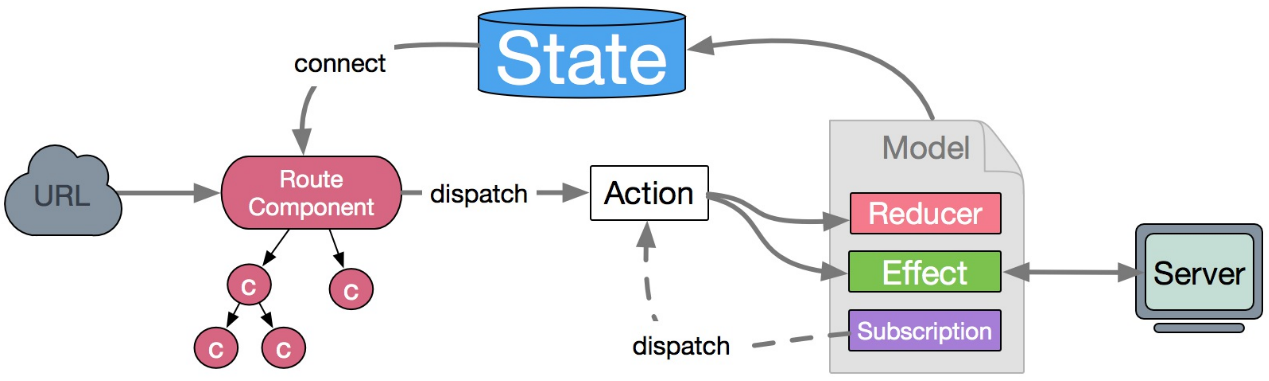
link: null
title: 珠峰架构师成长计划
description: null
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=81 sentences=207, words=2064

1.dva介绍

- dva (https://github.com/dvajs/dva)首先是一个基于 `redux` 和 `redux-saga` 的数据流方案，然后为了简化开发体验,dva 还额外内置了 `react-router` 和 `fetch`,所以也可以理解为一个轻量级的应用框架



1.1 前置知识

- react
- react-router-dom
- redux
- react-redux
- react-saga
- redux-first-history

2. 初始化项目

```
create-react-app zhufeng-dva-source
cd  zhufeng-dva-source
npm install dva
npm install  redux react-redux redux-saga react-router-dom redux-first-history  --save
npm start
```

3. 基本计数器

3.1 使用

3.1.1 index.js

```
import React from 'react';
import dva, { connect } from './dva';
const app = dva();
app.model({
    namespace: 'counter',
    state: { number: 0 },
    reducers: {
        add(state) {
            return { number: state.number + 1 };
        }
    }
});
const actionCreator = app.createAction();
function Counter(props) {
    return (
        <div>
            <p>{props.number}p>
            <button onClick={() => props.dispatch({ type: "counter/add" })}>+button>
        div>
    )
}
const ConnectedCounter = connect(
    (state) => state.counter
)(Counter

);
app.router(() => <ConnectedCounter />);
app.start('#root');
```

3.2 实现

3.2.1 dva\index.js

src\dva\index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import { createStore, combineReducers } from 'redux';
import { connect, Provider } from 'react-redux';
import prefixNamespace from './prefixNamespace';
export { connect };

function dva() {
    const app = {
        _models: [],
        model,
        router,
        _router: null,
        start,
        createAction
    }
    const initialReducers = {};
    function model(model) {
        const prefixedModel = prefixNamespace(model);
        app._models.push(prefixedModel);
        return prefixedModel;
    }
    function router(router) {
        app._router = router;
    }
    function createAction() {
        let actionCreators = {};
        for (const model of app._models) {
            let { reducers } = model;
            for (let key in reducers) {

                actionCreators[key] = () => ({ type: key })
            }
        }
        return actionCreators;
    }
    function start(root) {
        for (const model of app._models) {
            initialReducers[model.namespace] = getReducer(model);
        }
        let rootReducer = createReducer();
        let store = createStore(rootReducer);
        ReactDOM.render(<Provider store={store}>{app._router()}Provider>, document.querySelector(root));
        function createReducer() {
            return combineReducers(initialReducers);
        }
    }

    return app;
}

function getReducer(model) {
    let { reducers, state: defaultState } = model;
    let reducer = (state = defaultState, action) => {
        let reducer = reducers[action.type];
        if (reducer) {
            return reducer(state, action);
        }
        return state;
    }
    return reducer;
}

export default dva;
```

3.2.2 prefixNamespace.js

src\dva\prefixNamespace.js

```
function prefix(obj, namespace) {
    return Object.keys(obj).reduce((memo, key) => {
        const newKey = `${namespace}/${key}`;
        memo[newKey] = obj[key];
        return memo;
    }, {});
}
export default function prefixNamespace(model) {
    if (model.reducers)
        model.reducers = prefix(model.reducers, model.namespace);
    return model;
}
```

4. 支持effects

4.1 使用

4.1.1 src\index.js

src\index.js

```
import React from 'react';
import dva, { connect } from './dva';
const app = dva();
app.model({
    namespace: 'counter',
    state: { number: 0 },
    reducers: {
        add(state) {
            return { number: state.number + 1 };
        }
    },
+    effects: {
+        *asyncAdd(action, { call, put }) {
+            yield call(delay, 1000);
+            yield put({ type: 'counter/add' });
+        }
+    }
});
function Counter(props) {
    return (

            {props.number}
             props.dispatch({ type: "counter/add" })}>+
+             props.dispatch({ type: "counter/asyncAdd" })}>异步+

    )
}
const ConnectedCounter = connect(
    (state) => state.counter
)(Counter);
app.router(() => );
app.start('#root');

+function delay(ms) {
+    return new Promise((resolve) => {
+        setTimeout(function () {
+            resolve();
+        }, ms);
+    });
+}
```

4.2 实现

4.2.1 dva\index.js

src\dva\index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
+import { createStore, combineReducers, applyMiddleware } from 'redux';
+import createSagaMiddleware from 'redux-saga';
+import * as sagaEffects from 'redux-saga/effects';
import { connect, Provider } from 'react-redux';
import prefixNamespace from './prefixNamespace';
export { connect };

function dva() {
    const app = {
        _models: [],
        model,
        router,
        _router: null,
        start
    }
    const initialReducers = {};
    function model(model) {
        const prefixedModel = prefixNamespace(model);
        app._models.push(prefixedModel);
        return prefixedModel;
    }
    function router(router) {
        app._router = router;
    }

    function start(root) {
        for (const model of app._models) {
            initialReducers[model.namespace] = getReducer(model);
        }
        let rootReducer = createReducer();
+        const sagas = getSagas(app);
+        const sagaMiddleware = createSagaMiddleware();
+        let store = createStore(rootReducer, applyMiddleware(sagaMiddleware));
+        sagas.forEach(saga => sagaMiddleware.run(saga));
        ReactDOM.render({app._router()}, document.querySelector(root));
        function createReducer() {
            return combineReducers(initialReducers);
        }
    }
+    function getSagas(app) {
+        let sagas = [];
+        for (const model of app._models) {
+            sagas.push(getSaga(model));
+        }
+        return sagas;
+    }
    return app;
}
+function getSaga(model) {
+    const { effects } = model;
+    return function* () {
+        for (const key in effects) {//key=asyncAdd
+            yield sagaEffects.takeEvery(key, function* (action) {
+                yield effects[key](action, {
+                    ...sagaEffects, put: action => sagaEffects.put(
+                        { ...action, type: prefixType(action.type, model.namespace) })
+                });
+            });
+        }
+    }
+}
+function prefixType(type, model) {
+    if (type.indexOf('/') === -1) {
+        return `${model.namespace}/${type}`;
+    }
+    return type;
+}
function getReducer(model) {
    let { reducers, state: defaultState } = model;
    let reducer = (state = defaultState, action) => {
        let reducer = reducers[action.type];
        if (reducer) {
            return reducer(state, action);
        }
        return state;
    }
    return reducer;
}

export default dva;
```

4.2.2 prefixNamespace.js #

src\dva\prefixNamespace.js

```
import { NAMESPACE_SEP } from './constants';
function prefix(obj, namespace) {
    return Object.keys(obj).reduce((memo, key) => {
        const newKey = `${namespace}${NAMESPACE_SEP}${key}`;
        memo[newKey] = obj[key];
        return memo;
    }, {});
}
export default function prefixNamespace(model) {
    if (model.reducers)
        model.reducers = prefix(model.reducers, model.namespace);
+    if (model.effects) {
+       model.effects = prefix(model.effects, model.namespace);
+    }
    return model;
}
```

5. 支持路由

5.1 使用

5.1.1 src\index.js

src\index.js

```
import React from 'react';
import dva, { connect } from './dva';
+import { Router, Route,Link } from './dva/router';
const app = dva();
app.model({
    namespace: 'counter',
    state: { number: 0 },
    reducers: {
        add(state) {
            return { number: state.number + 1 };
        }
    },
    effects: {
        *asyncAdd(action, { call, put }) {
            yield call(delay, 1000);
            yield put({ type: 'counter/add' });
        }
    }
});
function Counter(props) {
    return (

            {props.number}
             props.dispatch({ type: "counter/add" })}>+
             props.dispatch({ type: "counter/asyncAdd" })}>异步+

    )
}
const ConnectedCounter = connect(
    (state) => state.counter
)(Counter);
+const Home = () => Home
+app.router(() => (
+         <>
+           Home
+           Counter
+
+             } />
+             } />
+
+         </>
+));
app.start('#root');

function delay(ms) {
    return new Promise((resolve) => {
        setTimeout(function () {
            resolve();
        }, ms);
    });
}
```

5.2 实现

5.2.1 dva\index.js

src\dva\index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import { createStore, combineReducers, applyMiddleware } from 'redux';
import createSagaMiddleware from 'redux-saga';
import * as sagaEffects from 'redux-saga/effects';
import { NAMESPACE_SEP } from './constants';
import { connect, Provider } from 'react-redux';
import prefixNamespace from './prefixNamespace';
import { BrowserRouter } from 'react-router-dom';
export { connect };

function dva() {
    const app = {
        _models: [],
        model,
        router,
        _router: null,
        start
    }
    const initialReducers = {};
    function model(model) {
        const prefixedModel = prefixNamespace(model);
        app._models.push(prefixedModel);
        return prefixedModel;
    }
    function router(router) {
        app._router = router;
    }

    function start(root) {
        for (const model of app._models) {
            initialReducers[model.namespace] = getReducer(model);
        }
        let rootReducer = createReducer();
        const sagas = getSagas(app);
        const sagaMiddleware = createSagaMiddleware();
        let store = createStore(rootReducer, applyMiddleware(sagaMiddleware));
        sagas.forEach(saga => sagaMiddleware.run(saga));
        ReactDOM.render(
+
+
+                {app._router()}
+
+            ,
        document.querySelector(root))
        function createReducer() {
            return combineReducers(initialReducers);
        }
    }
    function getSagas(app) {
        let sagas = [];
        for (const model of app._models) {
            sagas.push(getSaga(model));
        }
        return sagas;
    }
    return app;
}
function getSaga(model) {
    const { effects } = model;
    return function* () {
        for (const key in effects) {//key=asyncAdd
            yield sagaEffects.takeEvery(key, function* (action) {
                yield effects[key](action, {
                    ...sagaEffects, put: action => sagaEffects.put(
                        { ...action, type: prefixType(action.type, model.namespace) })
                });
            });
        }
    }
}
function prefixType(type, model) {
    if (type.indexOf('/')
        return `${model.namespace}${NAMESPACE_SEP}${type}`;
    }
    return type;
}
function getReducer(model) {
    let { reducers, state: defaultState } = model;
    let reducer = (state = defaultState, action) => {
        let reducer = reducers[action.type];
        if (reducer) {
            return reducer(state, action);
        }
        return state;
    }
    return reducer;
}

export default dva;
```

5.2.2 router.js

src\dva\router.js

```
export * from 'react-router-dom';
```

6. 跳转路径

6.1 使用

6.1.1 src\index.js

src\index.js

```
import React from 'react';
import dva, { connect } from './dva';
+import { Router, Route,Link,routerRedux } from './dva/router';
const app = dva();
app.model({
    namespace: 'counter',
    state: { number: 0 },
    reducers: {
        add(state) {
            return { number: state.number + 1 };
        }
    },
    effects: {
        *asyncAdd(action, { call, put }) {
            yield call(delay, 1000);
            yield put({ type: 'counter/add' });
        },
+       *goto({ to }, { put }) {
+          yield put(routerRedux.push(to));
+       }
    }
});
function Counter(props) {
    return (

            {props.number}
             props.dispatch({ type: "counter/add" })}>+
             props.dispatch({ type: "counter/asyncAdd" })}>异步+
+             props.dispatch({ type: "counter/goto", to: '/' })}>跳转到/

    )
}
const ConnectedCounter = connect(
    (state) => state.counter
)(Counter);
const Home = () => Home
app.router(() => (
        <>
            Home
            Counter

        </>
));
app.start('#root');

function delay(ms) {
    return new Promise((resolve) => {
        setTimeout(function () {
            resolve();
        }, ms);
    });
}
```

6.2 实现

6.2.1 dva\index.js

src\dva\index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import { createStore, combineReducers, applyMiddleware } from 'redux';
import createSagaMiddleware from 'redux-saga';
import * as sagaEffects from 'redux-saga/effects';
import { NAMESPACE_SEP } from './constants';
import { connect, Provider } from 'react-redux';
import prefixNamespace from './prefixNamespace';
+import { createBrowserHistory } from 'history';
+import { HistoryRouter } from "redux-first-history/rr6";
+import { createReduxHistoryContext } from "redux-first-history";
+const { routerReducer, routerMiddleware, createReduxHistory } = createReduxHistoryContext({ history: createBrowserHistory() });
export { connect };

function dva() {
    const app = {
        _models: [],
        model,
        router,
        _router: null,
        start
    }
+   const initialReducers = { router: routerReducer };
    function model(model) {
        const prefixedModel = prefixNamespace(model);
        app._models.push(prefixedModel);
        return prefixedModel;
    }
    function router(router) {
        app._router = router;
    }

    function start(root) {
        for (const model of app._models) {
            initialReducers[model.namespace] = getReducer(model);
        }
        let rootReducer = createReducer();
        const sagas = getSagas(app);
        const sagaMiddleware = createSagaMiddleware();
+       const store = createStore(rootReducer, applyMiddleware(routerMiddleware, sagaMiddleware));
        sagas.forEach(saga => sagaMiddleware.run(saga));
        ReactDOM.render(

                {app._router()}

            , document.querySelector(root))
        function createReducer() {
            return combineReducers(initialReducers);
        }
    }
    function getSagas(app) {
        let sagas = [];
        for (const model of app._models) {
            sagas.push(getSaga(model));
        }
        return sagas;
    }
    return app;
}
function getSaga(model) {
    const { effects } = model;
    return function* () {
        for (const key in effects) {
            yield sagaEffects.takeEvery(key, function* (action) {
                yield effects[key](action, {
                    ...sagaEffects, put: action => sagaEffects.put(
                        { ...action, type: prefixType(action.type, model.namespace) })
                });
            });
        }
    }
}
function prefixType(type, model) {
    if (type.indexOf('/')
        return `${model.namespace}${NAMESPACE_SEP}${type}`;
    }
    return type;
}
function getReducer(model) {
    let { reducers, state: defaultState } = model;
    let reducer = (state = defaultState, action) => {
        let reducer = reducers[action.type];
        if (reducer) {
            return reducer(state, action);
        }
        return state;
    }
    return reducer;
}

export default dva;
```

6.2.2 src\dva\router.js

src\dva\router.js

```
import * as routerRedux from 'redux-first-history';
export * from 'react-router-dom';
export {
    routerRedux
}
```