

Next, create src folder under the root directory of the application.

Next, create *components* folder under *src* folder.

Next, create a file, *HelloWorld.js* under *src/components* folder and start editing.

Next, import *React* and animation library.

```
import React from 'react';
import { Transition } from 'react-transition-group'
```

Next, create the *HelloWorld* component.

```
class HelloWorld extends React.Component {
   constructor(props) {
      super(props);
   }
}
```

Next, define transition related styles as JavaScript objects in the constructor.

```
this.duration = 2000;
this.defaultStyle = {
   transition: `opacity ${this.duration}ms ease-in-out`,
   opacity: 0,
}
this.transitionStyles = {
   entering: { opacity: 1 },
   entered: { opacity: 1 },
   exiting: { opacity: 0 },
   exited: { opacity: 0 },
};
```

Here,

defaultStyles sets the transition animation
transitionStyles set the styles for various states

Next, set the initial state for the element in the constructor.

```
this.state = {
   inProp: true
```

```
}
```

Next, simulate the animation by changing the *inProp* values every 3 seconds.

```
setInterval(() => {
   this.setState((state, props) => {
      let newState = {
         inProp: !state.inProp
      };
      return newState;
   })
}, 3000);
```

Next, create a *render* function.

```
render() {
   return (
   );
}
```

Next, add *Transition* component. Use *this.state.inProp* for *in* prop and *this.duration* for *timeout prop* . Transition component expects a function, which returns the user interface. It is basically a *Render props*.

```
render() {
   return (
      <Transition in={this.state.inProp} timeout={this.duration}>
         {state => ({
            ... component's user interface.
         })
      </Transition>
   );
}
```

Next, write the components user interface inside a container and set the *defaultStyle* and *transitionStyles* for the container.

```
render() {
   return (
      <Transition in={this.state.inProp} timeout={this.duration}>
         {state => (
            <div style={{
               ...this.defaultStyle,
               ...this.transitionStyles[state]
            }}>
```

```
               <h1>Hello World!</h1>
            </div>
         )}
      </Transition>
   );
}
```

Finally, expose the component.

```
export default HelloWorld
```

The complete source code of the component is as follows −

```
import React from "react";
import { Transition } from 'react-transition-group';

class HelloWorld extends React.Component {
   constructor(props) {
      super(props);
      this.duration = 2000;
      this.defaultStyle = {
         transition: `opacity ${this.duration}ms ease-in-out`,
         opacity: 0,
      }
      this.transitionStyles = {
         entering: { opacity: 1 },
         entered: { opacity: 1 },
         exiting: { opacity: 0 },
         exited: { opacity: 0 },
      };
      this.state = {
         inProp: true
      }
      setInterval(() => {
         this.setState((state, props) => {
            let newState = {
               inProp: !state.inProp
            };
            return newState;
         })
      }, 3000);
   }
   render() {
      return (
         <Transition in={this.state.inProp} timeout={this.duration}>
            {state => (
               <div style={{
                  ...this.defaultStyle,
                  ...this.transitionStyles[state]
               }}>
```

```
                  <h1>Hello World!</h1>
               </div>
            )}
         </Transition>
      );
   }
}
export default HelloWorld;
```

Next, create a file, *index.js* under the *src* folder and use *HelloWorld* component.

```
import React from 'react';
import ReactDOM from 'react-dom';
import HelloWorld from './components/HelloWorld';

ReactDOM.render(
   <React.StrictMode
      <HelloWorld /
   </React.StrictMode   ,
   document.getElementById('root')
);
```

Finally, create a *public* folder under the root folder and create *index.html* file.

```
<!DOCTYPE html>
<html lang="en">
   <head>
      <meta charset="utf-8">
      <title>React Containment App</title>
   </head>
   <body>
      <div id="root"></div>
      <script type="text/JavaScript" src="./index.js"></script>
   </body>
</html>
```

Next, serve the application using npm command.

```
npm start
```

Next, open the browser and enter *http://localhost:3000* in the address bar and press enter.

Clicking the remove link will remove the item from redux store.

Animation

Image not found or type unknown

CSSTransition

CSSTransition is built on top of *Transition* component and it improves *Transition* component by introducing *classNames* prop. *classNames* prop refers the css class name used for various state of the element.

For example, *classNames=hello* prop refers below css classes.

```
.hello-enter {
   opacity: 0;
}
.hello-enter-active {
   opacity: 1;
   transition: opacity 200ms;
}
.hello-exit {
   opacity: 1;
}
.hello-exit-active {
   opacity: 0;
   transition: opacity 200ms;
}
```

Let us create a new component *HelloWorldCSSTransition* using *CSSTransition* component.

First, open our *react-animation-app application* in your favorite editor.

Next, create a new file, *HelloWorldCSSTransition.css* under *src/components* folder and enter transition classes.

```
.hello-enter {
   opacity: 1;
   transition: opacity 2000ms ease-in-out;
}
.hello-enter-active {
   opacity: 1;
   transition: opacity 2000ms ease-in-out;
}
.hello-exit {
   opacity: 0;
   transition: opacity 2000ms ease-in-out;
}
.hello-exit-active {
   opacity: 0;
   transition: opacity 2000ms ease-in-out;
}
```

Next, create a new file, *HelloWorldCSSTransition.js* under *src/components* folder and start editing.

Next, import *React* and animation library.

```
import React from 'react';
import { CSSTransition } from 'react-transition-group'
```

Next, import *HelloWorldCSSTransition.css*.

```
import './HelloWorldCSSTransition.css'
```

Next, create the *HelloWorld* component.

```
class HelloWorldCSSTransition extends React.Component {
   constructor(props) {
      super(props);
   }
}
```

Next, define duration of the transition in the constructor.

```
this.duration = 2000;
```

Next, set the initial state for the element in the constructor.

```
this.state = {
   inProp: true
}
```

Next, simulate the animation by changing the inProp values every 3 seconds.

```
setInterval(() => {
   this.setState((state, props) => {
      let newState = {
         inProp: !state.inProp
      };
      return newState;
   })
}, 3000);
```

Next, create a *render* function.

```
render() {
   return (
   );
}
```

Next, add *CSSTransition* component. Use *this.state.inProp* for in prop, *this.duration* for *timeout* prop and *hello* for *classNames* prop. *CSSTransition* component expects user interface as child prop.

```
render() {
   return (
      <CSSTransition in={this.state.inProp} timeout={this.duration}
         classNames="hello">
         // ... user interface code ...
      </CSSTransition>
   );
}
```

Next, write the components user interface.

```
render() {
   return (
       <CSSTransition in={this.state.inProp} timeout={this.duration}
      classNames="hello">
      <div>
          <h1>Hello World!</h1>
      </div>
       </CSSTransition>
   );
}
```

Finally, expose the component.

```
export default HelloWorldCSSTransition;
```

The complete source code of the component is given below −

```
import React from 'react';
import { CSSTransition } from 'react-transition-group'
import './HelloWorldCSSTransition.css'

class HelloWorldCSSTransition extends React.Component {
   constructor(props) {
      super(props);
      this.duration = 2000;
      this.state = {
```

```
            inProp: true
        }
        setInterval(() => {
            this.setState((state, props) => {
                let newState = {
                    inProp: !state.inProp
                };
                return newState;
            })
        }, 3000);
    }
    render() {
        return (
            <CSSTransition in={this.state.inProp} timeout={this.duration}
                classNames="hello">
                <div>
                    <h1>Hello World!</h1>
                </div>
            </CSSTransition>
        );
    }
}
export default HelloWorldCSSTransition;
```

Next, create a file, *index.js* under the *src* folder and use *HelloWorld* component.

```
import React from 'react';
import ReactDOM from 'react-dom';
import HelloWorldCSSTransition from './components/HelloWorldCSSTransition';

ReactDOM.render(
    <React.StrictMode>
        <HelloWorldCSSTransition />
    </React.StrictMode>,
    document.getElementById('root')
);
```

Next, serve the application using npm command.

```
npm start
```

Next, open the browser and enter *http://localhost:3000* in the address bar and press enter.

The message will fade in and out for every 3 seconds.

Animation

Image not found or type unknown

TransitionGroup

TransitionGroup is a container component, which manages multiple transition component in a list. For example, while each item in a list use *CSSTransition, TransitionGroup* can be used to group all the item for proper animation.

```
<TransitionGroup>
   {items.map(({ id, text }) => (
      <CSSTransition key={id} timeout={500} classNames="item" >
         <Button
            onClick={() =>
               setItems(items =>
                  items.filter(item => item.id !== id)
               )
            }
            >
            &times;
         </Button>
         {text}
      </CSSTransition>
   ))}
</TransitionGroup>
```

Revision #1
Created 14 December 2022 10:47:39 by Admin
Updated 14 December 2022 10:48:23 by Admin