link: null
title: 珠峰架构师成长计划
description: src\index.js
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=40 sentences=199, words=1460

1.安装 dva-cli

```
$ npm install -g dva-cli
```

2. 生成项目

```
dva new zfchat2 --demo
cd zfchat
npm i --registry=https:
npm start
```

3. 配置路由

src\index.js

```
import dva from 'dva';
import { Router, Route, Switch } from 'dva/router';
import Login from './routes/Login';
import Rooms from './routes/Rooms';
import Room from './routes/Room';
import 'antd/dist/antd.css';
import user from './model/user';
import rooms from './model/rooms';
import room from './model/room';
const app = dva();

app.model(user);
app.model(rooms);
app.model(room);

app.router(({ history }) =>
  <Router history={history}>
    <Switch>
      <Route path="/" exact component={Login} />
      <Route path="/rooms" exact component={Rooms} />
      <Route path="/rooms/:id" exact component={Room} />
    Switch>
  Router>
);

app.start('#root');
```

4. 编写登录页面

/src/components/Header.js

```
import React,{Component} from 'react';
import {Layout,Menu,Icon} from 'antd';
import {Link} from 'dva/router'
import styles from './Header';
export default class Header extends Component{
    render() {
        return (
            <Layout.Header>
                    <Menu
                        theme="dark"
                        mode="horizontal"
                        defaultSelectedKeys={["home"]}
                        style={{lineHeight:'64px'}}
                    >
                        <Menu.Item key="home">
                            <Link to="/"><Icon type="home"/>首页Link>
                        Menu.Item>
                    Menu>
            Layout.Header>
        )
    }
}
```

/src/routes/Login.js

```
import React,{Component} from 'react';
import {Layout,Row,Col} from 'antd';
import LoginForm from '../components/LoginForm';
import Header from '../components/Header';
import {connect} from 'dva';
class Login extends Component{
    render() {
        return (
            <Layout className="layout">
                <Header/>
                <Layout.Content>
                    <Row>
                        <Col offset={2} span={20}>
                            <LoginForm dispatch={this.props.dispatch}/>
                        Col>
                    Row>
                Layout.Content>
            Layout>
        )
    }
}
export default connect(
    state=>state.user
)(Login);
```

```
import React,{Component} from 'react';
import {Form,Input,Button,Icon} from 'antd';
function hasErrors(fieldsError) {
    return Object.keys(fieldsError).some(field=>fieldsError[field]);
}
class LoginForm extends Component{
    componentDidMount() {
        this.props.form.validateFields();
    }
    handleSubmit=(event) => {
        event.preventDefault();
        this.props.form.validateFields((err,values) => {
            if (!err) {
                this.props.dispatch({type: 'user/login',payload: values});
                this.props.form.resetFields();
            }
        });
    }
    render() {
        let {getFieldDecorator,isFieldTouched,getFieldError,getFieldsError}=this.props.form;
        const contentError=isFieldTouched('email')&&getFieldError('email');
        return(
            <Form onSubmit={this.handleSubmit}>
                <Form.Item
                  validateStatus={contentError?'error':''}
                  help={contentError||''}
                >
                    {
                        getFieldDecorator('email',{type:'email',rules: [{required: true,message: '请输入邮箱'}]})(
                            <Input  prefix={<Icon type="email" style={{color:'rgba(0,0,0,.25)'}}/>} placeholder="请输入邮箱"/>
                        )
                    }
                    <Button type="primary" htmlType="submit" disabled={hasErrors(getFieldsError())}>
                      <Icon type="mail"/> 登陆
                    Button>
                Form.Item>
            Form>
        )
    }
}
export default Form.create()(LoginForm);
```

/src/models/user.js

```
import userService from '../service/user';
import { routerRedux } from 'dva/router';
export default {
    namespace: 'user',
    state: {
        user: null
    },
    reducers: {
        user(state, action) {
            return { ...state, user: action.payload };
        }
    },
    effects: {
        *login({ payload }, { put, call }) {
            const { data: token } = yield call(userService.login, payload);
            sessionStorage.setItem('token', token);
            yield put(routerRedux.push('/rooms'));
        },
        *validate(action, { put, call }) {
            const { code, data: user } = yield call(userService.validate);
            if (code == 0) {
                yield put({ type: 'user', payload: user });
            } else {
                yield put(routerRedux.push('/'));
            }
        }
    }
}
```

```
import request from '../utils/request';
function login(data) {
    return request('http://localhost:7001/login', {
        method: 'POST',
        headers: {
            "Content-Type": "application/json"
        },
        body: JSON.stringify(data)
    });
}
function validate() {
    return request('http://localhost:7001/validate', {
        method: 'POST',
        headers: {
            "Content-Type": "application/json"
        },
        body: JSON.stringify({ token: sessionStorage.getItem('token') })
    });
}
export default {
    login,
    validate
}
```

5. 编写房间列表

src/routes/Rooms.js

```
import React, { Component } from 'react';
import { Layout, Card, Row, Col, Input, Badge, Avatar, Button } from 'antd';
import { connect } from 'dva';
import { Link } from 'dva/router';
import Header from '../../components/Header';

class Rooms extends Component {
    handleChange = (event) => {
        this.props.dispatch({ type: 'rooms/changeKeyword', payload: event.target.value });
    }
    createRoom = () => {
        this.props.dispatch({ type: 'rooms/createRoom', payload: { name: this.props.keyword } });
    }
    render() {
        return (
            <Layout>
                <Header />
                <Row style={{ margin: '10px 0px' }}>
                    <Col span={24}>
                        <Input placeholder="搜索房间" onChange={this.handleChange} />
                    Col>
                Row>
                <Row gutter={16} style={{ margin: '10px 0px' }}>
                    {
                        this.props.rooms.map(room => (
                            <Col key={room._id} span={6} style={{ margin: '10px' }}>
                                <Link to={`/rooms/${room._id}`}>
                                    <Card
                                        title={room.name}
                                        extra={<Badge count={(room.users ? room.users.length : 0) + '人'} />}
                                    >
                                        {
                                            room.users && room.users.map(user => (
                                                <Avatar key={user.id} src={user.avatar} />
                                            ))
                                        }
                                    Card>
                                Link>
                            Col>
                        ))
                    }
                Row>
                <Row>
                    {
                        this.props.rooms.length == 0 && (
                            <Row>
                                <Col span={24} style={{ textAlign: 'center' }}>
                                    <Button type="primary" onClick={this.createRoom}>点击创建Button>
                                Col>
                            Row>
                        )
                    }
                Row>
            Layout>
        )
    }
}
export default connect(
    state => (
        {
            keyword: state.rooms.keyword,
            rooms: state.rooms.rooms.filter(item => item.name.indexOf(state.rooms.keyword) != -1)
        }
    )
)(Rooms);
```

src/models/rooms.js

```
import roomsService from '../service/rooms';
export default {
    namespace: 'rooms',
    state: {
        rooms: [],
        keyword: ''
    },
    subscriptions: {
        setup({ dispatch, history }) {
            history.listen(({ pathname }) => {
                if (pathname == '/rooms') {
                    dispatch({ type: 'user/validate' });
                    dispatch({ type: 'getAllRooms' });
                }
            });
        }
    },
    effects: {
        *getAllRooms({ }, { call, put }) {
            const { data: rooms } = yield call(roomsService.getAllRooms);
            console.log('rooms', rooms);
            yield put({ type: 'allRooms', payload: rooms });
        },
        *createRoom({ payload }, { put, call, select }) {
            const { data: room } = yield call(roomsService.createRoom, payload);
            yield put({ type: 'roomAdded', payload: room });
        }
    },
    reducers: {
        changeKeyword(state, { payload }) {
            return { ...state, keyword: payload };
        },
        allRooms(state, { payload }) {
            return { ...state, rooms: [...payload] };
        },
        roomAdded(state, { payload }) {
            return { ...state, rooms: [...state.rooms, payload] };
        }
    }
}
```

src/services/rooms.js

```
import request from '../utils/request';
function createRoom(data) {
    return request('http://localhost:7001/createRoom', {
        method: 'POST',
        headers: {
            "Content-Type": "application/json"
        },
        body: JSON.stringify(data)
    });
}
function getAllRooms(data) {
    return request('http://localhost:7001/getAllRooms', {
        method: 'GET'
    });
}
export default {
    createRoom,
    getAllRooms
}
```

6. 编写房间列表

src\routes\Room\index.js

```
import React, { Component } from 'react';
import { connect } from 'dva';
import Header from '../../components/Header';
import { Layout, Row, Col, List, Input, Icon, Avatar } from 'antd';
import ChatForm from '../../components/ChatForm';

class Room extends Component {
    componentDidMount() {
        this.scrollToTop();
    }
    componentDidUpdate() {
        this.scrollToTop();
    }
    scrollToTop = () => {
        this.chatList.scrollTop = this.chatList.scrollHeight;
    }
    render() {
        return (
            <Layout className="layout">
                <Header />
                <Layout.Content>
                    <Row>
                        <Col offset={2} span={20}>
                            <div
                                ref={ref => this.chatList = ref}
                                style={{ maxHeight: '600px', minHeight: '300px', overflow: 'auto' }}
                            >
                                <List
                                    header={<div style={{ textAlign: 'center' }}>欢迎来到珠峰聊天室div>}
                                    size="large"
                                    bordered={false}
                                    dataSource={this.props.messages}
                                    renderItem={
                                        item => (
                                            <List.Item>
                                                <List.Item.Meta
                                                    avatar={<Avatar src={item.user && item.user.avatar} />}
                                                    title={item.user && item.user.name}
                                                    description={item.user && item.user.email}
                                                />
                                                <div>{item.content}div>
                                            List.Item>
                                        )
                                    }
                                >
                                List>
                            div>
                            <ChatForm
                                dispatch={this.props.dispatch}
                                room={this.props.match.params.id} />
                        Col>
                    Row>
                Layout.Content>
            Layout>
        )
    }
}
export default connect(
    state => state.room
)(Room);
```

/src/models/room.js

```
import pathToRegexp from 'path-to-regexp';
import routerRedux from 'dva/router';
export default {
    namespace: 'room',
    state: {
        messages: [],
        room: ''
    },
    subscriptions: {
        setup({ dispatch, history }) {
            history.listen(({ pathname }) => {
                const match = pathToRegexp('/rooms/:id').exec(pathname);
                if (match) {
                    let room = match[1];
                    dispatch({ type: 'setRoom', payload: room });
                    dispatch({ type: 'user/validate' });
                    let io = require('socket.io-client');
                    let socket = io('http://localhost:7001', {
                        query: { token: sessionStorage.getItem('token'), room }
                    });
                    socket.on('connect', () => {
                        socket.emit('getAllMessages', room);
                    });
                    socket.on('allMessages', messages => {
                        dispatch({ type: 'allMessages', payload: messages });
                    });
                    socket.on('messageAdded', message => {
                        dispatch({ type: 'messageAdded', payload: message });
                    });
                    socket.on('online', message => {
                        dispatch({ type: 'messageAdded', payload: message });
                    });
                    socket.on('offline', message => {
                        dispatch({ type: 'messageAdded', payload: message });
                    });
                    socket.on('needLogin', message => {
                        dispatch(routerRedux.push('/'));
                    });
                    window.socket = socket;
                }
            });
        }
    },
    effects: {
        *addMessage(action, { put, call, select }) {
            const { user, room } = yield select(state => state);
            let message = action.payload;
            message.user = user.user._id;
            message.room = room.room;
            window.socket.emit('addMessage', message);
        }
    },
    reducers: {
        setRoom(state, action) {
            return { ...state, room: action.payload };
        },
        allMessages(state, action) {
            return { ...state, messages: action.payload };
        },
        messageAdded(state, action) {
            return { ...state, messages: [...state.messages, action.payload] }
        }
    }

}
```

src/components/ChatForm.js

```
import React, { Component } from 'react';
import { Form, Input, Button, Icon } from 'antd';
class ChatForm extends Component {
    componentDidMount() {
        this.props.form.validateFields();
    }
    handleSubmit = event => {
        event.preventDefault();
        this.props.form.validateFields((err, values) => {
            if (!err) {
                this.props.dispatch({ type: 'room/addMessage', payload: values });
                this.props.form.resetFields();
            }
        });
    }
    render() {
        let { getFieldDecorator, isFieldTouched, getFieldError, getFieldsError } = this.props.form;
        const contentError = isFieldTouched('content') && getFieldError('content');
        return (
            <Form onSubmit={this.handleSubmit}>
                <Form.Item
                    validateStatus={contentError ? 'error' : ''}
                    help={contentError || ''}
                >
                    {
                        getFieldDecorator('content', {
                            rules: [
                                [{ required: true, message: '请输入内容' }]
                            ]
                        })(
                            <Input
                                placeholder="请输入聊天信息"
                                prefix={<Icon type="wechat" style={{ color: 'rgba(0,0,0,.25)' }} />} />
                        )
                    }
                    <Button type="primary" htmlType="submit" disabled={contentError}>
                        <Icon type="mail" />发言
                    Button>
                Form.Item>
            Form>
        )

    }
}

export default Form.create()(ChatForm);
```

参考