**安装包**

npm install react-motion --save

**简单示例**

import React from 'react'

import IComponent from 'BaseCMSManage/Components/IETemplateComponents/IEAnimation/IComponent'

import { Motion, spring, presets } from 'react-motion'

class IEAnimation extends IComponent {

    constructor(props) {

        super(props);

        this.state = {

            value: 0

        }

}

    componentDidMount() {

        this.setState({ value: 100 });

}

    render() {

        // presets 是预设的几种动画类型

        // 动画执行过程

        // 初始化时 x = 0，

        // 当我们 setState 时，x = 100

        // 这时将产生一个 x 从 0 到 100 的一个变化，如（0, 20, 40, 60, 80, 100）,如何变化取决于我们的预设 presets

        // 每次变化将会调用我们的 动画组件 生成组件

        return (<div>

            <Motion style={{ x: spring(this.state.value, presets.wobbly) }}>

                {/\* 动画组件 \*/}

                {interpolatingStyle => {

                    return (

                        <div style={{ left: `${interpolatingStyle.x - 100}%`, position: 'relative' }}>

                            {

                                this.props.children

                            }

                        </div>

                    )

                }}

            </Motion>

        </div>);

    }

}

**联动动画StaggeredMotion组件**

联动动画执行过程n次变化如下：

1. 组件1[x=10]，组件2[x=0]，组件3[x=0]
2. 组件1[x=20]，组件2[x=10], 组件3[x=0]
3. 组件1[x=30]，组件2[x=20]，组件3[x=10]

示例：

import React from 'react'

import IComponent from 'BaseCMSManage/Components/IETemplateComponents/IEAnimation/IComponent'

import { Motion, spring, presets, StaggeredMotion } from 'react-motion'

class IEAnimation extends IComponent {

    constructor(props) {

        super(props);

        this.state = {

            value: 0

        }

}

    componentDidMount() {

        this.setState({ value: 100 });

}

    render() {

        let childrens = [

            {childVal: this.state.value},

            {childVal: this.state.value},

            {childVal: this.state.value}

        ]

        return (<div>

            <StaggeredMotion

                defaultStyles={childrens}

                styles={prevInterpolatedStyles => {

                    // 这一步很重要，prevInterpolatedStyles 是上一次变化的 styles

                    // 而我们要返回的是这一次变化的 styles

                    // 我们这一次变化的 styles，要基于上一次变化

                    // 也就是说这一次的 styles[1] 要等于上一次的 prevInterpolatedStyles[0]，同理 2->1，3->2 ...

                    return prevInterpolatedStyles.map((\_, i) => {

                        return i === 0

                        ? {childVal: spring(this.state.value)}

                        : {childVal: spring(prevInterpolatedStyles[i - 1].childVal, presets.wobbly)}

                    })

                }}

            >

                {interpolatingStyles => {

                    return <>

                        {

                            interpolatingStyles.map((e, i)=>(

                                <div style={{ left: `${e.childVal - 100}%`, position: 'relative' }}></div>

                            ))

                        }

                    </>

                }}

            </StaggeredMotion>

        </div>);

    }

}

**挂载卸载动画TransitionMotion**

没试过，直接使用官方示例：

import createReactClass from 'create-react-class';

const Demo = createReactClass({

  getInitialState() {

    return {

      items: [{key: 'a', size: 10}, {key: 'b', size: 20}, {key: 'c', size: 30}],

    };

  },

  componentDidMount() {

    this.setState({

      items: [{key: 'a', size: 10}, {key: 'b', size: 20}], // remove c.

    });

  },

  willLeave() {

    // triggered when c's gone. Keeping c until its width/height reach 0.

    return {width: spring(0), height: spring(0)};

  },

  render() {

    return (

      <TransitionMotion

        willLeave={this.willLeave}

        styles={this.state.items.map(item => ({

          key: item.key,

          style: {width: item.size, height: item.size},

        }))}>

        {interpolatedStyles =>

          // first render: a, b, c. Second: still a, b, c! Only last one's a, b.

          <div>

            {interpolatedStyles.map(config => {

              return <div key={config.key} style={{...config.style, border: '1px solid'}} />

            })}

          </div>

        }

      </TransitionMotion>

    );

  },

});