**React环境搭建**

1. 安装nodejs
2. 安装npm
3. 安装create-react-app

npm install –g create-react-app

1. 创建一个helloworld项目

create-react-app helloworld

E:\react>create-react-app helloworld

Creating a new React app in E:\react\helloworld.

Installing packages. This might take a couple of minutes.

Installing react, react-dom, and react-scripts...

yarn add v1.16.0

[1/4] Resolving packages...

warning react-scripts > fsevents@2.0.6: Please update: there are crash fixes

[2/4] Fetching packages...

info fsevents@1.2.9: The platform "win32" is incompatible with this module.

info "fsevents@1.2.9" is an optional dependency and failed compatibility check. Excluding it from installation.

info fsevents@2.0.6: The platform "win32" is incompatible with this module.

info "fsevents@2.0.6" is an optional dependency and failed compatibility check. Excluding it from installation.

[3/4] Linking dependencies...

warning "react-scripts > @typescript-eslint/eslint-plugin@1.6.0" has unmet peer dependency "typescript@\*".

warning "react-scripts > @typescript-eslint/parser@1.6.0" has unmet peer dependency "typescript@\*".

warning "react-scripts > @typescript-eslint/eslint-plugin > @typescript-eslint/typescript-estree@1.6.0" has unmet peer d

ependency "typescript@\*".

warning "react-scripts > @typescript-eslint/eslint-plugin > tsutils@3.10.0" has unmet peer dependency "typescript@>=2.8.

0 || >= 3.2.0-dev || >= 3.3.0-dev || >= 3.4.0-dev".

[4/4] Building fresh packages...

success Saved lockfile.

success Saved 11 new dependencies.

info Direct dependencies

├─ react-dom@16.8.6

├─ react-scripts@3.0.1

└─ react@16.8.6

info All dependencies

├─ babel-preset-react-app@9.0.0

├─ eslint-config-react-app@4.0.1

├─ fork-ts-checker-webpack-plugin@1.1.1

├─ microevent.ts@0.1.1

├─ react-app-polyfill@1.0.1

├─ react-dev-utils@9.0.1

├─ react-dom@16.8.6

├─ react-error-overlay@5.1.6

├─ react-scripts@3.0.1

├─ react@16.8.6

└─ worker-rpc@0.1.1

Done in 194.45s.

Success! Created helloworld at E:\react\helloworld

Inside that directory, you can run several commands:

yarn start

Starts the development server.

yarn build

Bundles the app into static files for production.

yarn test

Starts the test runner.

yarn eject

Removes this tool and copies build dependencies, configuration files

and scripts into the app directory. If you do this, you can’t go back!

We suggest that you begin by typing:

cd helloworld

yarn start

Happy hacking!

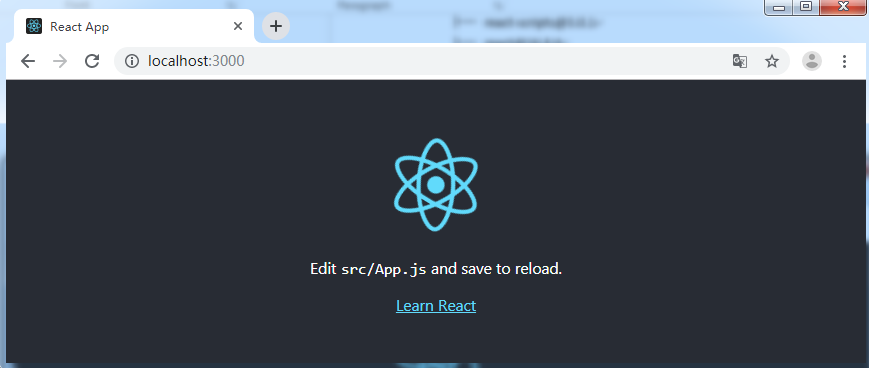
1. 启动项目

cd helloworld

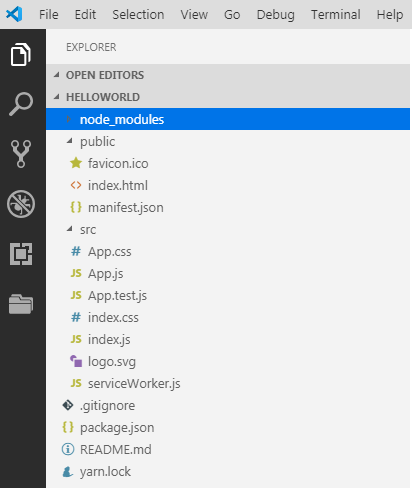
yarn start

或者npm run start

浏览器默认打开如下页面



**工程目录简介**



node\_modules: node依赖第三方包文件

public: 项目公用文件夹

favicon.ico: 网页标签的图标

index.html: 项目首页html模板

manifest.json: 移动app配置文件

src: 项目源码文件夹

App.css: 模块样式文件

App.js: 模块文件

App.test.js: 模块测试文件

index.css: 入口样式文件

index.js: 项目的入口文件

logo.svg: 首页logo图片文件

serviceWorker.js：手机app缓存文件

.gitignore： git忽略修改的文件。

package.json: 脚手架项目的配置文件。

README.md: 项目说明文件

yarn.lock: 项目依赖安装包版本号等信息，一般不修改此文件。

**react组件**

1、通过继承React.Component来创建一个组件

import React, { Component } from ‘react’;

//如果没有import Component，用如下形式

//import React from ‘react’;

//class App extends React.Component

class App extends Component {

render() {

return (

<div>

Hello world

</div>

);

}

}

export default App;

2、通过function的方式定义一个无状态组件，提供performance

import React from ‘react’;

function App() {

return (

<div>

Hello world

</div>

);

}

export default App;

3、在index.js中使用组件

import React from ‘react’;

import ReactDOM from ‘react-dom‘;

import App from ‘./App‘;

ReactDOM.render(<App />, document.getElementById(‘root’));

**TodoList组件的实现**

TodoList.js源码

import React, { Component, Fragment } from 'react';

class TodoList extends Component {

constructor(props) {

super(props);

//state用来保存组件数据

this.state = {

inputValue: '',

list: []

}

}

render() {

return (

<Fragment>

<div>

{/\*必须bind(this)\*/}

<input

value={this.state.inputValue}

onChange={this.handleInputChange.bind(this)}

/>

<button onClick={this.handleBtnClick.bind(this)}>提交</button>

</div>

<ul>

{

this.state.list.map((item, index) => {

return (

<li

key={index}

onClick={this.handleItemDelete.bind(this, index)}

>

{item}

</li>

)

})

}

</ul>

</Fragment>

);

}

handleInputChange(e) {

this.setState({

inputValue: e.target.value

});

}

handleBtnClick() {

this.setState({

list: [...this.state.list, this.state.inputValue],

inputValue: ''

});

}

handleItemDelete(index) {

//不能直接修改this.state.list

//this.state.list.splice(index, 1)

const list = [...this.state.list];

list.splice(index, 1);

this.setState({

list: list

});

}

}

export default TodoList;

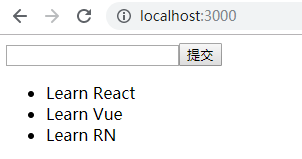
index.js源码

import React from 'react';

import ReactDOM from 'react-dom';

import TodoList from './TodoList';

ReactDOM.render(<TodoList />, document.getElementById('root'));



**JSX语法补充：**

1. 注释用{/\* \*/}
2. 注释或者用

{

//注释

}

1. JSX里样式的class用className
2. 如何需要转义list中的item（<h1>Hello World</h1>）可以在li中添加如下属性，并删除{item}

dangerouslySetInnerHTML={{\_\_html:item}}

1. lable的for属性，当点击label时，光标focus到对应的input

<label htmlFor=”insertArea”>输入内容</label>

<input id=”insertArea”/>

**TodoList组件的拆分和传值：**

TodoList.js源码：

import React, { Component, Fragment } from 'react';

import TodoItem from './TodoItem.js'

class TodoList extends Component {

constructor(props) {

super(props);

this.state = {

inputValue: '',

list: []

}

}

render() {

return (

<Fragment>

<div>

<label htmlFor="inputArea">输入内容：</label>

<input

id="inputArea"

value={this.state.inputValue}

onChange={this.handleInputChange.bind(this)}

/>

<button onClick={this.handleBtnClick.bind(this)}>提交</button>

</div>

<ul>

{

this.state.list.map((item, index) => {

return (

<div>

<TodoItem

content={item}

index={index}

deleteItem={this.handleItemDelete.bind(this)}

/>

</div>

)

})

}

</ul>

</Fragment>

);

}

handleInputChange(e) {

this.setState({

inputValue: e.target.value

});

}

handleBtnClick() {

this.setState({

list: [...this.state.list, this.state.inputValue],

inputValue: ''

});

}

handleItemDelete(index) {

const list = [...this.state.list];

list.splice(index, 1);

this.setState({

list: list

});

}

}

export default TodoList;

TODOItem.js源码：

import React, { Component } from 'react';

class TodoItem extends Component {

constructor(props) {

super(props);

this.handleClick = this.handleClick.bind(this);

}

render() {

return (

<div onClick={this.handleClick}>

{this.props.content}

</div>

)

}

handleClick() {

this.props.deleteItem(this.props.index);

}

}

export default TodoItem;

**上述代码优化**

TodoList.js源码：

import React, { Component, Fragment } from 'react';

import TodoItem from './TodoItem.js'

class TodoList extends Component {

constructor(props) {

super(props);

this.state = {

inputValue: '',

list: []

};

this.handleInputChange = this.handleInputChange.bind(this);

this.handleBtnClick = this.handleBtnClick.bind(this);

this.handleItemDelete = this.handleItemDelete.bind(this);

}

render() {

return (

<Fragment>

<div>

<label htmlFor="inputArea">输入内容：</label>

<input

id="inputArea"

value={this.state.inputValue}

onChange={this.handleInputChange}

/>

<button onClick={this.handleBtnClick}>提交</button>

</div>

<ul>

{ this.getTodoItem() }

</ul>

</Fragment>

);

}

getTodoItem() {

return this.state.list.map((item, index) => {

return (

<TodoItem

key={index}

content={item}

index={index}

deleteItem={this.handleItemDelete}

/>

)

});

}

handleInputChange(e) {

const value = e.target.value;

this.setState(() => ({

inputValue: value

}));

}

handleBtnClick() {

this.setState((prevState) => ({

list: [...prevState.list, prevState.inputValue],

inputValue: ''

}));

}

handleItemDelete(index) {

this.setState((prevState) => {

const list = [...prevState.list];

list.splice(index, 1);

return {list};

});

}

}

export default TodoList;

TODOItem.js源码：

import React, { Component } from 'react';

class TodoItem extends Component {

constructor(props) {

super(props);

this.handleClick = this.handleClick.bind(this);

}

render() {

const { content } = this.props;

return (

<div onClick={this.handleClick}>

{content}

</div>

)

}

handleClick() {

const { deleteItem, index } = this.props;

deleteItem(index);

}

}

export default TodoItem;

**React设计理念**

1. 声明式开发，通过改变数据来改变UI，而不是像jQuery直接操纵dom
2. 可以与其他框架并存
3. 组件化
4. 单向数据流
5. 视图层框架
6. 函数式编程

**PropTypes与DefaultProps的应用**

详细使用方法可以查看官方文档。

import React, { Component } from 'react';

import PropTypes from 'prop-types';

class TodoItem extends Component {

constructor(props) {

super(props);

this.handleClick = this.handleClick.bind(this);

}

render() {

const { content, test } = this.props;

return (

<div onClick={this.handleClick}>

{test} - {content}

</div>

)

}

handleClick() {

const { deleteItem, index } = this.props;

deleteItem(index);

}

}

TodoItem.propTypes = {

test: PropTypes.string.isRequired,

content: PropTypes.string,

deleteItem: PropTypes.func,

index: PropTypes.number

}

TodoItem.defaultProps = {

test: 'hello world'

}

export default TodoItem;

**虚拟DOM**

1. state数据
2. JSX模板
3. 数据 + 模板结合，生成真实的DOM来显示
4. state发生改变
5. 数据 + 模板结合，生成真实的DOM，替换原始的DOM

缺陷：

第一次生成了一个完整的DOM片段

第二次生成了一个完整的DOM片段

第二次的DOM替换第一次的DOM，非常耗性能

1. state数据
2. JSX模板
3. 数据 + 模板结合，生成真实的DOM来显示
4. state发生改变
5. 数据 + 模板结合，生成真实的DOM，并不直接替换原始的DOM
6. 新的DOM和原始的DOM做比较，找差异
7. 找出发生变化的element
8. 只用新的DOM中发生变化的element，替换掉老的DOM中相应的element元素

缺陷：

性能提升并不明显

1. state数据
2. JSX模板
3. 数据 + 模板结合，生成真实的DOM来显示

<div id=’abc’><span>hello world</span></div>

1. 生成虚拟DOM（虚拟DOM就是一个JS对象，用它来描述真实DOM）（损耗了性能）

[‘div’, {id: ‘abc’}, [‘span’, {}, ‘hello world’]]

1. state发生变化
2. 数据 + 模板生成新的虚拟DOM（极大的提升了性能）

[‘div’, {id: ‘abc’}, [‘span’, {}, ‘bye bye’]]

1. 比较原始虚拟DOM和新的虚拟DOM的区别，找到区别是span中的内容
2. 直接操作DOM，改变span中的内容

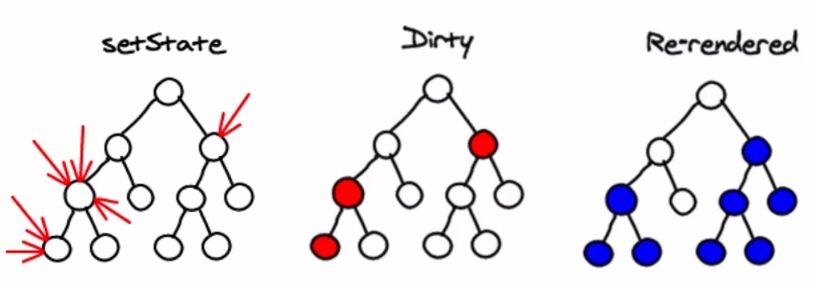
优点：

1. 性能提升了。
2. 它使得跨端应用得以实现。

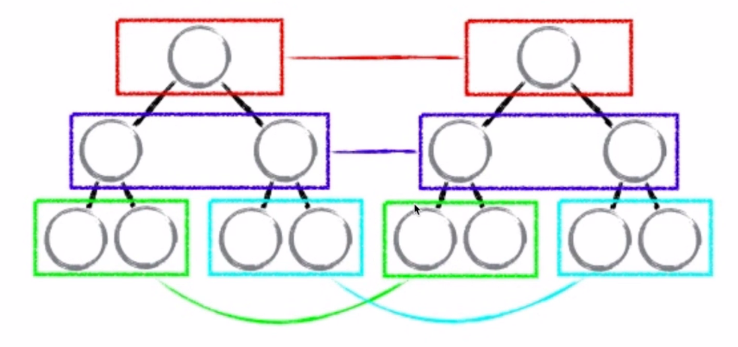
<div>item<div>等价下述写法

React.createElement(‘div’, {}, ‘item’);

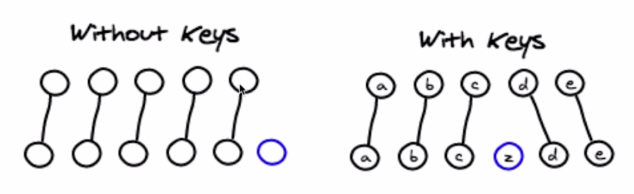
**虚拟DOM中的Diff算法**



同层比对，当第一层不同时，其子节点不继续做比对，直接全部替换。



有keys时，提高了比对性能。一般不用数组的index作为key值，因为index有可能发生变化，不利于作比对。Key值推荐选用稳定不变的值。



**React中ref的使用**

一般不推荐使用ref，尤为不要用ref来操作DOM，因为setState是异步函数，需要把相应操作写在setState里。

import React, { Component, Fragment } from 'react';

import TodoItem from './TodoItem.js'

class TodoList extends Component {

constructor(props) {

super(props);

this.state = {

inputValue: '',

list: []

};

this.handleInputChange = this.handleInputChange.bind(this);

this.handleBtnClick = this.handleBtnClick.bind(this);

this.handleItemDelete = this.handleItemDelete.bind(this);

}

render() {

return (

<Fragment>

<div>

<label htmlFor="inputArea">输入内容：</label>

<input

id="inputArea"

value={this.state.inputValue}

onChange={this.handleInputChange}

ref={(input) => {this.input = input}}

/>

<button onClick={this.handleBtnClick}>提交</button>

</div>

<ul ref={(ul) => {this.ul = ul}}>

{ this.getTodoItem() }

</ul>

</Fragment>

);

}

getTodoItem() {

return this.state.list.map((item, index) => {

return (

<TodoItem

key={index}

content={item}

index={index}

deleteItem={this.handleItemDelete}

/>

)

});

}

handleInputChange() {

const value = this.input.value;

this.setState(() => ({

inputValue: value

}));

}

handleBtnClick() {

this.setState((prevState) => ({

list: [...prevState.list, prevState.inputValue],

inputValue: ''

}),

() => {

console.log(this.ul.querySelectorAll('div').length);

});

}

handleItemDelete(index) {

this.setState((prevState) => {

const list = [...prevState.list];

list.splice(index, 1);

return {list};

});

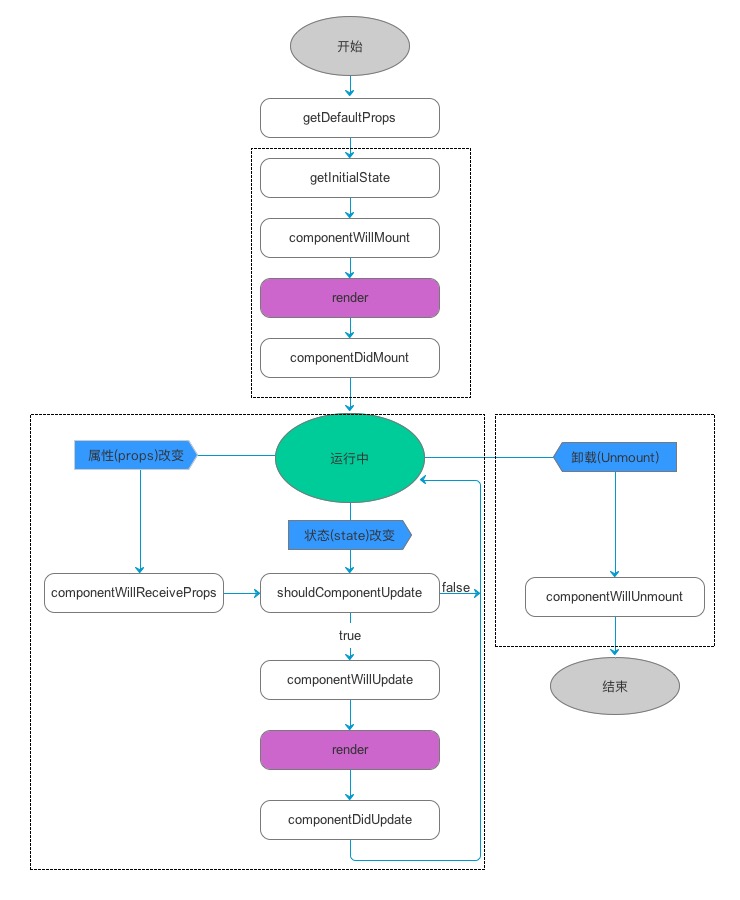
}

}

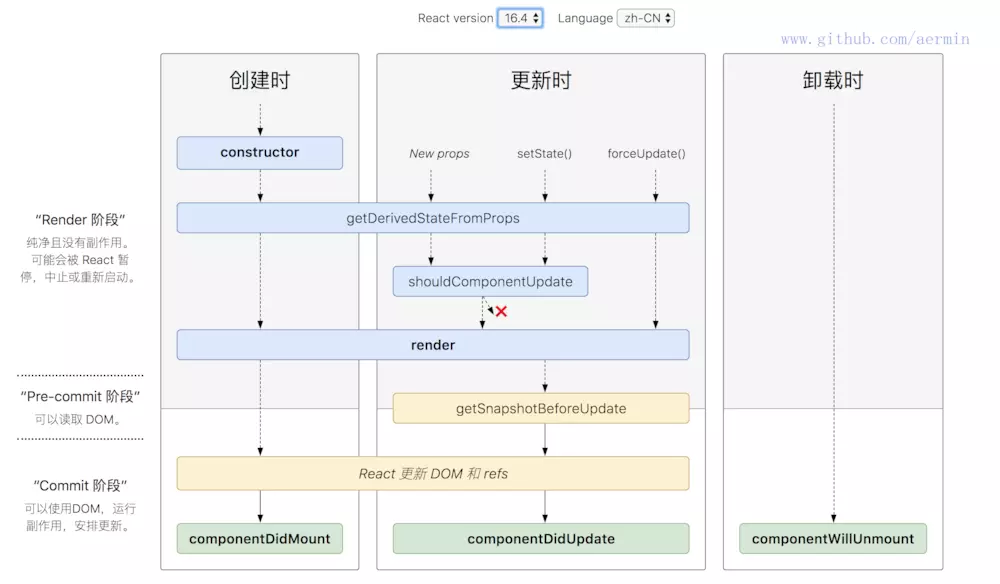
export default TodoList;

**React生命周期函数**

老版图解



新版16.4图解



**生命周期应用场景**

1. 子组件使用shouldComponentUpdate来提升性能

import React, { Component } from 'react';

import PropTypes from 'prop-types';

class TodoItem extends Component {

constructor(props) {

super(props);

this.handleClick = this.handleClick.bind(this);

}

shouldComponentUpdate(nextProps, nextState) {

if(nextProps.content != this.props.content) {

return true;

}else{

return false;

}

}

render() {

const { content, test } = this.props;

return (

<div onClick={this.handleClick}>

{content}

</div>

)

}

handleClick() {

const { deleteItem, index } = this.props;

deleteItem(index);

}

}

TodoItem.propTypes = {

content: PropTypes.string,

deleteItem: PropTypes.func,

index: PropTypes.number

}

export default TodoItem;

1. Ajax数据请求一般放在componentDidMount函数，componentWillMount虽然也执行一次，但是有些高级用法会有冲突。原则上也能放入constructor里执行。

**Ajax请求**

1. 添加ajax模块

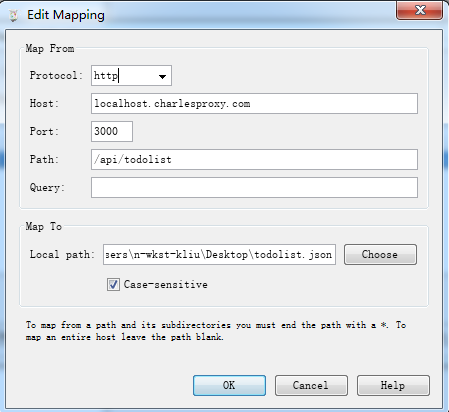
yarn add axios

1. 创建本地todolist.json文件

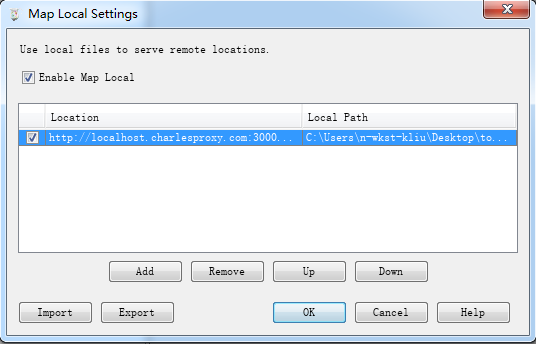
["Dell", "Lee", "Tom"]

1. 使用charles(4.2.8)来模拟rest数据

Tools -> Edit Mapping，配置如下（Host要使用localhost.charlesproxy.com）。



勾选“Enable Map Local”和配置项



1. 在componentDidMount里使用ajax获取数据

import React, { Component, Fragment } from 'react';

import axios from 'axios';

import TodoItem from './TodoItem.js'

class TodoList extends Component {

constructor(props) {

super(props);

this.state = {

inputValue: '',

list: []

};

this.handleInputChange = this.handleInputChange.bind(this);

this.handleBtnClick = this.handleBtnClick.bind(this);

this.handleItemDelete = this.handleItemDelete.bind(this);

}

componentDidMount() {

axios.get('/api/todolist')

.then((res) => {

this.setState(() => ({

list: [...res.data]

}));

})

.catch(() => {alert('error')});

}

render() {

return (

<Fragment>

<div>

<label htmlFor="inputArea">输入内容：</label>

<input

id="inputArea"

value={this.state.inputValue}

onChange={this.handleInputChange}

/>

<button onClick={this.handleBtnClick}>提交</button>

</div>

<ul>

{ this.getTodoItem() }

</ul>

</Fragment>

);

}

getTodoItem() {

return this.state.list.map((item, index) => {

return (

<TodoItem

key={index}

content={item}

index={index}

deleteItem={this.handleItemDelete}

/>

)

});

}

handleInputChange(e) {

const value = e.target.value;

this.setState(() => ({

inputValue: value

}));

}

handleBtnClick() {

this.setState((prevState) => ({

list: [...prevState.list, prevState.inputValue],

inputValue: ''

}));

}

handleItemDelete(index) {

this.setState((prevState) => {

const list = [...prevState.list];

list.splice(index, 1);

return {list};

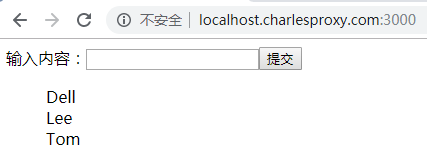
});

}

}

export default TodoList;

1. 访问React要用<http://localhost.charlesproxy.com:3000/>

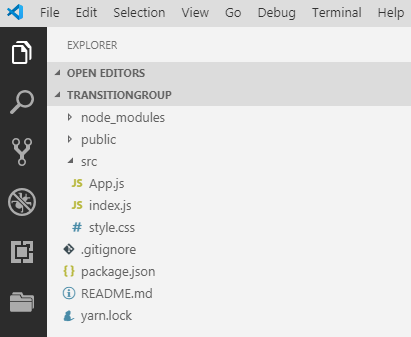


**使用react-transition-group实现动画**

1. 添加react-transition-group模块

yarn add react-transition-group

1. 项目结构



1. 项目源码

index.js源码

import React from 'react';

import ReactDOM from 'react-dom';

import App from './App.js';

ReactDOM.render(<App />, document.getElementById('root'));

App.js源码

unmountOnExit：当div隐藏时，移除该div

onEntered：钩子函数

appear：当首次打开页面时，也有动画效果

import React, { Component, Fragment } from 'react';

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

import './style.css';

class App extends Component {

constructor(props) {

super(props);

this.state = {

show: true

}

this.handleToggole = this.handleToggole.bind(this);

}

render() {

return (

<Fragment>

<div>

<CSSTransition

in={this.state.show}

timeout={1000}

classNames="fade"

unmountOnExit

onEntered={(el) => {el.style.color = 'blue'}}

appear={true}

>

<div>hello</div>

</CSSTransition>

<button onClick={this.handleToggole}>toggle</button>

</div>

</Fragment>

);

}

handleToggole() {

this.setState(() => ({

show: this.state.show ? false : true

}));

}

}

export default App;

style.css源码

.fade-enter, .fade-appear {

opacity: 0;

}

.fade-enter-active, .fade-appear-active {

opacity: 1;

transition: opacity 1s ease-in;

}

.fade-enter-done {

opacity: 1;

}

.fade-exit {

opacity: 1;

}

.fade-exit-active {

opacity: 0;

transition: opacity 1s ease-in;

}

.fade-exit-done {

opacity: 0;

}

1. react-transition-group的使用

App.js使用react-transition-group

import React, { Component, Fragment } from 'react';

import { CSSTransition, TransitionGroup } from 'react-transition-group';

import './style.css';

class App extends Component {

constructor(props) {

super(props);

this.state = {

list: []

}

this.handleAddItem = this.handleAddItem.bind(this);

}

render() {

return (

<Fragment>

<TransitionGroup>

{

this.state.list.map((item, index) => {

return (

<CSSTransition

in={this.state.show}

timeout={1000}

classNames="fade"

unmountOnExit

onEntered={(el) => { el.style.color = 'blue' }}

appear={true}

key={index}

>

<div>{item}</div>

</CSSTransition>

)

})

}

</TransitionGroup>

<div>

<button onClick={this.handleAddItem}>toggle</button>

</div>

</Fragment>

);

}

handleAddItem() {

this.setState((prevState) => {

return {

list: [...prevState.list, 'item']

}

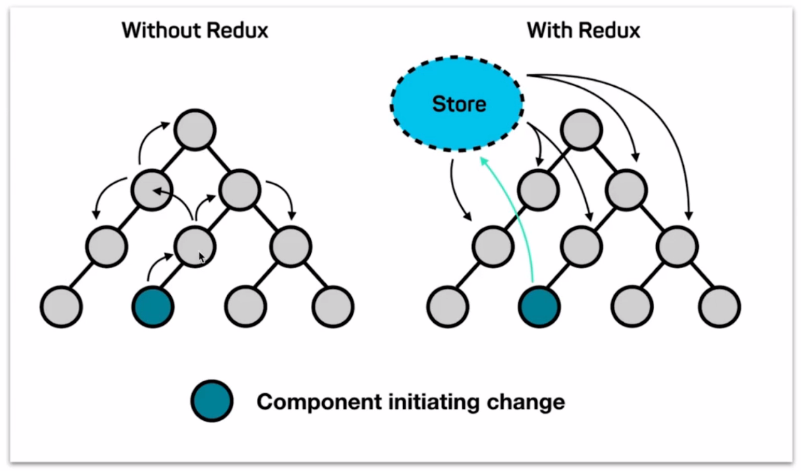
});

}

}

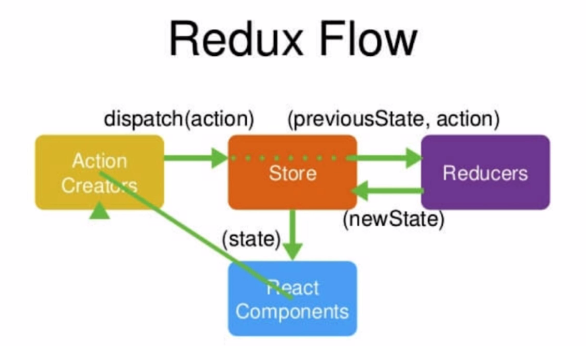
export default App;

**Redux简介**



Redux = Reducer + Flux

**Redux工作流程**



**使用AntD实现TodoList布局**

1. 添加antd模块

yarn add antd

1. TodoList布局实现

import React, { Component } from 'react';

import 'antd/dist/antd.css';

import { Input, Button, List } from 'antd';

const data = [

'Racing car sprays burning fuel into crowd.',

'Japanese princess to wed commoner.',

'Australian walk 100km after outback crash.'

];

class TodoList extends Component {

render() {

return (

<div style={{margin: '10px'}}>

<div>

<Input placeholder='todo info' style={{width: '300px', marginRight: '10px'}}/>

<Button type="primary">提交</Button>

</div>

<List

style={{marginTop: '10px', width:'350px'}}

bordered

dataSource={data}

renderItem={item => (<List.Item>{item}</List.Item>)}

/>

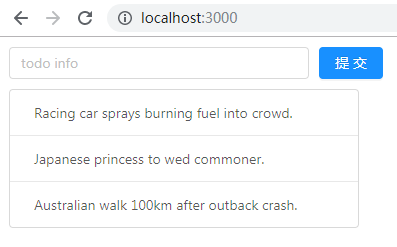
</div>

)

}

}

export default TodoList;



**使用Redux实现TodoList**

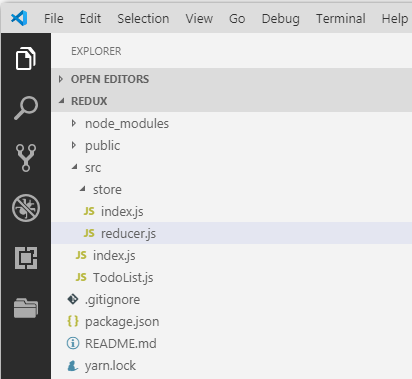
1. 添加redux模块

yarn add redux

1. 添加antd模块

yarn add antd

1. 项目结构



1. 项目源码

\src\index.js源码

import React from 'react';

import ReactDOM from 'react-dom';

import TodoList from './TodoList';

ReactDOM.render(<TodoList />, document.getElementById('root'));

\src\TodoList.js源码

import React, { Component } from 'react';

import 'antd/dist/antd.css';

import { Input, Button, List } from 'antd';

import store from './store/';

class TodoList extends Component {

constructor(props) {

super(props);

this.state = store.getState();

this.handleInputChange = this.handleInputChange.bind(this);

this.handleBtnClick = this.handleBtnClick.bind(this);

this.handleStoreChange = this.handleStoreChange.bind(this);

store.subscribe(this.handleStoreChange);

}

render() {

return (

<div style={{margin: '10px'}}>

<div>

<Input value={this.state.inputValue}

placeholder='todo info'

style={{width: '300px', marginRight: '10px'}}

onChange={this.handleInputChange}

/>

<Button type="primary" onClick={this.handleBtnClick}>提交</Button>

</div>

<List

style={{marginTop: '10px', width:'350px'}}

bordered

dataSource={this.state.list}

renderItem={(item, index) => (<List.Item onClick={this.handleItemDelete.bind(this, index)}>{item}</List.Item>)}

/>

</div>

)

}

handleInputChange(e) {

const action = {

type: 'change\_input\_value',

value: e.target.value

};

store.dispatch(action);

}

handleBtnClick() {

const action = {

type: 'add\_todo\_item'

};

store.dispatch(action);

}

handleItemDelete(index) {

const action = {

type: 'delete\_todo\_item',

index

};

store.dispatch(action);

}

handleStoreChange(){

this.setState(store.getState());

}

}

export default TodoList;

\src\store\index.js源码

import { createStore } from 'redux';

import reducer from './reducer.js';

//第二个参数为使用redux调试插件的默认参数

const store = createStore(

reducer,

window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_()

);

export default store;

\src\store\reduce.js源码

const defaultState = {

inputValue: 'Hello',

list: ['Learn React', 'Learn Vue', 'Learn Angular']

};

//reduce可以接受state，但是绝不能修改state

export default (state = defaultState, action) => {

if(action.type === 'change\_input\_value') {

const newState = JSON.parse(JSON.stringify(state));

newState.inputValue = action.value;

return newState;

}

if(action.type === 'add\_todo\_item') {

const newState = JSON.parse(JSON.stringify(state));

newState.list.push(newState.inputValue);

newState.inputValue = '';

return newState;

}

if(action.type === 'delete\_todo\_item') {

const newState = JSON.parse(JSON.stringify(state));

newState.list.splice(action.index, 1);

return newState;

}

return state;

}

1. actionType的拆分

action的type在TodoList和reduce里均有使用，如果不小心拼错，定位错误很耗时。所以建议在store文件夹下创建一个actionTypes.js来定义这些常量。

export const CHANGE\_INPUT\_VALUE = 'change\_input\_value';

export const ADD\_TODO\_ITEM = 'add\_todo\_item';

export const DELETE\_TODO\_ITEM = 'delete\_todo\_item';

1. 使用actionCreator统一创建action

为了方便action的统一管理，自动测试，在store文件夹下创建一个actionCreators.js文件

import { CHANGE\_INPUT\_VALUE, ADD\_TODO\_ITEM, DELETE\_TODO\_ITEM } from './actionTypes';

export const getInputChangeAction = (value) => ({

type: CHANGE\_INPUT\_VALUE,

value

});

export const getAddItemAction = () => ({

type: ADD\_TODO\_ITEM

});

export const getDeleteItemAction = (index) => ({

type: DELETE\_TODO\_ITEM,

index

});

TodoList.js修改如下：

import React, { Component } from 'react';

import 'antd/dist/antd.css';

import { Input, Button, List } from 'antd';

import store from './store/';

import { getInputChangeAction, getAddItemAction, getDeleteItemAction } from './store/actionCreators';

class TodoList extends Component {

constructor(props) {

super(props);

this.state = store.getState();

this.handleInputChange = this.handleInputChange.bind(this);

this.handleBtnClick = this.handleBtnClick.bind(this);

this.handleStoreChange = this.handleStoreChange.bind(this);

store.subscribe(this.handleStoreChange);

}

render() {

return (

<div style={{margin: '10px'}}>

<div>

<Input value={this.state.inputValue}

placeholder='todo info'

style={{width: '300px', marginRight: '10px'}}

onChange={this.handleInputChange}

/>

<Button type="primary" onClick={this.handleBtnClick}>提交</Button>

</div>

<List

style={{marginTop: '10px', width:'350px'}}

bordered

dataSource={this.state.list}

renderItem={(item, index) => (<List.Item onClick={this.handleItemDelete.bind(this, index)}>{item}</List.Item>)}

/>

</div>

)

}

handleInputChange(e) {

const action = getInputChangeAction(e.target.value);

store.dispatch(action);

}

handleBtnClick() {

const action = getAddItemAction();;

store.dispatch(action);

}

handleItemDelete(index) {

const action = getDeleteItemAction(index);

store.dispatch(action);

}

handleStoreChange(){

this.setState(store.getState());

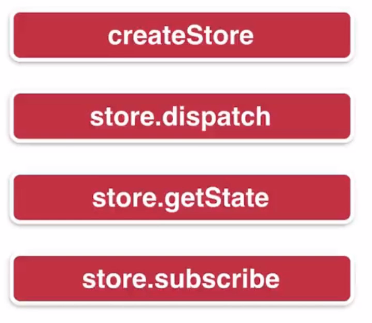
}

}

export default TodoList;

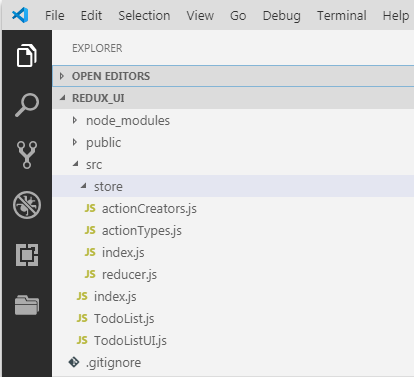
**Redux设计使用原则**

1. store必须是唯一的
2. 只有store能够改变自己的内容
3. Reduce必须是纯函数（纯函数指的是，给定固定的输入，就一定有固定的输出，而且不会有副作用）



**UI组件和容器组件**

1. UI组件仅负责显示，不负责逻辑，也叫傻瓜组件；容器组件仅关注业务逻辑
2. 项目结构



1. 上述项目中的TodoList拆分一个TodoListUI.js

TodoListUI.js源码

import React, { Component } from 'react';

import 'antd/dist/antd.css';

import { Input, Button, List } from 'antd';

class TodoListUI extends Component {

render() {

return (

<div style={{margin: '10px'}}>

<div>

<Input value={this.props.inputValue}

placeholder='todo info'

style={{width: '300px', marginRight: '10px'}}

onChange={this.props.handleInputChange}

/>

<Button type="primary" onClick={this.props.handleBtnClick}>提交</Button>

</div>

<List

style={{marginTop: '10px', width:'350px'}}

bordered

dataSource={this.props.list}

renderItem={(item, index) => (<List.Item onClick={(index) => this.props.handleItemDelete(index)}>{item}</List.Item>)}

/>

</div>

)

}

}

export default TodoListUI;

TodoList.js源码

import React, { Component } from 'react';

import store from './store/';

import { getInputChangeAction, getAddItemAction, getDeleteItemAction } from './store/actionCreators';

import TodoListUI from './TodoListUI';

class TodoList extends Component {

constructor(props) {

super(props);

this.state = store.getState();

this.handleInputChange = this.handleInputChange.bind(this);

this.handleBtnClick = this.handleBtnClick.bind(this);

this.handleStoreChange = this.handleStoreChange.bind(this);

this.handleItemDelete = this.handleItemDelete.bind(this);

store.subscribe(this.handleStoreChange);

}

render() {

return (

<TodoListUI

inputValue={this.state.inputValue}

list={this.state.list}

handleInputChange={this.handleInputChange}

handleBtnClick={this.handleBtnClick}

handleItemDelete={this.handleItemDelete}

/>

)

}

handleInputChange(e) {

const action = getInputChangeAction(e.target.value);

store.dispatch(action);

}

handleBtnClick() {

const action = getAddItemAction();;

store.dispatch(action);

}

handleItemDelete(index) {

const action = getDeleteItemAction(index);

store.dispatch(action);

}

handleStoreChange(){

this.setState(store.getState());

}

}

export default TodoList;

**无状态组件**

当一个组件只有render函数时就可以改成无状态组件，无状态组件没有相关的生命周期，性能更高。上述项目中的TodoListUI.js可以改写如下：

import React, { Component } from 'react';

import 'antd/dist/antd.css';

import { Input, Button, List } from 'antd';

const TodoListUI = (props) => {

return (

<div style={{ margin: '10px' }}>

<div>

<Input value={props.inputValue}

placeholder='todo info'

style={{ width: '300px', marginRight: '10px' }}

onChange={props.handleInputChange}

/>

<Button type="primary" onClick={props.handleBtnClick}>提交</Button>

</div>

<List

style={{ marginTop: '10px', width: '350px' }}

bordered

dataSource={props.list}

renderItem={(item, index) => (<List.Item onClick={() => props.handleItemDelete(index)}>{item}</List.Item>)}

/>

</div>

)

}

export default TodoListUI;

**Redux发送异步请求获取数据**

TodoList.js添加如下异步请求

componentDidMount() {

axios.get('/list.json').then((res) => {

const data = res.data;

const action = initListAction(data);

store.dispatch(action);

});

}

reduce.js添加如下代码

if(action.type === INIT\_LIST\_ACTION) {

const newState = JSON.parse(JSON.stringify(state));

newState.list = action.data;

return newState;

}

actionCreators.js添加如下代码

export const initListAction = (data) => ({

type: INIT\_LIST\_ACTION,

data

});

其他细节略。

**使用Redux-thunk中间件实现ajax数据请求**

1. 添加redux-thunk模块

yarn add redux-thunk

1. 项目结构同上
2. 源码改动如下

\src\store\index.js

import { createStore, applyMiddleware, compose } from 'redux';

import reducer from './reducer.js';

import thunk from 'redux-thunk';

const composeEnhancers = window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_ ? window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_({}) : compose;

const enhancer = composeEnhancers(

applyMiddleware(thunk),

);

const store = createStore(reducer, enhancer);

export default store;

\src\TodoList.js改动componentDidMount方法

componentDidMount() {

const action = getTodoList();

store.dispatch(action);

}

\src\stroe\actionCreators.js添加getTodoList()方法

export const getTodoList = () => {

return (dispatch) => {

axios.get('/list.json').then((res) => {

const data = res.data;

const action = initListAction(data);

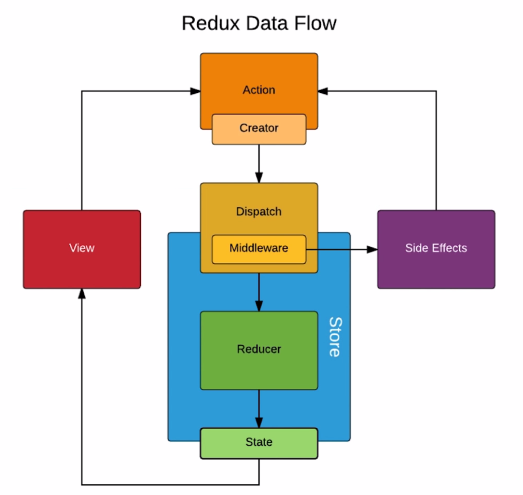
dispatch(action);

})

}

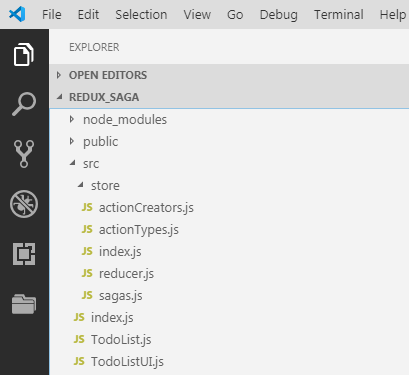
}

**什么是Redux的中间件**



**Redux-saga中间件的使用**

1. 添加redux-saga模块
2. 工程结构图



1. 主要源码改动

\src\store\index.js

import { createStore, applyMiddleware, compose } from 'redux';

import reducer from './reducer.js';

import createSagaMiddleware from 'redux-saga';

import todoSagas from './sagas';

const sagaMiddleware = createSagaMiddleware();

const composeEnhancers = window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_ ? window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_({}) : compose;

const enhancer = composeEnhancers(

applyMiddleware(sagaMiddleware),

);

const store = createStore(reducer, enhancer);

sagaMiddleware.run(todoSagas);

export default store;

\src\store\actionCreators.js

import { CHANGE\_INPUT\_VALUE, ADD\_TODO\_ITEM, DELETE\_TODO\_ITEM, INIT\_LIST\_ACTION, GET\_INIT\_LIST } from './actionTypes';

export const getInputChangeAction = (value) => ({

type: CHANGE\_INPUT\_VALUE,

value

});

export const getAddItemAction = () => ({

type: ADD\_TODO\_ITEM

});

export const getDeleteItemAction = (index) => ({

type: DELETE\_TODO\_ITEM,

index

});

export const initListAction = (data) => ({

type: INIT\_LIST\_ACTION,

data

});

export const getInitList = () => ({

type: GET\_INIT\_LIST

});

\src\store\actionTpyes.js

export const CHANGE\_INPUT\_VALUE = 'change\_input\_value';

export const ADD\_TODO\_ITEM = 'add\_todo\_item';

export const DELETE\_TODO\_ITEM = 'delete\_todo\_item';

export const INIT\_LIST\_ACTION = 'init\_list';

export const GET\_INIT\_LIST = 'get\_init\_lits';

\src\store\sagas.js

import { takeEvery, put } from 'redux-saga/effects';

import { GET\_INIT\_LIST } from './actionTypes';

import { initListAction } from './actionCreators';

import axios from 'axios';

function\* getInitList() {

try {

const res = yield axios.get('/list.json');

const action = initListAction(res.data);

yield put(action);

}catch(e) {

console.log('list.json网络请求失败');

}

}

// generator 函数

function\* mySaga() {

yield takeEvery(GET\_INIT\_LIST, getInitList);

}

export default mySaga;

\src\TodoList.js

import React, { Component } from 'react';

import store from './store/';

import { getInputChangeAction, getAddItemAction, getDeleteItemAction, getInitList } from './store/actionCreators';

import TodoListUI from './TodoListUI';

class TodoList extends Component {

constructor(props) {

super(props);

this.state = store.getState();

this.handleInputChange = this.handleInputChange.bind(this);

this.handleBtnClick = this.handleBtnClick.bind(this);

this.handleStoreChange = this.handleStoreChange.bind(this);

this.handleItemDelete = this.handleItemDelete.bind(this);

store.subscribe(this.handleStoreChange);

}

render() {

return (

<TodoListUI

inputValue={this.state.inputValue}

list={this.state.list}

handleInputChange={this.handleInputChange}

handleBtnClick={this.handleBtnClick}

handleItemDelete={this.handleItemDelete}

/>

)

}

componentDidMount() {

const action = getInitList();

store.dispatch(action);

}

handleInputChange(e) {

const action = getInputChangeAction(e.target.value);

store.dispatch(action);

}

handleBtnClick() {

const action = getAddItemAction();;

store.dispatch(action);

}

handleItemDelete(index) {

const action = getDeleteItemAction(index);

store.dispatch(action);

}

handleStoreChange(){

this.setState(store.getState());

}

}

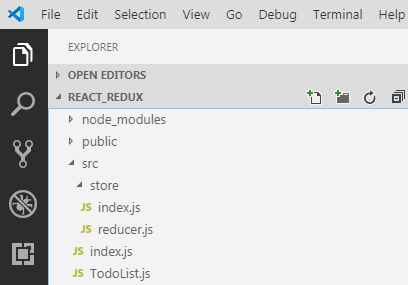
export default TodoList;

**React-redux的使用**

1. 添加react-redux模块

yarn add react-redux

1. 项目结构



1. 项目源码

\store\index.js

import { createStore } from 'redux';

import reducer from './reducer.js';

//第二个参数为使用redux调试插件的默认参数

const store = createStore(

reducer,

window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_()

);

export default store;

\store\reducer.js

const defaultState = {

inputValue: 'hello world',

list: []

}

export default (state = defaultState, action) => {

if (action.type === 'change\_input\_value') {

const newState = JSON.parse(JSON.stringify(state));

newState.inputValue = action.value;

return newState;

}

if (action.type === 'add\_item') {

const newState = JSON.parse(JSON.stringify(state));

newState.list.push(state.inputValue);

newState.inputValue = '';

return newState;

}

if (action.type === 'delete\_item') {

const newState = JSON.parse(JSON.stringify(state));

newState.list.splice(action.index, 1);

return newState;

}

return state;

}

\src\index.js

import React from 'react';

import ReactDOM from 'react-dom';

import TodoList from './TodoList';

import { Provider } from 'react-redux';

import store from './store';

const App = (

<Provider store={store}>

<TodoList />

</Provider>

);

ReactDOM.render(App, document.getElementById('root'));

\src\TodoList.js

import React from 'react';

import { connect } from 'react-redux';

const TodoList = (props) => {

const { inputValue, changeInputValue, handleClick, handleDelete } = props;

return (

<div>

<div>

<input value={inputValue} onChange={changeInputValue} />

<button onClick={handleClick}>提交</button>

</div>

<ul>

{

props.list.map((item, index) => {

return <li onClick={() => { handleDelete(index) }} key={index}>{item}</li>

})

}

</ul>

</div>

)

}

const mapStateToProps = (state) => {

return {

inputValue: state.inputValue,

list: state.list

}

}

const mapDispatchToProps = (dispatch) => {

return {

changeInputValue(e) {

const action = {

type: 'change\_input\_value',

value: e.target.value

};

dispatch(action);

},

handleClick() {

const action = {

type: 'add\_item'

}

dispatch(action);

},

handleDelete(index) {

const action = {

type: 'delete\_item',

index: index

};

dispatch(action);

}

}

}

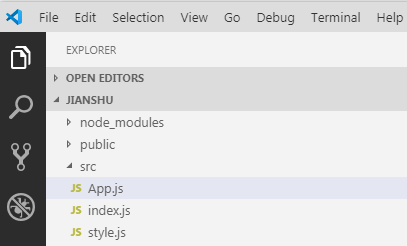
export default connect(mapStateToProps, mapDispatchToProps)(TodoList);

**Styled-Components与Reset.css结合使用**

1. 添加styled-components模块，方便样式在js里管理，作用域可控制，避免样式冲突。

yarn add styled-components

1. 项目目录结构



1. 项目源码

\src\App.js

import React, { Fragment } from 'react';

import Header from './common/header';

import { GlobalStyle } from './style';

function App() {

return (

<Fragment>

<GlobalStyle />

<div>Hello</div>

</Fragment>

);

}

export default App;

\src\index.js

import React from 'react';

import ReactDOM from 'react-dom';

import App from './App';

import './style.js';

ReactDOM.render(<App />, document.getElementById('root'));

\src\style.js，使用Reset.css中的全局样式，可以保持不同浏览器中样式保持一致。React4.x版本的styled-components弃用injectGlobal变为createGlobalStyle。

import { createGlobalStyle } from 'styled-components';

export const GlobalStyle = createGlobalStyle`

html, body, div, span, applet, object, iframe,

h1, h2, h3, h4, h5, h6, p, blockquote, pre,

a, abbr, acronym, address, big, cite, code,

del, dfn, em, img, ins, kbd, q, s, samp,

small, strike, strong, sub, sup, tt, var,

b, u, i, center,

dl, dt, dd, ol, ul, li,

fieldset, form, label, legend,

table, caption, tbody, tfoot, thead, tr, th, td,

article, aside, canvas, details, embed,

figure, figcaption, footer, header, hgroup,

menu, nav, output, ruby, section, summary,

time, mark, audio, video {

   margin: 0;

  padding: 0;

  border: 0;

  font-size: 100%;

  font: inherit;

   vertical-align: baseline;

}

article, aside, details, figcaption, figure,

footer, header, hgroup, menu, nav, section {

   display: block;

}

body {

   line-height: 1;

}

ol, ul {

   list-style: none;

}

blockquote, q {

  quotes: none;

}

blockquote:before, blockquote:after,

q:before, q:after {

  content: '';

  content: none;

}

table {

  border-collapse: collapse;

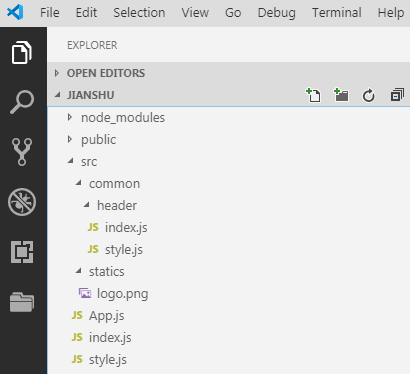
  border-spacing: 0;

}

`;

**使用styled-components完成Header组件布局**

1. 项目结构



1. 项目源码

\src\App.js

import React, { Fragment } from 'react';

import Header from './common/header';

import { GlobalStyle } from './style';

function App() {

return (

<Fragment>

<GlobalStyle />

<Header />

</Fragment>

);

}

export default App;

\src\common\header\index.js

import React, { Component } from 'react';

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

NavSearch,

Addition,

Button

} from './style';

class Header extends Component {

render() {

return (

<HeaderWrapper>

<Logo/>

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>Aa</NavItem>

<NavSearch></NavSearch>

</Nav>

<Addition>

<Button className='writting'>写文章</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

)

}

}

export default Header;

\src\common\header\style.js

import styled from 'styled-components';

import logoPic from '../../statics/logo.png';

export const HeaderWrapper = styled.div`

height: 56px;

border-bottom: 1px solid #f0f0f0;

`

export const Logo = styled.a.attrs({

href: '/'

})`

position: absolute;

top: 0;

left: 0;

display: block;

width: 100px;

height: 56px;

background: url(${logoPic});

background-size: contain;

`

export const Nav = styled.div`

width: 960px;

height: 100%;

padding-right: 70px;

box-sizing: border-box;

margin: 0 auto;

`

export const NavItem = styled.div`

line-height: 56px;

padding: 0 15px;

font-size: 17px;

color: #333;

&.left {

float: left;

}

&.right {

float: right;

color: #969696;

}

&.active {

color: #ea6f5a;

}

`

export const NavSearch = styled.input.attrs({

placeholder: '搜索'

})`

width: 160px;

height: 38px;

padding: 0 20px;

margin-top: 9px;

margin-left: 20px;

box-sizing: border-box;

border: none;

outline: none;

border-radius: 19px;

background: #eee;

font-size: 14px;

&::placeholder {

color: #999;

}

`

export const Addition = styled.div`

position: absolute;

right: 0;

top: 0;

height: 56px;

`

export const Button = styled.div`

float: right;

margin-top: 9px;

margin-right: 20px;

padding: 0 20px;

line-height: 38px;

border-radius: 19px;

border: 1px solid #ec6149;

font-size: 14px;

&.reg {

color: #ec6149;

}

&.writting {

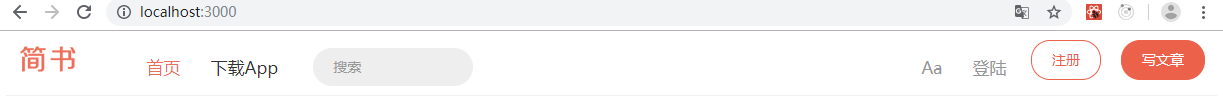
color: #fff;

background: #ec6149;

}

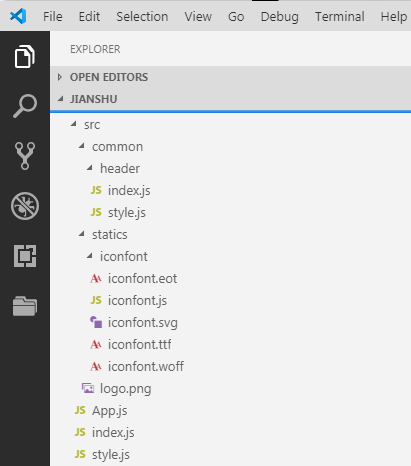
`;

1. 项目运行图



**使用iconfont嵌入头部组件**

1. 项目结构



1. 项目源码

\statics\iconfont\iconfont.js

import { createGlobalStyle } from 'styled-components';

export const GlobalIconStyle = createGlobalStyle`

@font-face {

font-family: "iconfont";

src: url('./iconfont.eot?t=1561705470345'); /\* IE9 \*/

src: url('./iconfont.eot?t=1561705470345#iefix') format('embedded-opentype'), /\* IE6-IE8 \*/

url('data:application/x-font-woff2;charset=utf-8;base64,') format('woff2'),

url('./iconfont.woff?t=1561705470345') format('woff'),

url('./iconfont.ttf?t=1561705470345') format('truetype'), /\* chrome, firefox, opera, Safari, Android, iOS 4.2+ \*/

url('./iconfont.svg?t=1561705470345#iconfont') format('svg'); /\* iOS 4.1- \*/

}

.iconfont {

font-family: "iconfont" !important;

font-size: 16px;

font-style: normal;

-webkit-font-smoothing: antialiased;

-moz-osx-font-smoothing: grayscale;

}

`;

\src\App.js

import React, { Fragment } from 'react';

import Header from './common/header';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

function App() {

return (

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<Header />

</Fragment>

);

}

export default App;

\src\common\header\index.js

import React, { Component } from 'react';

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

Addition,

Button

} from './style';

class Header extends Component {

render() {

return (

<HeaderWrapper>

<Logo/>

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<NavSearch></NavSearch>

<i className="iconfont">&#xe637;</i>

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

)

}

}

export default Header;

\src\common\header\style.js，添加如下代码

export const SearchWrapper = styled.div`

position: relative;

float: left;

.iconfont {

position: absolute;

right: 5px;

bottom: 5px;

width: 30px;

line-height: 30px;

border-radius: 15px;

text-align: center;

}

`

**搜索框动画实现效果**

1. 添加react-transition-group模块

yarn add react-gransition-group

1. 源代码更改

\src\common\header\index.js

import React, { Component } from 'react';

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

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

Addition,

Button

} from './style';

class Header extends Component {

constructor(props) {

super(props);

this.state = {

focused: false

}

this.handleInputFocus = this.handleInputFocus.bind(this);

this.handleInputBlur = this.handleInputBlur.bind(this);

}

render() {

return (

<HeaderWrapper>

<Logo/>

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<CSSTransition

in={this.state.focused}

timeout={200}

classNames="slide"

>

<NavSearch

className={this.state.focused ? 'focused' : ''}

onFocus={this.handleInputFocus}

onBlur={this.handleInputBlur}

>

</NavSearch>

</CSSTransition>

<i className={this.state.focused ? 'focused iconfont' : 'iconfont'}>

&#xe637;

</i>

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

)

}

handleInputFocus() {

this.setState({

focused: true

})

}

handleInputBlur() {

this.setState({

focused: false

})

}

}

export default Header;

\src\common\header\style.js添加如下代码，transition的样式最后放到NavSearch。

export const SearchWrapper = styled.div`

position: relative;

float: left;

.slide-enter {

width: 160px;

transition: all .2s ease-out;

}

.slide-enter-active {

width: 240px;

}

.slide-exit {

transition: all .2s ease-out;

}

.slide-exit-active {

width: 160px;

}

.iconfont {

position: absolute;

right: 5px;

bottom: 5px;

width: 30px;

line-height: 30px;

border-radius: 15px;

text-align: center;

&.focused {

background: #777;

color: #fff;

}

}

`

export const NavSearch = styled.input.attrs({

placeholder: '搜索'

})`

width: 160px;

height: 38px;

padding: 0 30px 0 20px;

margin-top: 9px;

margin-left: 20px;

box-sizing: border-box;

border: none;

outline: none;

border-radius: 19px;

background: #eee;

font-size: 14px;

color: #666;

&::placeholder {

color: #999;

}

&.focused {

width: 240px;

}

`

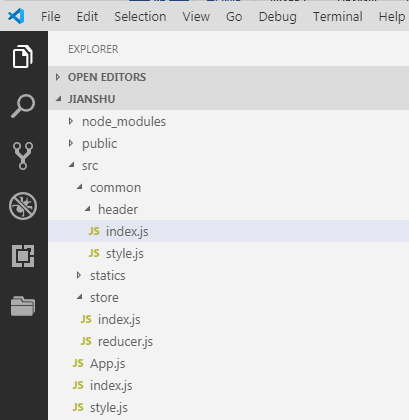
**使用React-Redux进行应用数据的管理**

1. 添加redux和react-redux模块

yarn add redux

yarn add react-redux

1. 工程目录如下



1. 源码改动

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import Header from './common/header';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<Header />

</Fragment>

</Provider>

);

}

export default App;

\src\store\index.js

import { createStore } from 'redux';

import reducer from './reducer';

const store = createStore(reducer);

export default store;

\src\store\reducer.js

const defaultState = {

focused: false

};

export default (state = defaultState, action) => {

if(action.type === 'search\_focus') {

return {

focused: true

}

}

if(action.type === 'search\_blur') {

return {

focused: false

}

}

return state;

}

\src\common\hearder\index.js

import React from 'react';

import { connect } from 'react-redux';

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

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

Addition,

Button

} from './style';

const Header = (props) => {

return (

<HeaderWrapper>

<Logo />

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<CSSTransition

in={props.focused}

timeout={200}

classNames="slide"

>

<NavSearch

className={props.focused ? 'focused' : ''}

onFocus={props.handleInputFocus}

onBlur={props.handleInputBlur}

>

</NavSearch>

</CSSTransition>

<i className={props.focused ? 'focused iconfont' : 'iconfont'}>

&#xe637;

</i>

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

);

}

const mapStateToProps = (state) => {

return {

focused: state.focused

}

}

const mapDispathToProps = (dispatch) => {

return {

handleInputFocus() {

const action = {

type: 'search\_focus'

};

dispatch(action);

},

handleInputBlur() {

const action = {

type: 'search\_blur'

}

dispatch(action);

}

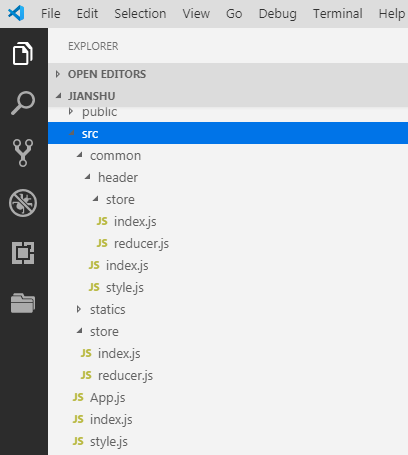
}

}

export default connect(mapStateToProps, mapDispathToProps)(Header);

**使用combineReducers完成对store数据的拆分管理**

1. 项目结构



1. 源码改动

\src\store\index.js

import { createStore, compose } from 'redux';

import reducer from './reducer';

const composeEnhancers = window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_COMPOSE\_\_ || compose;

const store = createStore(reducer, composeEnhancers());

export default store;

\src\store\reducer.js

import { combineReducers } from 'redux';

import { reducer as headerReducer } from '../common/header/store';

const reducer = combineReducers({

header: headerReducer

});

export default reducer;

\src\common\header\store\index.js

import reducer from './reducer';

export { reducer };

\src\common\header\store\reducer.js

const defaultState = {

focused: false

};

export default (state = defaultState, action) => {

if(action.type === 'search\_focus') {

return {

focused: true

}

}

if(action.type === 'search\_blur') {

return {

focused: false

}

}

return state;

}

\src\common\header\index.js修改focused的值

const mapStateToProps = (state) => {

return {

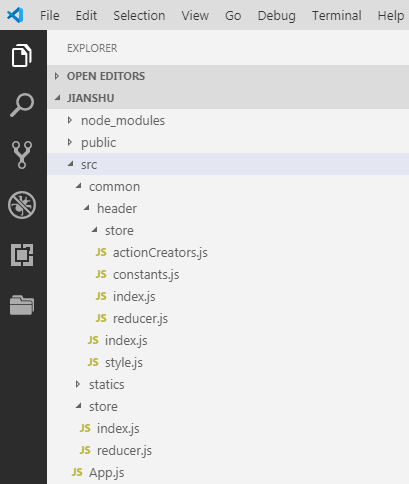
focused: state.header.focused

}

}

**actionCreators与constants的拆分**

1. 目录结构



1. 源码改动

\src\common\header\store\actionCreators.js

import \* as constants from './constants';

export const searchFocus = () => ({

type: constants.SEARCH\_FOUCUS

});

export const searchBlur = () => ({

type: constants.SEARCH\_BLUR

});

\src\common\header\store\constants.js

export const SEARCH\_FOUCUS = 'header/SEARCH\_FOCUS';

export const SEARCH\_BLUR = 'hearder/SEARCH\_BLUR';

\src\common\header\store\index.js

import reducer from './reducer';

import \* as actionCreators from './actionCreators';

import \* as constants from './constants';

export { reducer, actionCreators, constants };

\src\common\header\store\reducer.js

import \* as constants from './constants';

const defaultState = {

focused: false

};

export default (state = defaultState, action) => {

if(action.type === constants.SEARCH\_FOUCUS) {

return {

focused: true

}

}

if(action.type === constants.SEARCH\_BLUR) {

return {

focused: false

}

}

return state;

}

\src\common\header\index.js

import React from 'react';

import { connect } from 'react-redux';

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

import { actionCreators } from './store';

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

Addition,

Button

} from './style';

const Header = (props) => {

return (

<HeaderWrapper>

<Logo />

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<CSSTransition

in={props.focused}

timeout={200}

classNames="slide"

>

<NavSearch

className={props.focused ? 'focused' : ''}

onFocus={props.handleInputFocus}

onBlur={props.handleInputBlur}

>

</NavSearch>

</CSSTransition>

<i className={props.focused ? 'focused iconfont' : 'iconfont'}>

&#xe637;

</i>

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

);

}

const mapStateToProps = (state) => {

return {

focused: state.header.focused

}

}

const mapDispathToProps = (dispatch) => {

return {

handleInputFocus() {

dispatch(actionCreators.searchFocus());

},

handleInputBlur() {

dispatch(actionCreators.searchBlur());

}

}

}

export default connect(mapStateToProps, mapDispathToProps)(Header);

**使用immutable.js来管理store中的数据**

1. 安装immutable模块

yarn add immutable

1. 源码改动

\src\common\header\store\reducer.js

import \* as constants from './constants';

import { fromJS } from 'immutable';

const defaultState = fromJS({

focused: false

});

export default (state = defaultState, action) => {

if(action.type === constants.SEARCH\_FOUCUS) {

// immutable对象的set方法，会结合之前immutable对象的值

// 和设置的值，返回一个全新的对象

return state.set('focused', true);

}

if(action.type === constants.SEARCH\_BLUR) {

return state.set('focused', false);

}

return state;

}

\src\common\header\index.js

const mapStateToProps = (state) => {

return {

focused: state.header.get('focused')

}

}

**使用redux-immutable统一数据格式**

1. 添加redux-immutable模块

yarn add redux-immutable

1. 源码修改

\src\store\reducer.js

import { combineReducers } from 'redux-immutable';

import { reducer as headerReducer } from '../common/header/store';

const reducer = combineReducers({

header: headerReducer

});

export default reducer;

\src\common\header\index.js

const mapStateToProps = (state) => {

return {

//或者focused: state.get('header').get('focused')

focused: state.getIn(['header', 'focused'])

}

}

**热门搜索样式布局**

1. 源码修改

\src\common\header\index.js

import React from 'react';

import { connect } from 'react-redux';

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

import { actionCreators } from './store';

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

SearchInfo,

SearchInfoTitle,

SearchInfoSwitch,

SearchInfoList,

SearchInfoItem,

Addition,

Button

} from './style';

const getListArea = (show) => {

if(show) {

return (

<SearchInfo>

<SearchInfoTitle>

热门搜索

<SearchInfoSwitch>换一批</SearchInfoSwitch>

</SearchInfoTitle>

<SearchInfoList>

<SearchInfoItem>教育</SearchInfoItem>

<SearchInfoItem>教育</SearchInfoItem>

<SearchInfoItem>教育</SearchInfoItem>

<SearchInfoItem>教育</SearchInfoItem>

<SearchInfoItem>教育</SearchInfoItem>

<SearchInfoItem>教育</SearchInfoItem>

</SearchInfoList>

</SearchInfo>

)

}else {

return null;

}

}

const Header = (props) => {

return (

<HeaderWrapper>

<Logo />

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<CSSTransition

in={props.focused}

timeout={200}

classNames="slide"

>

<NavSearch

className={props.focused ? 'focused' : ''}

onFocus={props.handleInputFocus}

onBlur={props.handleInputBlur}

>

</NavSearch>

</CSSTransition>

<i className={props.focused ? 'focused iconfont' : 'iconfont'}>

&#xe637;

</i>

{getListArea(props.focused)}

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

);

}

const mapStateToProps = (state) => {

return {

//或者focused: state.get('header').get('focused')

focused: state.getIn(['header', 'focused'])

}

}

const mapDispathToProps = (dispatch) => {

return {

handleInputFocus() {

dispatch(actionCreators.searchFocus());

},

handleInputBlur() {

dispatch(actionCreators.searchBlur());

}

}

}

export default connect(mapStateToProps, mapDispathToProps)(Header);

\src\common\header\style.js添加如下样式

export const SearchInfo = styled.div`

position: absolute;

left: 0;

top: 56px;

width: 240px;

padding: 0 20px;

box-shadow: 0 0 8px rgba(0, 0, 0, .2);

`;

export const SearchInfoTitle = styled.div`

margin-top: 20px;

margin-bottom: 15px;

line-height: 20px;

font-size: 14px;

color:#969696;

`;

export const SearchInfoSwitch = styled.span`

float: right;

font-size: 13px;

`;

export const SearchInfoList = styled.div`

overflow: hidden;

`;

export const SearchInfoItem = styled.a`

display: block;

float: left;

line-height: 20px;

padding: 0 5px;

margin-right: 10px;

margin-bottom: 15px;

font-size: 12px;

border: 1px solid #ddd;

color: #787878;

border-radius: 3px;

`;



**Ajax获取数据以及代码优化**

1. 添加redux-thunk模块

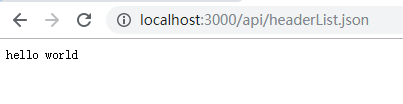
yarn add redux-thunk

1. 添加axios模块

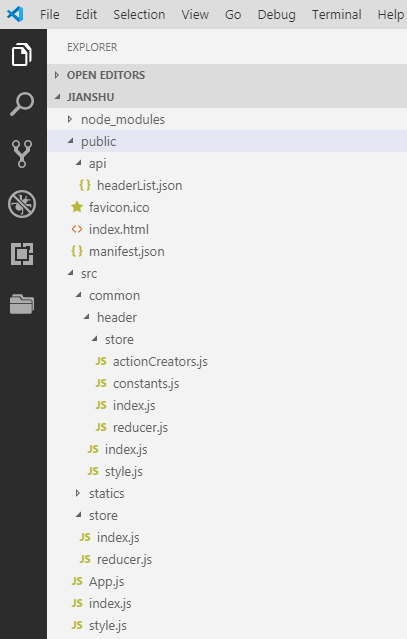
yarn add axios

1. 模拟rest api数据，在public下创建\api\headerList.json文件作为假数据。

hello world



1. 项目结构



1. 源码改动

\public\api\headerList.json

{

"success": true,

"data": ["区块链","小程序","vue","毕业","PHP","故事","flutter","理财","美食","投稿","手帐","书法","PPT","穿搭","打碗碗花","简书","姥姥的澎湖湾","设计","创业","交友","籽盐","教育","思维导图","疯哥哥","梅西","时间管理","golang","连载","自律","职场","考研","慢世人","悦欣","一纸vr","spring","eos","足球","程序员","林露含","彩铅","金融","木风杂谈","日更","成长","外婆是方言","docker"]

}

\src\common\header\store\actionCreators.js

import \* as constants from './constants';

import { fromJS } from 'immutable';

import axios from 'axios';

const changeList = (data) => ({

type: constants.CHANGE\_LIST,

data: fromJS(data)

});

export const searchFocus = () => ({

type: constants.SEARCH\_FOCUS

});

export const searchBlur = () => ({

type: constants.SEARCH\_BLUR

});

export const getList = () => {

return (dispatch) => {

axios.get('/api/headerList.json').then((res) => {

const data = res.data;

dispatch(changeList(data.data));

}).catch(() => {

console.log('error');

});

}

};

\src\common\header\store\constants.js

export const SEARCH\_FOCUS = 'header/SEARCH\_FOCUS';

export const SEARCH\_BLUR = 'hearder/SEARCH\_BLUR';

export const CHANGE\_LIST = 'header/CHANGE\_LIST';

\src\common\header\store\reducer.js

import \* as constants from './constants';

import { fromJS } from 'immutable';

const defaultState = fromJS({

focused: false,

list: []

});

export default (state = defaultState, action) => {

switch(action.type) {

case constants.SEARCH\_FOCUS:

return state.set('focused', true);

case constants.SEARCH\_BLUR:

return state.set('focused', false);

case constants.CHANGE\_LIST:

return state.set('list', action.data);

default:

return state;

}

}

\src\common\header\index.js

import React, { Component } from 'react';

import { connect } from 'react-redux';

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

import { actionCreators } from './store';

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

SearchInfo,

SearchInfoTitle,

SearchInfoSwitch,

SearchInfoList,

SearchInfoItem,

Addition,

Button

} from './style';

class Header extends Component {

getListArea() {

const { focused, list } = this.props;

if(focused) {

return (

<SearchInfo>

<SearchInfoTitle>

热门搜索

<SearchInfoSwitch>换一批</SearchInfoSwitch>

</SearchInfoTitle>

<SearchInfoList>

{

list.map((item) => {

return <SearchInfoItem key={item}>{item}</SearchInfoItem>

})

}

</SearchInfoList>

</SearchInfo>

)

}else {

return null;

}

}

render() {

const { focused, handleInputFocus, handleInputBlur } = this.props;

return (

<HeaderWrapper>

<Logo />

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<CSSTransition

in={focused}

timeout={200}

classNames="slide"

>

<NavSearch

className={focused ? 'focused' : ''}

onFocus={handleInputFocus}

onBlur={handleInputBlur}

>

</NavSearch>

</CSSTransition>

<i className={focused ? 'focused iconfont' : 'iconfont'}>

&#xe637;

</i>

{this.getListArea(focused)}

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

);

}

}

const mapStateToProps = (state) => {

return {

//或者focused: state.get('header').get('focused')

focused: state.getIn(['header', 'focused']),

list: state.getIn(['header', 'list'])

}

}

const mapDispathToProps = (dispatch) => {

return {

handleInputFocus() {

dispatch(actionCreators.getList());

dispatch(actionCreators.searchFocus());

},

handleInputBlur() {

dispatch(actionCreators.searchBlur());

}

}

}

export default connect(mapStateToProps, mapDispathToProps)(Header);

**热门搜索换页功能实现**

1. 源码修改

\src\common\header\store\actionCreators.js

import \* as constants from './constants';

import { fromJS } from 'immutable';

import axios from 'axios';

const changeList = (data) => ({

type: constants.CHANGE\_LIST,

data: fromJS(data),

totalPage: Math.ceil(data.length / 10)

});

export const searchFocus = () => ({

type: constants.SEARCH\_FOCUS

});

export const searchBlur = () => ({

type: constants.SEARCH\_BLUR

});

export const mouseEnter = () => ({

type: constants.MOUSE\_ENTER

});

export const mouseLeave = () => ({

type: constants.MOUSE\_LEAVE

});

export const changePage = (page) => ({

type: constants.CHANGE\_PAGE,

page

});

export const getList = () => {

return (dispatch) => {

axios.get('/api/headerList.json').then((res) => {

const data = res.data;

dispatch(changeList(data.data));

}).catch(() => {

console.log('error');

});

}

};

\src\common\header\store\constants.js

export const SEARCH\_FOCUS = 'header/SEARCH\_FOCUS';

export const SEARCH\_BLUR = 'hearder/SEARCH\_BLUR';

export const CHANGE\_LIST = 'header/CHANGE\_LIST';

export const MOUSE\_ENTER = 'header/MOUSE\_ENTER';

export const MOUSE\_LEAVE = 'header/MOUSE\_LEAVE';

export const CHANGE\_PAGE = 'header/CHANGE\_PAGE';

\src\common\header\store\reducer.js

import \* as constants from './constants';

import { fromJS } from 'immutable';

const defaultState = fromJS({

focused: false,

mouseIn: false,

list: [],

page: 1,

totalPage: 1

});

export default (state = defaultState, action) => {

switch(action.type) {

case constants.SEARCH\_FOCUS:

return state.set('focused', true);

case constants.SEARCH\_BLUR:

return state.set('focused', false);

case constants.CHANGE\_LIST:

//或者return state.set('list', action.data).set('totalPage', action.totalPage);

return state.merge({

list: action.data,

totalPage: action.totalPage

});

case constants.MOUSE\_ENTER:

return state.set('mouseIn', true);

case constants.MOUSE\_LEAVE:

return state.set('mouseIn', false);

case constants.CHANGE\_PAGE:

return state.set('page', action.page);

default:

return state;

}

}

\src\common\header\index.js

import React, { Component } from 'react';

import { connect } from 'react-redux';

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

import { actionCreators } from './store';

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

SearchInfo,

SearchInfoTitle,

SearchInfoSwitch,

SearchInfoList,

SearchInfoItem,

Addition,

Button

} from './style';

class Header extends Component {

getListArea() {

const { focused, list, page, totalPage, mouseIn, handleMouseEnter, handleMouseLeave, handleChangePage } = this.props;

const newList = list.toJS();

const pageList = [];

if(newList.length){

for (let i = (page - 1) \* 10; i < page \* 10; i++) {

pageList.push(

<SearchInfoItem key={newList[i]}>{newList[i]}</SearchInfoItem>

)

}

}

if(focused || mouseIn) {

return (

<SearchInfo

onMouseEnter={handleMouseEnter}

onMouseLeave={handleMouseLeave}

>

<SearchInfoTitle>

热门搜索

<SearchInfoSwitch onClick={() => handleChangePage(page, totalPage)}>换一批</SearchInfoSwitch>

</SearchInfoTitle>

<SearchInfoList>

{pageList}

</SearchInfoList>

</SearchInfo>

)

}else {

return null;

}

}

render() {

const { focused, handleInputFocus, handleInputBlur } = this.props;

return (

<HeaderWrapper>

<Logo />

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

<NavItem className='right'>登陆</NavItem>

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<CSSTransition

in={focused}

timeout={200}

classNames="slide"

>

<NavSearch

className={focused ? 'focused' : ''}

onFocus={handleInputFocus}

onBlur={handleInputBlur}

>

</NavSearch>

</CSSTransition>

<i className={focused ? 'focused iconfont' : 'iconfont'}>

&#xe637;

</i>

{this.getListArea(focused)}

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

);

}

}

const mapStateToProps = (state) => {

return {

//或者focused: state.get('header').get('focused')

focused: state.getIn(['header', 'focused']),

list: state.getIn(['header', 'list']),

page: state.getIn(['header', 'page']),

totalPage: state.getIn(['header', 'totalPage']),

mouseIn: state.getIn(['header', 'mouseIn'])

}

}

const mapDispathToProps = (dispatch) => {

return {

handleInputFocus() {

dispatch(actionCreators.getList());

dispatch(actionCreators.searchFocus());

},

handleInputBlur() {

dispatch(actionCreators.searchBlur());

},

handleMouseEnter() {

dispatch(actionCreators.mouseEnter());

},

handleMouseLeave() {

dispatch(actionCreators.mouseLeave());

},

handleChangePage(page, totalPage) {

if (page < totalPage) {

dispatch(actionCreators.changePage(page + 1));

}else{

dispatch(actionCreators.changePage(1));

}

}

}

}

export default connect(mapStateToProps, mapDispathToProps)(Header);

**换页旋转动画效果的实现**

1. 源码改动

\src\common\hearder\index.js

<SearchInfoTitle>

热门搜索

<SearchInfoSwitch onClick={() => handleChangePage(page, totalPage, this.spinIcon)}>

<i ref={(icon) => {this.spinIcon = icon}} className="iconfont spin">&#xe851;</i>

换一批

</SearchInfoSwitch>

</SearchInfoTitle>

…

<i className={focused ? 'focused iconfont zoom' : 'iconfont zoom'}>

&#xe637;

</i

…

handleChangePage(page, totalPage, spin) {

let originAngle = spin.style.transform.replace(/[^0-9]/ig, '');

if(originAngle) {

originAngle = parseInt(originAngle, 10);

}else{

originAngle = 0;

}

spin.style.transform = 'rotate(' + (originAngle + 360) + 'deg)';

if (page < totalPage) {

dispatch(actionCreators.changePage(page + 1));

}else{

dispatch(actionCreators.changePage(1));

}

\src\common\header\style.js

export const SearchWrapper = styled.div`

position: relative;

float: left;

.zoom {

position: absolute;

right: 5px;

bottom: 5px;

width: 30px;

line-height: 30px;

border-radius: 15px;

text-align: center;

&.focused {

background: #777;

color: #fff;

}

}

`;

export const SearchInfoSwitch = styled.span`

float: right;

font-size: 13px;

.spin {

display: block;

float: left;

font-size: 12px;

margin-right: 2px;

transition: all .2s ease-in;

transform-origin: center center;

}

`;

**如何在React中使用路由功能**

1. 添加路由模块react-router-dom

yarn add react-router-dom

1. 项目源码修改

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import { BrowserRouter, Route } from 'react-router-dom';

import Header from './common/header';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<Header />

<BrowserRouter>

<div>

<Route path='/' exact render={() => <div>home</div>}></Route>

<Route path='/detail' exact render={() => <div>detail</div>}></Route>

</div>

</BrowserRouter>

</Fragment>

</Provider>

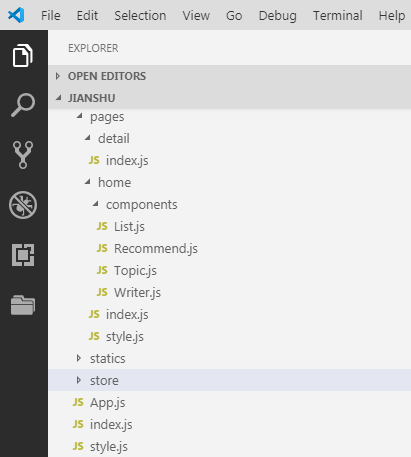
);

}

export default App;

**首页组件的拆分**

1. 项目结构



1. 源码改动

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import { BrowserRouter, Route } from 'react-router-dom';

import Header from './common/header';

import Home from './pages/home';

import Detail from './pages/detail';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<Header />

<BrowserRouter>

<div>

<Route path='/' exact component={Home}></Route>

<Route path='/detail' exact component={Detail}></Route>

</div>

</BrowserRouter>

</Fragment>

</Provider>

);

}

export default App;

\src\pages\detail\index.js

import React, {Component} from 'react';

class Detail extends Component {

render() {

return (

<div>Detail~</div>

)

}

}

export default Detail

\src\pages\home\components\List.js

import React, {Component} from 'react';

class List extends Component {

render() {

return (

<div>List~</div>

)

}

}

export default List;

\src\pages\home\components\Recommend.js

import React, {Component} from 'react';

class Recommend extends Component {

render() {

return (

<div>Recommend~</div>

)

}

}

export default Recommend;

\src\pages\home\components\Topic.js

import React, {Component} from 'react';

class Topic extends Component {

render() {

return (

<div>Topic~</div>

)

}

}

export default Topic;

\src\pages\home\components\Writer.js

import React, {Component} from 'react';

class Writer extends Component {

render() {

return (

<div>Writer~</div>

)

}

}

export default Writer;

\src\pages\home\index.js

import React, {Component} from 'react';

import Topic from './components/Topic';

import List from './components/List';

import Recommend from './components/Recommend';

import Writer from './components/Writer';

import {

HomeWrapper,

HomeLeft,

HomeRight

} from './style';

class Home extends Component {

render() {

return (

<HomeWrapper>

<HomeLeft>

<img className='banner-img' src="//upload.jianshu.io/admin\_banners/web\_images/4675/569882aa7e239c755c754c4d170a859eca26f752.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/1250/h/540" />

<Topic />

<List />

</HomeLeft>

<HomeRight>

<Recommend />

<Writer />

</HomeRight>

</HomeWrapper>

)

}

}

export default Home;

\src\pages\home\style.js

import styled from 'styled-components';

export const HomeWrapper = styled.div`

width: 960px;

margin: 0 auto;

height: 300px;

`;

export const HomeLeft = styled.div`

float: left;

margin-left: 15px;

padding-top: 30px;

width: 625px;

.banner-img {

width: 625px;

height: 270px;

}

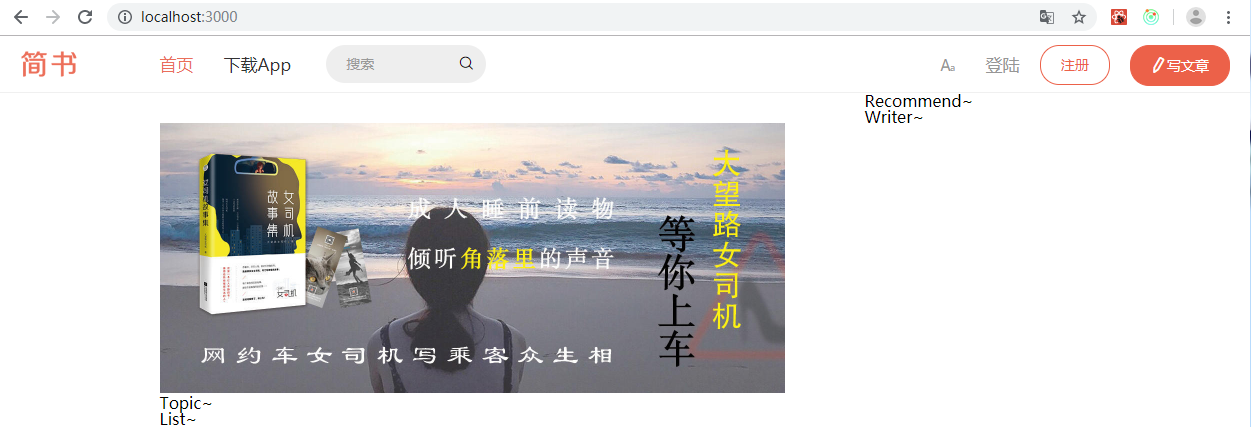
`;

export const HomeRight = styled.div`

width: 240px;

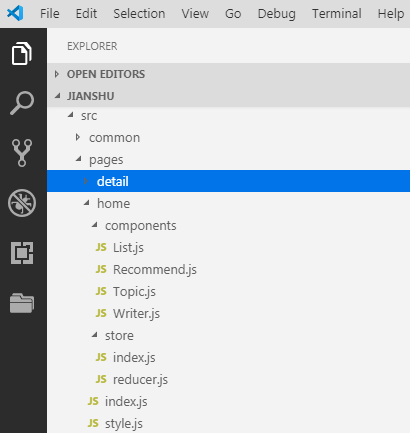
float: right;

`;



**首页专题区域布局及reducer的设计**

1. 项目结构



1. 源码改动

\src\pages\home\components\Topic.js

import React, {Component} from 'react';

import { connect } from 'react-redux';

import { TopicWrapper, TopicItem } from '../style';

class Topic extends Component {

render() {

const { list } = this.props;

return (

<TopicWrapper>

{

list.map((item) => (

<TopicItem key={item.get('id')}>

<img

className='topic-pic'

src={item.get('imgUrl')}

>

</img>

{item.get('title')}

</TopicItem>

))

}

</TopicWrapper>

)

}

}

const mapState = (state) => ({

list: state.get('home').get('topicList')

});

export default connect(mapState, null)(Topic);

\src\pages\home\store\index.js

import reducer from './reducer';

export { reducer };

\src\pages\home\store\reducer.js

import { fromJS } from 'immutable';

const defaultState = fromJS({

topicList: [{

id: 1,

title: '社会热点',

imgUrl: '//cdn2.jianshu.io/assets/web/zggsrz-5695587dccf490ca3e651f4228f7479e.png'

},{

id: 2,

title: '手绘',

imgUrl: '//cdn2.jianshu.io/assets/web/fanzha-10518f0f6b33635180b190975ae68ca6.jpg'

}]

});

export default (state = defaultState, action) => {

switch(action.type) {

default:

return state;

}

}

\src\pages\home\style.js

export const TopicWrapper = styled.div`

overflow: hidden;

padding: 20px 0 10px 0;

margin-left: -18px;

`;

export const TopicItem = styled.div`

float: left;

height: 32px;

line-height: 32px;

margin-left: 18px;

margin-bottom: 18px;

padding-right: 10px;

background: #f7f7f7f;

font-size: 14px;

color: #000;

border: 1px solid #dcdcdc;

border-radius: 4px;

.topic-pic {

display: block;

float: left;

width: 32px;

height: 32px;

margin-right: 10px;

}

`;

\src\store\reducer.js

import { combineReducers } from 'redux-immutable';

import { reducer as headerReducer } from '../common/header/store';

import { reducer as homeReducer } from '../pages/home/store';

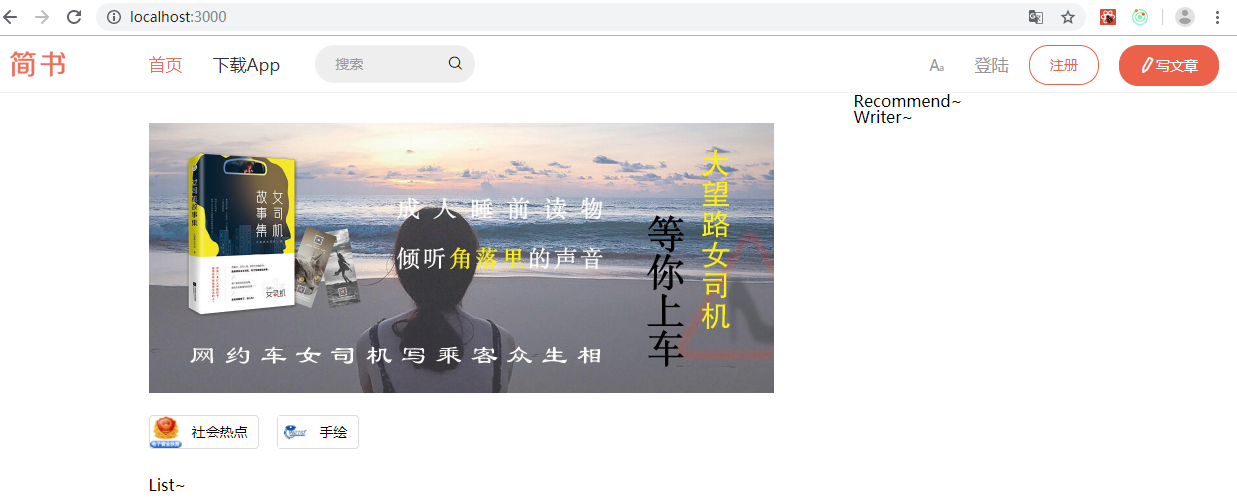
const reducer = combineReducers({

header: headerReducer,

home: homeReducer

});

export default reducer;



**首页文章列表制作**

1. 源码改动

\src\pages\home\store\reducer.js

import { fromJS } from 'immutable';

const defaultState = fromJS({

topicList: [{

id: 1,

title: '社会热点',

imgUrl: '//cdn2.jianshu.io/assets/web/zggsrz-5695587dccf490ca3e651f4228f7479e.png'

},{

id: 2,

title: '手绘',

imgUrl: '//cdn2.jianshu.io/assets/web/fanzha-10518f0f6b33635180b190975ae68ca6.jpg'

}],

articleList: [{

id: 1,

title: '如何不工作，也能有源源不断的收入',

desc: '尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。',

imgUrl: '//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240'

},{

id: 2,

title: '如何不工作，也能有源源不断的收入',

desc: '尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。',

imgUrl: '//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240'

},{

id: 3,

title: '如何不工作，也能有源源不断的收入',

desc: '尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。',

imgUrl: '//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240'

},{

id: 4,

title: '如何不工作，也能有源源不断的收入',

desc: '尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。',

imgUrl: '//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240'

}]

});

export default (state = defaultState, action) => {

switch(action.type) {

default:

return state;

}

}

\src\pages\home\components\List.js

import React, {Component} from 'react';

import { ListItem, ListInfo } from '../style';

import { connect } from 'react-redux';

class List extends Component {

render() {

const { list } = this.props;

return (

<div>

{

list.map((item) => {

return (

<ListItem key={item.get('id')}>

<img className='pic' src={item.get('imgUrl')} alt=''></img>

<ListInfo>

<h3 className='title'>{item.get('title')}</h3>

<p className='desc'>{item.get('desc')}</p>

</ListInfo>

</ListItem>

)

})

}

</div>

)

}

}

const mapState = (state) => ({

list: state.getIn(['home', 'articleList'])

})

export default connect(mapState, null)(List);

\src\pages\home\style.js

export const ListItem = styled.div`

overflow: hidden;

padding: 20px 0;

border-bottom: 1px solid #dcdcdc;

.pic {

display: block;

width: 125px;

height: 100px;

float: right;

border-radius: 10px;

}

`;

export const ListInfo = styled.div`

width: 500px;

float: left;

.title {

line-height: 27px;

font-size: 18px;

font-weight: bold;

color: #333;

}

.desc {

line-height: 24px;

font-size: 13px;

color: #999;

}

`;



**首页推荐部分代码编写**

1. 源码改动

\src\pages\home\components\Recommend.js

import React, {Component} from 'react';

import { RecommendWrapper, RecommendItem } from '../style';

import { connect } from 'react-redux';

class Recommend extends Component {

render() {

return (

<RecommendWrapper>

{

this.props.list.map((item) => {

return <RecommendItem imgUrl={item.get('imgUrl')} key={item.get('id')}/>

})

}

</RecommendWrapper>

)

}

}

const mapState = (state) => ({

list: state.getIn(['home', 'recommendList'])

})

export default connect(mapState, null)(Recommend);

\src\pages\home\components\Writer.js

import React, {Component} from 'react';

import { WriterWrapper } from '../style';

class Writer extends Component {

render() {

return (

<WriterWrapper>HomeWork</WriterWrapper>

)

}

}

export default Writer;

\src\pages\home\store\Reducer.js

recommendList: [{

id: 1,

imgUrl: "http://cdn2.jianshu.io/assets/web/banner-s-club-aa8bdf19f8cf729a759da42e4a96f366.png"

},{

id: 2,

imgUrl: "http://cdn2.jianshu.io/assets/web/banner-s-7-1a0222c91694a1f38e610be4bf9669be.png"

}]

\src\pages\home\style.js

export const RecommendWrapper = styled.div`

margin: 30px 0;

width: 280px;

`;

export const RecommendItem = styled.div`

width: 280px;

height: 50px;

background: url(${(props) => props.imgUrl});

background-size: contain;

`;

export const WriterWrapper = styled.div`

width: 278px;

border: 1px solid #dcdcdc;

border-radius: 3px;

line-height: 300px;

text-align: center;

`;

**首页异步数据获取**

1. 源码改动

\public\api\home.json

{

"success": true,

"data": {

"topicList": [{

"id": 1,

"title": "社会热点",

"imgUrl": "//cdn2.jianshu.io/assets/web/zggsrz-5695587dccf490ca3e651f4228f7479e.png"

},{

"id": 2,

"title": "手绘",

"imgUrl": "//cdn2.jianshu.io/assets/web/fanzha-10518f0f6b33635180b190975ae68ca6.jpg"

}],

"articleList": [{

"id": 1,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

},{

"id": 2,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

},{

"id": 3,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

},{

"id": 4,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

}],

"recommendList": [{

"id": 1,

"imgUrl": "http://cdn2.jianshu.io/assets/web/banner-s-club-aa8bdf19f8cf729a759da42e4a96f366.png"

},{

"id": 2,

"imgUrl": "http://cdn2.jianshu.io/assets/web/banner-s-7-1a0222c91694a1f38e610be4bf9669be.png"

}]

}

}

\src\pages\home\store\reducer.js

import { fromJS } from 'immutable';

const defaultState = fromJS({

topicList: [],

articleList: [],

recommendList: []

});

export default (state = defaultState, action) => {

switch(action.type) {

case 'change\_home\_data':

return state.merge({

topicList: fromJS(action.topicList),

articleList: fromJS(action.articleList),

recommendList: fromJS(action.recommendList)

});

default:

return state;

}

}

\src\pages\home\index.js

import React, {Component} from 'react';

import { connect } from 'react-redux';

import Topic from './components/Topic';

import List from './components/List';

import Recommend from './components/Recommend';

import Writer from './components/Writer';

import axios from 'axios';

import {

HomeWrapper,

HomeLeft,

HomeRight

} from './style';

class Home extends Component {

render() {

return (

<HomeWrapper>

<HomeLeft>

<img className='banner-img' src="//upload.jianshu.io/admin\_banners/web\_images/4675/569882aa7e239c755c754c4d170a859eca26f752.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/1250/h/540" alt=''/>

<Topic />

<List />

</HomeLeft>

<HomeRight>

<Recommend />

<Writer />

</HomeRight>

</HomeWrapper>

)

}

componentDidMount() {

axios.get('/api/home.json').then((res) => {

const result = res.data.data;

const action = {

type: 'change\_home\_data',

topicList: result.topicList,

articleList: result.articleList,

recommendList: result.recommendList

};

this.props.changeHomeData(action);

});

}

}

const mapDispatch = (dispatch) => ({

changeHomeData(action) {

dispatch(action);

}

});

export default connect(null, mapDispatch)(Home);

**异步操作代码拆分优化**

1. 源码修改

\src\pages\home\store\actionCreators.js

import axios from 'axios';

import \* as constants from './constants';

const changeHomeData = (result) => ({

type: constants.CHANGE\_HOME\_DATA,

topicList: result.topicList,

articleList: result.articleList,

recommendList: result.recommendList

});

export const getHomeInfo = () => {

return (dispatch) => {

axios.get('/api/home.json').then((res) => {

const result = res.data.data;

dispatch(changeHomeData(result));

});

}

}

\src\pages\home\store\constants.js

export const CHANGE\_HOME\_DATA = 'home/CHANGE\_HOME\_DATA';

\src\pages\home\store\index.js

import reducer from './reducer';

import \* as actionCreators from './actionCreators';

import \* as constants from './constants';

export { reducer, actionCreators, constants };

\src\pages\home\store\reducer.js

import { fromJS } from 'immutable';

import \* as constants from './constants';

const defaultState = fromJS({

topicList: [],

articleList: [],

recommendList: []

});

export default (state = defaultState, action) => {

switch(action.type) {

case constants.CHANGE\_HOME\_DATA:

return state.merge({

topicList: fromJS(action.topicList),

articleList: fromJS(action.articleList),

recommendList: fromJS(action.recommendList)

});

default:

return state;

}

}

\src\pages\home\index.js

import React, {Component} from 'react';

import { connect } from 'react-redux';

import Topic from './components/Topic';

import List from './components/List';

import Recommend from './components/Recommend';

import Writer from './components/Writer';

import { actionCreators } from './store';

import {

HomeWrapper,

HomeLeft,

HomeRight

} from './style';

class Home extends Component {

render() {

return (

<HomeWrapper>

<HomeLeft>

<img className='banner-img' src="//upload.jianshu.io/admin\_banners/web\_images/4675/569882aa7e239c755c754c4d170a859eca26f752.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/1250/h/540" alt=''/>

<Topic />

<List />

</HomeLeft>

<HomeRight>

<Recommend />

<Writer />

</HomeRight>

</HomeWrapper>

)

}

componentDidMount() {

this.props.changeHomeData();

}

}

const mapDispatch = (dispatch) => ({

changeHomeData() {

const action = actionCreators.getHomeInfo();

dispatch(action);

}

});

export default connect(null, mapDispatch)(Home);

**实现加载更多功能**

1. 源码改动

\public\api\homeList.json

{

"success": true,

"data": [

{

"id": 5,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

},

{

"id": 6,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

},

{

"id": 7,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

},

{

"id": 8,

"title": "如何不工作，也能有源源不断的收入",

"desc": "尽管现在你的本职工作收入不错，可是你有没有想过，人生漫长，未来不可知，如果有一天，因为某种原因，你不能再在职场上打拼了，你还拥有赚钱的能力吗？你还能保证岁月静好的模样吗？ 如何不工作，也能有源源不断的收入？这是个问题。",

"imgUrl": "//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/360/h/240"

}

]

}

\src\pages\home\components\List.js

import React, {Component} from 'react';

import { ListItem, ListInfo, LoadMore } from '../style';

import { connect } from 'react-redux';

import { actionCreators } from '../store';

class List extends Component {

render() {

const { list, getMoreList, page } = this.props;

return (

<div>

{

list.map((item, index) => {

return (

<ListItem key={index}>

<img className='pic' src={item.get('imgUrl')} alt=''></img>

<ListInfo>

<h3 className='title'>{item.get('title')}</h3>

<p className='desc'>{item.get('desc')}</p>

</ListInfo>

</ListItem>

)

})

}

<LoadMore onClick={() => getMoreList(page)}>更多文字</LoadMore>

</div>

)

}

}

const mapState = (state) => ({

list: state.getIn(['home', 'articleList']),

page: state.getIn(['home', 'articlePage'])

});

const mapDispatch = (dispatch) => ({

getMoreList(page) {

dispatch(actionCreators.getMoreList(page));

}

});

export default connect(mapState, mapDispatch)(List);

\src\pages\home\store\actionCreator.js

import axios from 'axios';

import \* as constants from './constants';

import { fromJS } from 'immutable';

const changeHomeData = (result) => ({

type: constants.CHANGE\_HOME\_DATA,

topicList: result.topicList,

articleList: result.articleList,

recommendList: result.recommendList

});

const addHomeList = (list, nextPage) => ({

type: constants.ADD\_ARTICLE\_LIST,

list: fromJS(list),

nextPage: nextPage

});

export const getHomeInfo = () => {

return (dispatch) => {

axios.get('/api/home.json').then((res) => {

const result = res.data.data;

dispatch(changeHomeData(result));

});

}

}

export const getMoreList = (page) => {

return (dispatch) => {

axios.get('/api/homeList.json?page=' + page).then((res) => {

const result = res.data.data;

dispatch(addHomeList(result, page + 1));

});

}

}

\src\pages\home\store\constants.js

export const CHANGE\_HOME\_DATA = 'home/CHANGE\_HOME\_DATA';

export const ADD\_ARTICLE\_LIST = 'home/ADD\_ARTICLE\_LIST';

\src\pages\home\store\reducer.js

import { fromJS } from 'immutable';

import \* as constants from './constants';

const defaultState = fromJS({

topicList: [],

articleList: [],

recommendList: [],

articlePage: 1

});

export default (state = defaultState, action) => {

switch(action.type) {

case constants.CHANGE\_HOME\_DATA:

return state.merge({

topicList: fromJS(action.topicList),

articleList: fromJS(action.articleList),

recommendList: fromJS(action.recommendList)

});

case constants.ADD\_ARTICLE\_LIST:

return state.merge({

'articleList': state.get('articleList').concat(action.list),

'articlePage': action.nextPage

})

default:

return state;

}

}

\src\pages\home\style.js

export const LoadMore = styled.div`

width: 100%;

height: 40px;

line-height: 40px;

margin: 30px 0;

background: #a5a5a5;

text-align: center;

border-radius: 20px;

color: #fff;

course: pointer;

`;



**返回顶部功能实现**

1. 源码改动

\src\pages\home\store\actionCreators.js

import axios from 'axios';

import \* as constants from './constants';

import { fromJS } from 'immutable';

const changeHomeData = (result) => ({

type: constants.CHANGE\_HOME\_DATA,

topicList: result.topicList,

articleList: result.articleList,

recommendList: result.recommendList

});

const addHomeList = (list, nextPage) => ({

type: constants.ADD\_ARTICLE\_LIST,

list: fromJS(list),

nextPage: nextPage

});

export const getHomeInfo = () => {

return (dispatch) => {

axios.get('/api/home.json').then((res) => {

const result = res.data.data;

dispatch(changeHomeData(result));

});

}

}

export const getMoreList = (page) => {

return (dispatch) => {

axios.get('/api/homeList.json?page=' + page).then((res) => {

const result = res.data.data;

dispatch(addHomeList(result, page + 1));

});

}

}

export const toggleTopShow = (show) => ({

type: constants.TOGGLE\_SCROLL\_TOP,

show

})

\src\pages\home\store\constants.js

export const CHANGE\_HOME\_DATA = 'home/CHANGE\_HOME\_DATA';

export const ADD\_ARTICLE\_LIST = 'home/ADD\_ARTICLE\_LIST';

export const TOGGLE\_SCROLL\_TOP = 'home/TOGGLE\_SCROLL\_TOP';

\src\pages\home\store\reducer.js

import { fromJS } from 'immutable';

import \* as constants from './constants';

const defaultState = fromJS({

topicList: [],

articleList: [],

recommendList: [],

articlePage: 1,

showScroll: false

});

const changeHomeData = (state, action) => {

return state.merge({

topicList: fromJS(action.topicList),

articleList: fromJS(action.articleList),

recommendList: fromJS(action.recommendList)

});

};

const addArticleList = (state, action) => {

return state.merge({

'articleList': state.get('articleList').concat(action.list),

'articlePage': action.nextPage

});

};

export default (state = defaultState, action) => {

switch(action.type) {

case constants.CHANGE\_HOME\_DATA:

return changeHomeData(state, action);

case constants.ADD\_ARTICLE\_LIST:

return addArticleList(state, action);

case constants.TOGGLE\_SCROLL\_TOP:

return state.set('showScroll', action.show);

default:

return state;

}

}

\src\pages\home\index.js

import React, {Component} from 'react';

import { connect } from 'react-redux';

import Topic from './components/Topic';

import List from './components/List';

import Recommend from './components/Recommend';

import Writer from './components/Writer';

import { actionCreators } from './store';

import {

HomeWrapper,

HomeLeft,

HomeRight,

BackTop

} from './style';

class Home extends Component {

handleScrollTop() {

window.scrollTo(0, 0);

}

render() {

return (

<HomeWrapper>

<HomeLeft>

<img className='banner-img' src="//upload.jianshu.io/admin\_banners/web\_images/4675/569882aa7e239c755c754c4d170a859eca26f752.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/1250/h/540" alt=''/>

<Topic />

<List />

</HomeLeft>

<HomeRight>

<Recommend />

<Writer />

</HomeRight>

{this.props.showScroll ? <BackTop onClick={this.handleScrollTop}>顶部</BackTop> : null}

</HomeWrapper>

)

}

componentDidMount() {

this.props.changeHomeData();

this.bindEvents();

}

componentWillUnmount() {

window.removeEventListener('scroll', this.props.changeScrollTopShow);

}

bindEvents() {

window.addEventListener('scroll', this.props.changeScrollTopShow);

}

}

const mapState = (state) => ({

showScroll: state.getIn(['home', 'showScroll'])

})

const mapDispatch = (dispatch) => ({

changeHomeData() {

const action = actionCreators.getHomeInfo();

dispatch(action);

},

changeScrollTopShow(e) {

if(document.documentElement.scrollTop > 100) {

dispatch(actionCreators.toggleTopShow(true));

}else{

dispatch(actionCreators.toggleTopShow(false));

}

}

});

export default connect(mapState, mapDispatch)(Home);

\src\pages\home\style.js

export const BackTop = styled.div`

position: fixed;

right: 100px;

bottom: 100px

width: 60px;

height: 60px;

line-height: 60px;

text-align: center;

border: 1px solid #ccc;

font-size: 14px;

`;

**首页性能优化及路由跳转**

1. 使用PureComponent代替Component来提高性能。List.js，Topic.js仅在需要时自动刷新。
2. 源码修改

\src\pages\home\index.js

import React, {PureComponent} from 'react';

import { connect } from 'react-redux';

import Topic from './components/Topic';

import List from './components/List';

import Recommend from './components/Recommend';

import Writer from './components/Writer';

import { actionCreators } from './store';

import {

HomeWrapper,

HomeLeft,

HomeRight,

BackTop

} from './style';

class Home extends PureComponent {

handleScrollTop() {

window.scrollTo(0, 0);

}

render() {

return (

<HomeWrapper>

<HomeLeft>

<img className='banner-img' src="//upload.jianshu.io/admin\_banners/web\_images/4675/569882aa7e239c755c754c4d170a859eca26f752.jpg?imageMogr2/auto-orient/strip|imageView2/1/w/1250/h/540" alt=''/>

<Topic />

<List />

</HomeLeft>

<HomeRight>

<Recommend />

<Writer />

</HomeRight>

{this.props.showScroll ? <BackTop onClick={this.handleScrollTop}>顶部</BackTop> : null}

</HomeWrapper>

)

}

componentDidMount() {

this.props.changeHomeData();

this.bindEvents();

}

componentWillUnmount() {

window.removeEventListener('scroll', this.props.changeScrollTopShow);

}

bindEvents() {

window.addEventListener('scroll', this.props.changeScrollTopShow);

}

}

const mapState = (state) => ({

showScroll: state.getIn(['home', 'showScroll'])

})

const mapDispatch = (dispatch) => ({

changeHomeData() {

const action = actionCreators.getHomeInfo();

dispatch(action);

},

changeScrollTopShow(e) {

if(document.documentElement.scrollTop > 100) {

dispatch(actionCreators.toggleTopShow(true));

}else{

dispatch(actionCreators.toggleTopShow(false));

}

}

});

export default connect(mapState, mapDispatch)(Home);

\src\pages\home\style.js

export const Logo = styled.div`

position: absolute;

top: 0;

left: 0;

display: block;

width: 100px;

height: 56px;

background: url(${logoPic});

background-size: contain;

`;

\src\pages\home\components\List.js

import React, {PureComponent} from 'react';

import { ListItem, ListInfo, LoadMore } from '../style';

import { connect } from 'react-redux';

import { actionCreators } from '../store';

import { Link } from 'react-router-dom';

class List extends PureComponent {

render() {

const { list, getMoreList, page } = this.props;

return (

<div>

{

list.map((item, index) => {

return (

<Link key={index} to='/detail'>

<ListItem>

<img className='pic' src={item.get('imgUrl')} alt=''></img>

<ListInfo>

<h3 className='title'>{item.get('title')}</h3>

<p className='desc'>{item.get('desc')}</p>

</ListInfo>

</ListItem>

</Link>

)

})

}

<LoadMore onClick={() => getMoreList(page)}>更多文字</LoadMore>

</div>

)

}

}

const mapState = (state) => ({

list: state.getIn(['home', 'articleList']),

page: state.getIn(['home', 'articlePage'])

});

const mapDispatch = (dispatch) => ({

getMoreList(page) {

dispatch(actionCreators.getMoreList(page));

}

});

export default connect(mapState, mapDispatch)(List);

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import { BrowserRouter, Route } from 'react-router-dom';

import Header from './common/header';

import Home from './pages/home';

import Detail from './pages/detail';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<BrowserRouter>

<div>

<Header />

<Route path='/' exact component={Home}></Route>

<Route path='/detail' exact component={Detail}></Route>

</div>

</BrowserRouter>

</Fragment>

</Provider>

);

}

export default App;

**详情页面布局**

1. 源码改动

\src\pages\detail\index.js

import React, {Component} from 'react';

import { DetailWrapper, Header, Content } from './style';

class Detail extends Component {

render() {

return (

<DetailWrapper>

<Header>如何不工作，也能有源源不断的收入</Header>

<Content>

<img src='//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg'></img>

<p><b>一、增加不工作的收入。</b> 比如说，银行存款的利息，买基金买保险的收益，房产每个月的房租，投资企业获得分红，这些都是不工作时的收入。 炒股需要投入时间和精力去分析和跟进，所以不算严格意义上的被动收入，除非是长期持有收益稳定、小风险的股种。 扪心自问，这些你有吗？ 以上几种被动收入的资本，钱，属于保守意义上的资本。在这个知识为王的时代，资本还可以有很多种呈现方式。 比如，靠知识产权赚被动收入。你写了一本书，每次再版都能获得新版税，而不需要再有多余投入。电影票房，开发软件，知识付费，网络课程，也都是同理。在这里，书本、影片、软件、内容……本身就成为了资本，让你不需劳动，就能获得源源不断的收入。 当然，积累资本的过程，也是不断投入的过程。在坐拥不断生产被动收入的资本之前，在你从付出时间和精力中解放之前，首先要投入大量的时间和精力，去积累知识、实力，和财富。 目前，我主要通过投资理财实现，投入债基和混合基金，坚持长期定投指数基金，以此获得收益。</p>

<p><b>二、改变收入结构。</b> 最近，我对这句话无比赞同。如果一个人到了三十岁还是只有工资收入，那么他的职业规划是失败的。 你的死工资只能满足日常温饱，并不能让你走向财富自由之路。怎么办？ 俗话说工资不够，兼职来凑。在做好主业的基础上，以此为圆心进行辐射。多去尝试副业，能接触多少开源的方式就接触多少，然后重点筛选出跟个人能力兴趣相匹配的开源模式。 比如身边有个朋友，年轻时考取 一级建造师证件和国家注册监理工程师证，挂靠一家监理公司，一年挂靠收入6万元。当真是一劳永逸旱涝保收啊。羡慕嫉妒恨呀！ 理财的核心，在于结构，而不是数量。 不仅仅是金钱上的宽裕，更多的是价值的认同和信心的提升，让我逃离了鸡毛蒜皮的日常，找到了人生的新篇章。那些达人、牛人能做到的的事情，我踮起脚尖奋力奔跑，也可以做到，实现草根逆袭的华丽转变。那种感觉是无与伦比的爽，那种被人认同的存在感和价值感，是任何物质也替代不了的满足感。</p>

<p><b>三、创造持续收入。</b>我个人的理解是，始终保持学习力和创新精神，勇敢的去尝试、去突破、去改变。凡是不可持续的，都是不值得羡慕的。 我认为的投资分两种，一种是资产投资，一种是个人投资。 我感觉在所有的投资品中，自我投资是最有前途、升值空间最大的一项投资。投资自己包括： ①提高技能。本职工作业务提高，写作、烘焙、摄影、绘画等等。 ②提升个人人脉，提升个人社交圈，扩展人脉资源。 ③提升精神生活。 这部分的钱是不能省的。 要知道光靠省是省不出一个未来的。 投资有风险，躺在床上很安全。你选你一个？ 20岁的我们不竭尽全力，40岁的我们如何淡泊名利？要知道生活的苟且是自费的，诗和远方则是需要门票的。 调整思维永远比付诸行动更重要。</p>

</Content>

</DetailWrapper>

)

}

}

export default Detail;

\src\pages\detail\style.js

import styled from 'styled-components';

export const DetailWrapper = styled.div`

overflow: hidden;

width: 620px;

margin: 0 auto;

padding-bottom: 100px;

`;

export const Header = styled.div`

margin: 50px 0 20px 0;

line-height: 44px;

font-size: 34px;

color: #333;

font-weight: bold;

`;

export const Content = styled.div`

color: #2f2f2f;

img {

width: 100%;

}

p {

margin: 25px 0;

font-size: 16px;

line-height: 30px;

}

b {

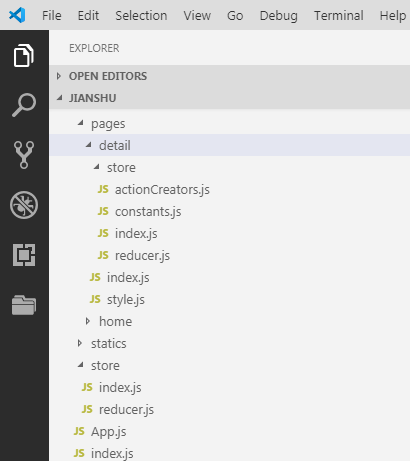
font-weight: bold;

}

`;

**使用redux管理详情页面数据**

1. 项目目录结构



1. 源码改动

\src\pages\detail\store\index.js

import reducer from './reducer';

import \* as actionCreators from './actionCreators';

import \* as constants from './constants';

export { reducer, actionCreators, constants };

\src\pages\detail\store\reducer.js

import { fromJS } from 'immutable';

import \* as constants from './constants';

const defaultState = fromJS({

title: '如何不工作，也能有源源不断的收入',

content: '<img src="//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg"></img><p><b>一、增加不工作的收入。</b> 比如说，银行存款的利息，买基金买保险的收益，房产每个月的房租，投资企业获得分红，这些都是不工作时的收入。 炒股需要投入时间和精力去分析和跟进，所以不算严格意义上的被动收入，除非是长期持有收益稳定、小风险的股种。 扪心自问，这些你有吗？ 以上几种被动收入的资本，钱，属于保守意义上的资本。在这个知识为王的时代，资本还可以有很多种呈现方式。 比如，靠知识产权赚被动收入。你写了一本书，每次再版都能获得新版税，而不需要再有多余投入。电影票房，开发软件，知识付费，网络课程，也都是同理。在这里，书本、影片、软件、内容……本身就成为了资本，让你不需劳动，就能获得源源不断的收入。 当然，积累资本的过程，也是不断投入的过程。在坐拥不断生产被动收入的资本之前，在你从付出时间和精力中解放之前，首先要投入大量的时间和精力，去积累知识、实力，和财富。 目前，我主要通过投资理财实现，投入债基和混合基金，坚持长期定投指数基金，以此获得收益。</p><p><b>二、改变收入结构。</b> 最近，我对这句话无比赞同。如果一个人到了三十岁还是只有工资收入，那么他的职业规划是失败的。 你的死工资只能满足日常温饱，并不能让你走向财富自由之路。怎么办？ 俗话说工资不够，兼职来凑。在做好主业的基础上，以此为圆心进行辐射。多去尝试副业，能接触多少开源的方式就接触多少，然后重点筛选出跟个人能力兴趣相匹配的开源模式。 比如身边有个朋友，年轻时考取 一级建造师证件和国家注册监理工程师证，挂靠一家监理公司，一年挂靠收入6万元。当真是一劳永逸旱涝保收啊。羡慕嫉妒恨呀！ 理财的核心，在于结构，而不是数量。 不仅仅是金钱上的宽裕，更多的是价值的认同和信心的提升，让我逃离了鸡毛蒜皮的日常，找到了人生的新篇章。那些达人、牛人能做到的的事情，我踮起脚尖奋力奔跑，也可以做到，实现草根逆袭的华丽转变。那种感觉是无与伦比的爽，那种被人认同的存在感和价值感，是任何物质也替代不了的满足感。</p><p><b>三、创造持续收入。</b>我个人的理解是，始终保持学习力和创新精神，勇敢的去尝试、去突破、去改变。凡是不可持续的，都是不值得羡慕的。 我认为的投资分两种，一种是资产投资，一种是个人投资。 我感觉在所有的投资品中，自我投资是最有前途、升值空间最大的一项投资。投资自己包括： ①提高技能。本职工作业务提高，写作、烘焙、摄影、绘画等等。 ②提升个人人脉，提升个人社交圈，扩展人脉资源。 ③提升精神生活。 这部分的钱是不能省的。 要知道光靠省是省不出一个未来的。 投资有风险，躺在床上很安全。你选你一个？ 20岁的我们不竭尽全力，40岁的我们如何淡泊名利？要知道生活的苟且是自费的，诗和远方则是需要门票的。 调整思维永远比付诸行动更重要。</p>'

});

export default (state = defaultState, action) => {

switch(action.type) {

default:

return state;

}

}

\src\pages\detail\index.js

import React, {Component} from 'react';

import { connect } from 'react-redux';

import { DetailWrapper, Header, Content } from './style';

class Detail extends Component {

render() {

return (

<DetailWrapper>

<Header>{this.props.title}</Header>

<Content

dangerouslySetInnerHTML={{\_\_html: this.props.content}}

/>

</DetailWrapper>

)

}

}

const mapState = (state) => ({

title: state.getIn(['detail', 'title']),

content: state.getIn(['detail', 'content'])

})

export default connect(mapState, null)(Detail);

\src\store\reducer.js

import { combineReducers } from 'redux-immutable';

import { reducer as headerReducer } from '../common/header/store';

import { reducer as homeReducer } from '../pages/home/store';

import { reducer as detailReducer } from '../pages/detail/store';

const reducer = combineReducers({

header: headerReducer,

home: homeReducer,

detail: detailReducer

});

export default reducer;



**异步获取数据**

1. 源码修改

\public\api\detail.json

{

"sucess": true,

"data" : {

"title": "如何不工作，也能有源源不断的收入",

"content": "<img src='//upload-images.jianshu.io/upload\_images/3994777-e3e72e82dba8282e.jpg'></img><p><b>一、增加不工作的收入。</b> 比如说，银行存款的利息，买基金买保险的收益，房产每个月的房租，投资企业获得分红，这些都是不工作时的收入。 炒股需要投入时间和精力去分析和跟进，所以不算严格意义上的被动收入，除非是长期持有收益稳定、小风险的股种。 扪心自问，这些你有吗？ 以上几种被动收入的资本，钱，属于保守意义上的资本。在这个知识为王的时代，资本还可以有很多种呈现方式。 比如，靠知识产权赚被动收入。你写了一本书，每次再版都能获得新版税，而不需要再有多余投入。电影票房，开发软件，知识付费，网络课程，也都是同理。在这里，书本、影片、软件、内容……本身就成为了资本，让你不需劳动，就能获得源源不断的收入。 当然，积累资本的过程，也是不断投入的过程。在坐拥不断生产被动收入的资本之前，在你从付出时间和精力中解放之前，首先要投入大量的时间和精力，去积累知识、实力，和财富。 目前，我主要通过投资理财实现，投入债基和混合基金，坚持长期定投指数基金，以此获得收益。</p><p><b>二、改变收入结构。</b> 最近，我对这句话无比赞同。如果一个人到了三十岁还是只有工资收入，那么他的职业规划是失败的。 你的死工资只能满足日常温饱，并不能让你走向财富自由之路。怎么办？ 俗话说工资不够，兼职来凑。在做好主业的基础上，以此为圆心进行辐射。多去尝试副业，能接触多少开源的方式就接触多少，然后重点筛选出跟个人能力兴趣相匹配的开源模式。 比如身边有个朋友，年轻时考取 一级建造师证件和国家注册监理工程师证，挂靠一家监理公司，一年挂靠收入6万元。当真是一劳永逸旱涝保收啊。羡慕嫉妒恨呀！ 理财的核心，在于结构，而不是数量。 不仅仅是金钱上的宽裕，更多的是价值的认同和信心的提升，让我逃离了鸡毛蒜皮的日常，找到了人生的新篇章。那些达人、牛人能做到的的事情，我踮起脚尖奋力奔跑，也可以做到，实现草根逆袭的华丽转变。那种感觉是无与伦比的爽，那种被人认同的存在感和价值感，是任何物质也替代不了的满足感。</p><p><b>三、创造持续收入。</b>我个人的理解是，始终保持学习力和创新精神，勇敢的去尝试、去突破、去改变。凡是不可持续的，都是不值得羡慕的。 我认为的投资分两种，一种是资产投资，一种是个人投资。 我感觉在所有的投资品中，自我投资是最有前途、升值空间最大的一项投资。投资自己包括： ①提高技能。本职工作业务提高，写作、烘焙、摄影、绘画等等。 ②提升个人人脉，提升个人社交圈，扩展人脉资源。 ③提升精神生活。 这部分的钱是不能省的。 要知道光靠省是省不出一个未来的。 投资有风险，躺在床上很安全。你选你一个？ 20岁的我们不竭尽全力，40岁的我们如何淡泊名利？要知道生活的苟且是自费的，诗和远方则是需要门票的。 调整思维永远比付诸行动更重要。</p>"

}

}

\src\pages\detail\store\actionCreators.js

import axios from 'axios';

import \* as constants from './constants';

const changeDetail = (title, content) => ({

type: constants.CHANGE\_DETAIL,

title,

content

});

export const getDetail = () => {

return (dispatch) => {

axios.get('/api/detail.json').then((res) => {

const result = res.data.data;

dispatch(changeDetail(result.title, result.content));

})

}

}

\src\pages\detail\store\constants.js

export const CHANGE\_DETAIL = 'detail/CHANGE\_DETAIL';

\src\pages\detail\store\reducer.js

import { fromJS } from 'immutable';

import \* as constants from './constants';

const defaultState = fromJS({

title: '',

content: ''

});

export default (state = defaultState, action) => {

switch(action.type) {

case constants.CHANGE\_DETAIL:

return state.merge({

title: action.title,

content: action.content

});

default:

return state;

}

}

\src\pages\details\index.js

import React, {Component} from 'react';

import { connect } from 'react-redux';

import { DetailWrapper, Header, Content } from './style';

import { actionCreators } from './store'

class Detail extends Component {

render() {

return (

<DetailWrapper>

<Header>{this.props.title}</Header>

<Content

dangerouslySetInnerHTML={{\_\_html: this.props.content}}

/>

</DetailWrapper>

)

}

componentDidMount() {

this.props.getDetail();

}

}

const mapState = (state) => ({

title: state.getIn(['detail', 'title']),

content: state.getIn(['detail', 'content'])

});

const mapDispatch = (dispatch) => ({

getDetail() {

dispatch(actionCreators.getDetail());

}

});

export default connect(mapState, mapDispatch)(Detail);

**页面路由参数传递**

1. 源码修改

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import { BrowserRouter, Route } from 'react-router-dom';

import Header from './common/header';

import Home from './pages/home';

import Detail from './pages/detail';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<BrowserRouter>

<div>

<Header />

<Route path='/' exact component={Home}></Route>

<Route path='/detail/:id' exact component={Detail}></Route>

</div>

</BrowserRouter>

</Fragment>

</Provider>

);

}

export default App;

\src\pages\home\components\List.js

import React, {PureComponent} from 'react';

import { ListItem, ListInfo, LoadMore } from '../style';

import { connect } from 'react-redux';

import { actionCreators } from '../store';

import { Link } from 'react-router-dom';

class List extends PureComponent {

render() {

const { list, getMoreList, page } = this.props;

return (

<div>

{

list.map((item, index) => {

return (

<Link key={index} to={'/detail/' + item.get('id')}>

<ListItem>

<img className='pic' src={item.get('imgUrl')} alt=''></img>

<ListInfo>

<h3 className='title'>{item.get('title')}</h3>

<p className='desc'>{item.get('desc')}</p>

</ListInfo>

</ListItem>

</Link>

)

})

}

<LoadMore onClick={() => getMoreList(page)}>更多文字</LoadMore>

</div>

)

}

}

const mapState = (state) => ({

list: state.getIn(['home', 'articleList']),

page: state.getIn(['home', 'articlePage'])

});

const mapDispatch = (dispatch) => ({

getMoreList(page) {

dispatch(actionCreators.getMoreList(page));

}

});

export default connect(mapState, mapDispatch)(List);

\src\pages\detail\store\actionCreators.js

import axios from 'axios';

import \* as constants from './constants';

const changeDetail = (title, content) => ({

type: constants.CHANGE\_DETAIL,

title,

content

});

export const getDetail = (id) => {

return (dispatch) => {

axios.get('/api/detail.json?id=' + id).then((res) => {

const result = res.data.data;

dispatch(changeDetail(result.title, result.content));

})

}

}

\src\pages\detail\index.js

import React, {Component} from 'react';

import { connect } from 'react-redux';

import { DetailWrapper, Header, Content } from './style';

import { actionCreators } from './store'

class Detail extends Component {

render() {

return (

<DetailWrapper>

<Header>{this.props.title}</Header>

<Content

dangerouslySetInnerHTML={{\_\_html: this.props.content}}

/>

</DetailWrapper>

)

}

componentDidMount() {

this.props.getDetail(this.props.match.params.id);

}

}

const mapState = (state) => ({

title: state.getIn(['detail', 'title']),

content: state.getIn(['detail', 'content'])

});

const mapDispatch = (dispatch) => ({

getDetail(id) {

dispatch(actionCreators.getDetail(id));

}

});

export default connect(mapState, mapDispatch)(Detail);

**登陆页面布局**

1. 源码修改

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import { BrowserRouter, Route } from 'react-router-dom';

import Header from './common/header';

import Home from './pages/home';

import Detail from './pages/detail';

import Login from './pages/login';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<BrowserRouter>

<div>

<Header />

<Route path='/' exact component={Home}></Route>

<Route path='/login' exact component={Login}></Route>

<Route path='/detail/:id' exact component={Detail}></Route>

</div>

</BrowserRouter>

</Fragment>

</Provider>

);

}

export default App;

\src\pages\login\index.js

import React, { PureComponent } from 'react';

import { connect } from 'react-redux';

import { LoginWrapper, LoginBox, Input, Button } from './style';

class Login extends PureComponent {

render() {

return (

<LoginWrapper>

<LoginBox>

<Input placeholder='账号'/>

<Input placeholder='密码'/>

<Button>登陆</Button>

</LoginBox>

</LoginWrapper>

)

}

}

const mapState = (state) => ({

});

const mapDispatch = (dispatch) => ({

});

export default connect(mapState, mapDispatch)(Login);

\src\pages\login\style.sj

import styled from 'styled-components';

export const LoginWrapper = styled.div`

z-index: 0;

position: absolute;

left: 0;

right: 0;

bottom: 0;

top: 56px;

background: #eee;

`;

export const LoginBox = styled.div`

width: 400px;

height: 180px;

margin: 100px auto;

padding-top: 20px;

background: #fff;

box-shadow: 0 0 8px rgba(0,0,0,.1);

`;

export const Input = styled.input`

display: block;

width: 200px;

height: 20px;

line-height: 30px;

padding: 0 10px;

margin: 10px auto;

color: #777;

`;

export const Button = styled.div`

width: 220px;

height: 30px;

line-height: 30px;

color: #fff;

background: #3194d0;

border-radius: 15px;

margin: 10px auto;

text-align: center;

`;

**登陆功能实现**

1. 源码改动

\public\api\login.json

{

"success": true,

"data": true

}

\src\pages\login\store\actionCreators.js

import axios from 'axios';

import \* as constants from './constants';

const changeLogin = () => ({

type: constants.CHANGE\_LOGIN,

value: true

});

export const logout = () => ({

type: constants.LOGOUT,

value: false

});

export const login = (account, password) => {

return (dispatch) => {

axios.get('/api/login.json?account=' + account + '&password=' + password).then((res) => {

const result = res.data.data;

if(result){

dispatch(changeLogin());

}else{

alert('登陆失败');

}

})

}

};

\src\pages\login\store\constant.js

export const CHANGE\_LOGIN = 'login/CHANGE\_LOGIN';

export const LOGOUT = 'login/LOGOUT';

\src\pages\login\store\reducer.js

import { fromJS } from 'immutable';

import \* as constants from './constants';

const defaultState = fromJS({

login: false

});

export default (state = defaultState, action) => {

switch(action.type) {

case constants.CHANGE\_LOGIN:

return state.set('login', action.value);

case constants.LOGOUT:

return state.set('login', action.value);

default:

return state;

}

}

\src\pages\login\index.js

import React, { PureComponent } from 'react';

import { Redirect } from 'react-router-dom';

import { connect } from 'react-redux';

import { LoginWrapper, LoginBox, Input, Button } from './style';

import { actionCreators } from './store';

class Login extends PureComponent {

render() {

const { loginStatus } = this.props;

if(!loginStatus) {

return (

<LoginWrapper>

<LoginBox>

<Input placeholder='账号' ref={(input) => {this.account = input}}/>

<Input placeholder='密码' type='password' ref={(input) => {this.password = input}}/>

<Button onClick={() => this.props.login(this.account, this.password)}>登陆</Button>

</LoginBox>

</LoginWrapper>

)

}else{

return <Redirect to='/'/>

}

}

}

const mapState = (state) => ({

loginStatus: state.getIn(['login', 'login'])

});

const mapDispatch = (dispatch) => ({

login(accountElem, passwordElem){

dispatch(actionCreators.login(accountElem.value, passwordElem.value));

}

});

export default connect(mapState, mapDispatch)(Login);

\src\store\reducer.js

import { combineReducers } from 'redux-immutable';

import { reducer as headerReducer } from '../common/header/store';

import { reducer as homeReducer } from '../pages/home/store';

import { reducer as detailReducer } from '../pages/detail/store';

import { reducer as loginReducer } from '../pages/login/store';

const reducer = combineReducers({

header: headerReducer,

home: homeReducer,

detail: detailReducer,

login: loginReducer

});

export default reducer;

\src\common\hearder\index.js

import React, { Component } from 'react';

import { connect } from 'react-redux';

import { Link } from 'react-router-dom';

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

import { actionCreators } from './store';

import { actionCreators as loginActionCreators } from '../../pages/login/store';

import {

HeaderWrapper,

Logo,

Nav,

NavItem,

SearchWrapper,

NavSearch,

SearchInfo,

SearchInfoTitle,

SearchInfoSwitch,

SearchInfoList,

SearchInfoItem,

Addition,

Button

} from './style';

class Header extends Component {

getListArea() {

const { focused, list, page, totalPage, mouseIn, handleMouseEnter, handleMouseLeave, handleChangePage } = this.props;

const newList = list.toJS();

const pageList = [];

if(newList.length){

for (let i = (page - 1) \* 10; i < page \* 10; i++) {

pageList.push(

<SearchInfoItem key={newList[i]}>{newList[i]}</SearchInfoItem>

)

}

}

if(focused || mouseIn) {

return (

<SearchInfo

onMouseEnter={handleMouseEnter}

onMouseLeave={handleMouseLeave}

>

<SearchInfoTitle>

热门搜索

<SearchInfoSwitch onClick={() => handleChangePage(page, totalPage, this.spinIcon)}>

<i ref={(icon) => {this.spinIcon = icon}} className="iconfont spin">&#xe851;</i>

换一批

</SearchInfoSwitch>

</SearchInfoTitle>

<SearchInfoList>

{pageList}

</SearchInfoList>

</SearchInfo>

)

}else {

return null;

}

}

render() {

const { focused, handleInputFocus, handleInputBlur, login, logout } = this.props;

return (

<HeaderWrapper>

<Link to='/'>

<Logo/>

</Link>

<Nav>

<NavItem className='left active'>首页</NavItem>

<NavItem className='left'>下载App</NavItem>

{

login ?

<NavItem onClick={logout} className='right'>退出</NavItem> :

<Link to='/login'><NavItem className='right'>登陆</NavItem></Link>

}

<NavItem className='right'>

<i className="iconfont">&#xe636;</i>

</NavItem>

<SearchWrapper>

<CSSTransition

in={focused}

timeout={200}

classNames="slide"

>

<NavSearch

className={focused ? 'focused' : ''}

onFocus={handleInputFocus}

onBlur={handleInputBlur}

>

</NavSearch>

</CSSTransition>

<i className={focused ? 'focused iconfont zoom' : 'iconfont zoom'}>

&#xe637;

</i>

{this.getListArea(focused)}

</SearchWrapper>

</Nav>

<Addition>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

<Button className='reg'>注册</Button>

</Addition>

</HeaderWrapper>

);

}

}

const mapStateToProps = (state) => {

return {

//或者focused: state.get('header').get('focused')

focused: state.getIn(['header', 'focused']),

list: state.getIn(['header', 'list']),

page: state.getIn(['header', 'page']),

totalPage: state.getIn(['header', 'totalPage']),

mouseIn: state.getIn(['header', 'mouseIn']),

login: state.getIn(['login', 'login'])

}

}

const mapDispathToProps = (dispatch) => {

return {

handleInputFocus() {

dispatch(actionCreators.getList());

dispatch(actionCreators.searchFocus());

},

handleInputBlur() {

dispatch(actionCreators.searchBlur());

},

handleMouseEnter() {

dispatch(actionCreators.mouseEnter());

},

handleMouseLeave() {

dispatch(actionCreators.mouseLeave());

},

handleChangePage(page, totalPage, spin) {

let originAngle = spin.style.transform.replace(/[^0-9]/ig, '');

if(originAngle) {

originAngle = parseInt(originAngle, 10);

}else{

originAngle = 0;

}

spin.style.transform = 'rotate(' + (originAngle + 360) + 'deg)';

if (page < totalPage) {

dispatch(actionCreators.changePage(page + 1));

}else{

dispatch(actionCreators.changePage(1));

}

},

logout() {

dispatch(loginActionCreators.logout());

}

}

}

export default connect(mapStateToProps, mapDispathToProps)(Header);

**登陆鉴权及代码优化**

1. 源码修改

\src\pages\write\index.js

import React, { PureComponent } from 'react';

import { Redirect } from 'react-router-dom';

import { connect } from 'react-redux';

class Write extends PureComponent {

render() {

const { loginStatus } = this.props;

if(loginStatus) {

return (

<div>写文章页面</div>

)

}else{

return <Redirect to='/login'/>

}

}

}

const mapState = (state) => ({

loginStatus: state.getIn(['login', 'login'])

});

export default connect(mapState, null)(Write);

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import { BrowserRouter, Route } from 'react-router-dom';

import Header from './common/header';

import Home from './pages/home';

import Detail from './pages/detail';

import Login from './pages/login';

import Write from './pages/write';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<BrowserRouter>

<div>

<Header />

<Route path='/' exact component={Home}></Route>

<Route path='/login' exact component={Login}></Route>

<Route path='/write' exact component={Write}></Route>

<Route path='/detail/:id' exact component={Detail}></Route>

</div>

</BrowserRouter>

</Fragment>

</Provider>

);

}

export default App;

\src\common\header\index.js

<Addition>

<Link to='/write'>

<Button className='writting'>

<i className="iconfont">&#xe615;</i>

写文章

</Button>

</Link>

<Button className='reg'>注册</Button>

</Addition>

**异步组件及withRouter路由方法的使用**

1. 添加react-loadable模块

yarn add react-loadable

1. 将Detail改成异步组件

\src\App.js

import React, { Fragment } from 'react';

import { Provider } from 'react-redux';

import { BrowserRouter, Route } from 'react-router-dom';

import Header from './common/header';

import Home from './pages/home';

import Detail from './pages/detail/loadable.js';

import Login from './pages/login';

import Write from './pages/write';

import { GlobalStyle } from './style';

import { GlobalIconStyle } from './statics/iconfont/iconfont';

import store from './store';

function App() {

return (

<Provider store={store}>

<Fragment>

<GlobalStyle />

<GlobalIconStyle />

<BrowserRouter>

<div>

<Header />

<Route path='/' exact component={Home}></Route>

<Route path='/login' exact component={Login}></Route>

<Route path='/write' exact component={Write}></Route>

<Route path='/detail/:id' exact component={Detail}></Route>

</div>

</BrowserRouter>

</Fragment>

</Provider>

);

}

export default App;

\src\pages\detail\index.js

import React, {PureComponent} from 'react';

import { connect } from 'react-redux';

import { withRouter } from 'react-router-dom';

import { DetailWrapper, Header, Content } from './style';

import { actionCreators } from './store'

class Detail extends PureComponent {

render() {

return (

<DetailWrapper>

<Header>{this.props.title}</Header>

<Content

dangerouslySetInnerHTML={{\_\_html: this.props.content}}

/>

</DetailWrapper>

)

}

componentDidMount() {

this.props.getDetail(this.props.match.params.id);

}

}

const mapState = (state) => ({

title: state.getIn(['detail', 'title']),

content: state.getIn(['detail', 'content'])

});

const mapDispatch = (dispatch) => ({

getDetail(id) {

dispatch(actionCreators.getDetail(id));

}

});

export default connect(mapState, mapDispatch)(withRouter(Detail));

\src\pages\detail\loadable.js

import React from 'react';

import Loadable from 'react-loadable';

const LoadableComponent = Loadable({

loader: () => import('./'),

loading() {

return <div>正在加载</div>

}

});

export default () => <LoadableComponent/>

上线

根目录下执行 npm run build

打包出来的一个目录./build

安装服务器npm i serve

在package.json下

增加”package”: “.”

在./build目录外执行serve -s build