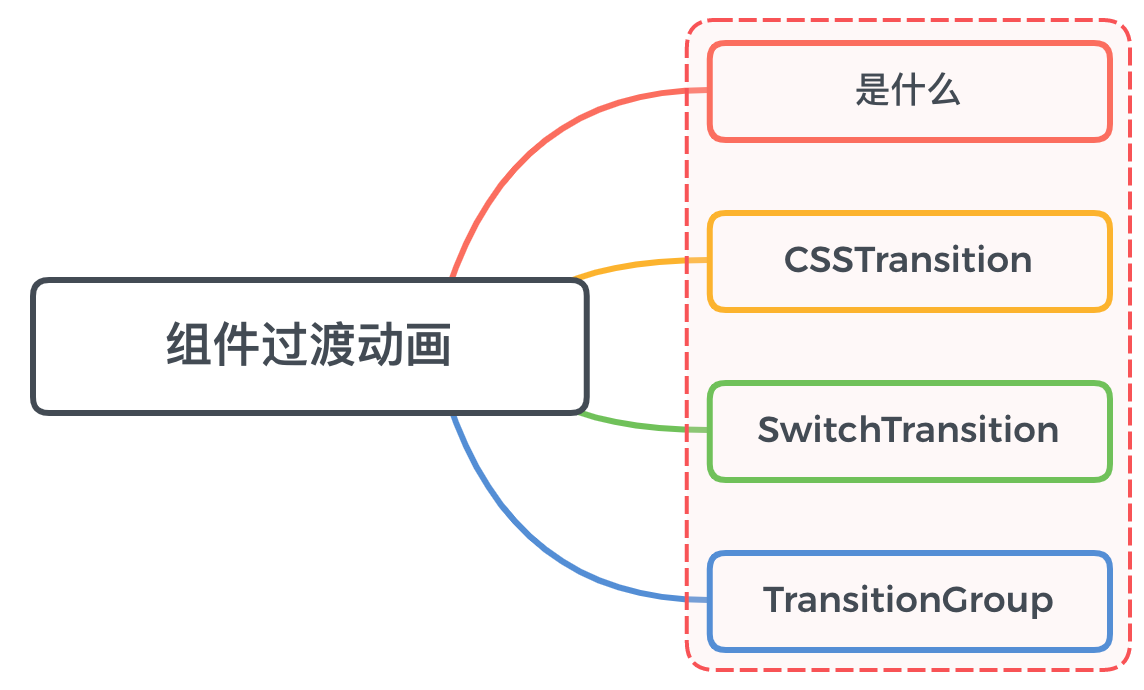
# 面试官：在react中组件间过渡动画如何实现？



## 一、是什么

在日常开发中，页面切换时的转场动画是比较基础的一个场景

当一个组件在显示与消失过程中存在过渡动画，可以很好的增加用户的体验

在react中实现过渡动画效果会有很多种选择，如react-transition-group，react-motion，Animated，以及原生的CSS都能完成切换动画

## 二、如何实现

在react中，react-transition-group是一种很好的解决方案，其为元素添加enter，enter-active，exit，exit-active这一系列勾子

可以帮助我们方便的实现组件的入场和离场动画

其主要提供了三个主要的组件：

* CSSTransition：在前端开发中，结合 CSS 来完成过渡动画效果
* SwitchTransition：两个组件显示和隐藏切换时，使用该组件
* TransitionGroup：将多个动画组件包裹在其中，一般用于列表中元素的动画

### CSSTransition

其实现动画的原理在于，当CSSTransition的in属性置为true时，CSSTransition首先会给其子组件加上xxx-enter、xxx-enter-active的class执行动画

当动画执行结束后，会移除两个class，并且添加-enter-done的class

所以可以利用这一点，通过css的transition属性，让元素在两个状态之间平滑过渡，从而得到相应的动画效果

当in属性置为false时，CSSTransition会给子组件加上xxx-exit和xxx-exit-active的class，然后开始执行动画，当动画结束后，移除两个class，然后添加-enter-done的class

如下例子：

export default class App2 extends React.PureComponent {  
  
 state = {show: true};  
  
 onToggle = () => this.setState({show: !this.state.show});  
  
 render() {  
 const {show} = this.state;  
 return (  
 <div className={'container'}>  
 <div className={'square-wrapper'}>  
 <CSSTransition  
 in={show}  
 timeout={500}  
 classNames={'fade'}  
 unmountOnExit={true}  
 >  
 <div className={'square'} />  
 </CSSTransition>  
 </div>  
 <Button onClick={this.onToggle}>toggle</Button>  
 </div>  
 );  
 }  
}

对应css样式如下：

.fade-enter {  
 opacity: 0;  
 transform: translateX(100%);  
}  
  
.fade-enter-active {  
 opacity: 1;  
 transform: translateX(0);  
 transition: all 500ms;  
}  
  
.fade-exit {  
 opacity: 1;  
 transform: translateX(0);  
}  
  
.fade-exit-active {  
 opacity: 0;  
 transform: translateX(-100%);  
 transition: all 500ms;  
}

### SwitchTransition

SwitchTransition可以完成两个组件之间切换的炫酷动画

比如有一个按钮需要在on和off之间切换，我们希望看到on先从左侧退出，off再从右侧进入

SwitchTransition中主要有一个属性mode，对应两个值：

* in-out：表示新组件先进入，旧组件再移除；
* out-in：表示就组件先移除，新组建再进入

SwitchTransition组件里面要有CSSTransition，不能直接包裹你想要切换的组件

里面的CSSTransition组件不再像以前那样接受in属性来判断元素是何种状态，取而代之的是key属性

下面给出一个按钮入场和出场的示例，如下：

import { SwitchTransition, CSSTransition } from "react-transition-group";  
  
export default class SwitchAnimation extends PureComponent {  
 constructor(props) {  
 super(props);  
  
 this.state = {  
 isOn: true  
 }  
 }  
  
 render() {  
 const {isOn} = this.state;  
  
 return (  
 <SwitchTransition mode="out-in">  
 <CSSTransition classNames="btn"  
 timeout={500}  
 key={isOn ? "on" : "off"}>  
 {  
 <button onClick={this.btnClick.bind(this)}>  
 {isOn ? "on": "off"}  
 </button>  
 }  
 </CSSTransition>  
 </SwitchTransition>  
 )  
 }  
  
 btnClick() {  
 this.setState({isOn: !this.state.isOn})  
 }  
}

css文件对应如下：

.btn-enter {  
 transform: translate(100%, 0);  
 opacity: 0;  
}  
  
.btn-enter-active {  
 transform: translate(0, 0);  
 opacity: 1;  
 transition: all 500ms;  
}  
  
.btn-exit {  
 transform: translate(0, 0);  
 opacity: 1;  
}  
  
.btn-exit-active {  
 transform: translate(-100%, 0);  
 opacity: 0;  
 transition: all 500ms;  
}

### TransitionGroup

当有一组动画的时候，就可将这些CSSTransition放入到一个TransitionGroup中来完成动画

同样CSSTransition里面没有in属性，用到了key属性

TransitionGroup在感知children发生变化的时候，先保存移除的节点，当动画结束后才真正移除

其处理方式如下：

* 插入的节点，先渲染dom，然后再做动画
* 删除的节点，先做动画，然后再删除dom

如下：

import React, { PureComponent } from 'react'  
import { CSSTransition, TransitionGroup } from 'react-transition-group';  
  
export default class GroupAnimation extends PureComponent {  
 constructor(props) {  
 super(props);  
  
 this.state = {  
 friends: []  
 }  
 }  
  
 render() {  
 return (  
 <div>  
 <TransitionGroup>  
 {  
 this.state.friends.map((item, index) => {  
 return (  
 <CSSTransition classNames="friend" timeout={300} key={index}>  
 <div>{item}</div>  
 </CSSTransition>  
 )  
 })  
 }  
 </TransitionGroup>  
 <button onClick={e => this.addFriend()}>+friend</button>  
 </div>  
 )  
 }  
  
 addFriend() {  
 this.setState({  
 friends: [...this.state.friends, "coderwhy"]  
 })  
 }  
}

对应css如下：

.friend-enter {  
 transform: translate(100%, 0);  
 opacity: 0;  
}  
  
.friend-enter-active {  
 transform: translate(0, 0);  
 opacity: 1;  
 transition: all 500ms;  
}  
  
.friend-exit {  
 transform: translate(0, 0);  
 opacity: 1;  
}  
  
.friend-exit-active {  
 transform: translate(-100%, 0);  
 opacity: 0;  
 transition: all 500ms;  
}

## 参考文献

* https://segmentfault.com/a/1190000018861018
* https://mp.weixin.qq.com/s/14HneI7SpfrRHKtqgosIiA