

react中TS类组件的写法

原创

ICPunk

于 2022-06-25 17:34:39 发布

1084

收藏

版权

分类专栏:

[react](#)

文章标签:

[react.js](#)

[javascript](#)

[前端](#)



react

专栏收录该内容

0 订阅

2 篇文章

订阅专

栏

类组件最多的就是Typescript接口类型的应用，限制组件状态或者属性符合接口定义

1. state.tsx

```
1 state = { name: 'xx' }
2 setState({
3   name: 100 // 这个值如果是后台返回的，你不知道类型，运行时才能看到报错；或者调用别人的
4 })
5 // 换种写法再试试
6 interface IState {
7   name: string
8 }
9 export ...
10 state: IState = { name: 'xx' }
11 //
12 setState({ // 此时这样改并不会提示，因为不是直接改，this.state.name = 100这样才有提示
13   name: 100
14 })
15 // react有专门的写法
16 export default class App extends Component<约定属性, 约定状态>
17 export default class App extends Component<any, IState> { // 属性不约束，状态约束
18   state = {
19     name: 'xxx' // 这时就会提示类型校验
20   }
21   render() {
22     return (
23       // ...
24       setState({name: "yyy"})
25       // ...
26     )
27   }
28 }
```

2. todo.tsx

```
1 interface IState {
2   text: string,
3   list: string[]
4 }
5 export default class App extends Component<any, IState> {
6   state = {
7     text: "",
8     list: []
9   }
10   myRef = React.createRef<HTMLInputElement>() // 需要指定ref的类型
11   render() {
12     return (
13       <div>
14         // <input type="text" value={this.state.text} onChange={
15         //   (evt) => {
16         //     setState({
17         //       text: evt.target.value
18         //     })
19         //   }
20         // }>
21         // 或者试试ref
22         <input ref={this.myRef}>
23         <button onClick={
24           // console.log(state.name)
25           // console.log(this.myRef.current.value)
26           console.log((this.myRef.current as HTMLInputElement).value)
27           this.setState({
28             list: [...this.sate.list, (this.myRef.current as HTMLInp
29             )]}
30         )></button>
31       {
32         this.state.list.map(item=>
```

```

33         <li key={item}>{item}</li>
34     )
35 }
36 </div>
37 )
38 }
39 }

```

3. props.tsx

```

1  export default class App extends Component {
2      render() {
3          return (
4              <div>
5                  <Child name="aaa">
6                      </div>
7              )
8          }
9      }
10 interface IProps {
11     name: string
12 }
13 class Child extends Component<IProps, any> {
14     render() {
15         return <div>
16             {this.props.name}
17         </div>
18     }
19 }
20

```

4. drawer.tsx

```

1  export default class App extends Component {
2      state = {
3          isShow: true,
4      };
5      render() {
6          return (
7              <div>
8                  app
9                  <Navbar
10                     title="first page"
11                     cb={() => {
12                         this.setState({
13                             isShow: !this.state.isShow,
14                         });
15                     }}
16                 />
17                 {this.state.isShow && <Sidebar></Sidebar>}
18             </div>
19         );
20     }
21 }
22 interface IProps {
23     title: string;
24     cb: () => void;
25 }
26 class Navbar extends Component<IProps, any> {
27     render() {
28         return (
29             <div>
30                 Navbar-{this.props.title}
31                 <button
32                     onClick={() => {
33                         this.props.cb();
34                     }}
35                 ></button>
36             </div>
37         );
38     }
39 }
40 class Sidebar extends Component {
41     render() {
42         return <div>Sidebar</div>;
43     }
44 }

```

