---

link: null
title: 珠峰架构师成长计划
description: 需要生成一个tsconfig.json文件来告诉ts-loader如何编译代码TypeScript代码
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=86 sentences=186, words=1355

---

1. 初始化项目

```
mkdir react-typesript
cd react-typesript
cnpm init -y
touch .gitignore
```

2.安装依赖

```
cnpm i react react-dom @types/react @types/react-dom react-router-dom @types/react-router-dom react-transition-group @types/react-transition-group react-swipe
@types/react-swipe  -S
cnpm i webpack webpack-cli webpack-dev-server html-webpack-plugin -D
cnpm i typescript ts-loader source-map-loader -D
cnpm i redux react-redux @types/react-redux redux-thunk  redux-logger @types/redux-logger -S
cnpm i connected-react-router -S
```

- ts-loader可以让Webpack使用TypeScript的标准配置文件tsconfig.json编译TypeScript代码。
- source-map-loader使用任意来自Typescript的sourcemap输出，以此通知webpack何时生成自己的sourcemaps,这让你在调试最终生成的文件时就好像在调试TypeScript源码一样。

3.支持typescript

需要生成一个tsconfig.json文件来告诉ts-loader如何编译代码TypeScript代码

```
tsc --init
```

```
{
  "compilerOptions": {
    "outDir": "./dist",
    "sourceMap": true,
    "noImplicitAny": true,
    "module": "commonjs",
    "target": "es5",
    "jsx": "react"
  },
  "include": [
    "./src/**/*"
  ]
}
```

- outDir 指定输出目录
- sourceMap：把 ts 文件编译成 js 文件的时候，同时生成对应的sourceMap文件
- noImplicitAny：如果为true的话，TypeScript 编译器无法推断出类型时，它仍然会生成 JavaScript 文件，但是它也会报告一个错误
- module：代码规范
- target：转换成es5
- jsx：react模式会生成React.createElement，在使用前不需要再进行转换操作了，输出文件的扩展名为.js
- include：需要编译的目录。

4.编写webpack配置文件

webpack.config.js

```
const webpack=require('webpack');
const HtmlWebpackPlugin=require('html-webpack-plugin');
const path=require('path');
module.exports={
    mode: 'development',
    entry: "./src/index.tsx",
    output: {
        filename: "bundle.js",
        path: path.join(__dirname,'dist')
    },
    devtool: "source-map",
    devServer: {
        hot: true,
        contentBase: path.join(__dirname,'dist'),
        historyApiFallback: {
            index:'./index.html'
        }
    },
    resolve: {
        extensions: [".ts", ".tsx", ".js", ".json"]
    },

    module: {
        rules: [{
                test: /\.tsx?$/,
                loader: "ts-loader"
            },

            {
                enforce: "pre",
                test: /\.js$/,
                loader: "source-map-loader"
            }
        ]
    },

    plugins: [
        new HtmlWebpackPlugin({
            template:'./src/index.html'
        }),
        new webpack.HotModuleReplacementPlugin()
    ],
};
```

5.计数器组件

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import Counter from './components/Counter';
ReactDOM.render((
    <Counter number={100}/>
),document.getElementById('root'));
```

src/components/Counter.tsx

```
import * as React from 'react';
export interface Props{
    number: number
}
export default class Counter extends React.Component<Props>{
    render() {
        const {number}=this.props;
        return (
            <div>
                <p>{number}p>
            div>
        )
    }
}
```

6. 使用redux

src/index.tsx

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import Counter from './components/Counter';
import { Provider } from 'react-redux';
import store from './store';
ReactDOM.render((
    <Provider store={store}>
        <Counter/>
    Provider>
),document.getElementById('root'));
```

src/components/Counter.tsx

```
import * as React from 'react';
import { connect } from 'react-redux';
import { Store } from '../store/types';
import * as actions from '../store/actions';
export interface Props{
    number: number,
    increment: any,
    decrement: any
}
class Counter extends React.Component<Props>{
    render() {
        const {number,increment,decrement}=this.props;
        return (
            <div>
                <p>{number}p>
                <button onClick={increment}>+button>
                <button onClick={decrement}>-button>
            div>
        )
    }
}

let mapStateToProps=function (state:Store):Store {
    return state;
}
export default connect(mapStateToProps,actions)(Counter);
```

src/store/index.tsx

```
import {createStore } from 'redux'
import reducers from './reducers';
let store=createStore(reducers);
export default store;
```

src/store/action-types.tsx

```
export const INCREMENT='INCREMENT';
export const DECREMENT='DECREMENT';
```

src/store/reducers/index.tsx

```
import * as types from '../action-types';
import { Store } from '../types';
import {Action} from '../actions';
export default function (state: Store={ number: 0 },action: Action): Store {
    switch (action.type) {
        case types.INCREMENT:
            return {...state,number:state.number+1};
        case types.DECREMENT:
            return {...state,number:state.number-1};
        default:
            return state;
    }
}
```

src/store/actions/index.tsx

```
import {INCREMENT,DECREMENT} from '../action-types';
export interface Increment{
    type:typeof INCREMENT
}
export interface Decrement{
    type:typeof DECREMENT
}
export type Action=Increment|Decrement;

export function increment(): Increment {
    return { type: INCREMENT };
}
export function decrement():Decrement {
    return { type: DECREMENT };
}
```

src/store/types/index.tsx

```
export interface Store{
    number: number
}
```

7. 合并reducers

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import Counter1 from './components/Counter1';
import Counter2 from './components/Counter2';
import { Provider } from 'react-redux';
import store from './store';
ReactDOM.render((
    <Provider store={store}>
        <React.Fragment>
          <Counter1/>
          <Counter2/>
        React.Fragment>
    Provider>
),document.getElementById('root'));
```

src/store/action-types.tsx

```
export const INCREMENT='INCREMENT';
export const DECREMENT='DECREMENT';

export const INCREMENT1='INCREMENT1';
export const DECREMENT1='DECREMENT1';

export const INCREMENT2='INCREMENT2';
export const DECREMENT2='DECREMENT2';
```

src/store/reducers/index.tsx

```
import counter1 from './counter1';
import counter2 from './counter2';
import { combineReducers } from 'redux';
let reducers=combineReducers({
    counter1,
    counter2
});
export default reducers;
```

src/store/types/index.tsx

```
export interface Store{
    counter1: Counter1,
    counter2: Counter2
}
export interface Counter1{
    number: number
}
export interface Counter2{
    number: number
}
```

src/components/Counter1.tsx

```
import * as React from 'react';
import { connect } from 'react-redux';
import * as types from '../store/types';
import * as actions from '../store/actions/counter1';
export interface Props{
    number: number,
    increment1: any,
    decrement1: any
}
class Counter1 extends React.Component<Props>{
    render() {
        const {number,increment1,decrement1}=this.props;
        return (
            <div>
                <p>{number}p>
                <button onClick={increment1}>+button>
                <button onClick={decrement1}>-button>
            div>
        )
    }
}

let mapStateToProps=function (state:types.Store):types.Counter1 {
    return state.counter1;
}

export default connect(mapStateToProps,actions)(Counter1);
```

src/components/Counter2.tsx

```
import * as React from 'react';
import { connect } from 'react-redux';
import * as types from '../store/types';
import * as actions from '../store/actions/counter2';
export interface Props{
    number: number,
    increment2: any,
    decrement2: any
}
class Counter2 extends React.Component<Props>{
    render() {
        const {number,increment2,decrement2}=this.props;
        return (
            <div>
                <p>{number}p>
                <button onClick={increment2}>+button>
                <button onClick={decrement2}>-button>
            div>
        )
    }
}

let mapStateToProps=function (state:types.Store):types.Counter2 {
    return state.counter2;
}

export default connect(mapStateToProps,actions)(Counter2);
```

src/store/actions/counter1.tsx

```
import {INCREMENT1,DECREMENT1} from '../action-types';
export interface Increment1{
    type:typeof INCREMENT1
}
export interface Decrement1{
    type:typeof DECREMENT1
}
export type Action=Increment1|Decrement1;

export function increment1(): Increment1 {
    return { type: INCREMENT1 };
}
export function decrement1():Decrement1 {
    return { type: DECREMENT1 };
}
```

src/store/actions/counter2.tsx

```
import {INCREMENT2,DECREMENT2} from '../action-types';
export interface Increment2{
    type:typeof INCREMENT2
}
export interface Decrement2{
    type:typeof DECREMENT2
}
export type Action=Increment2|Decrement2;

export function increment2(): Increment2 {
    return { type: INCREMENT2 };
}
export function decrement2():Decrement2 {
    return { type: DECREMENT2 };
}
```

src/store/reducers/counter1.tsx

```
import * as types from '../action-types';
import { Counter1 } from '../types';
import {Action} from '../actions/counter1';
export default function (state: Counter1={ number: 0 },action: Action): Counter1 {
    switch (action.type) {
        case types.INCREMENT1:
            return {...state,number:state.number+1};
        case types.DECREMENT1:
            return {...state,number:state.number-1};
        default:
            return state;
    }
}
```

src/store/reducers/counter2.tsx

```
import * as types from '../action-types';
import { Counter2 } from '../types';
import {Action} from '../actions/counter2';
export default function (state: Counter2={ number: 0 },action: Action): Counter2 {
    switch (action.type) {
        case types.INCREMENT2:
            return {...state,number:state.number+1};
        case types.DECREMENT2:
            return {...state,number:state.number-1};
        default:
            return state;
    }
}
```

8.配置路由

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import Counter1 from './components/Counter1';
import Counter2 from './components/Counter2';
import { Provider } from 'react-redux';
import store from './store';
import {BrowserRouter as Router, Route,Link } from 'react-router-dom';
ReactDOM.render((
    <Provider store={store}>
        <Router >
            <React.Fragment>
                <Link to="/counter1">counter1Link>
                <Link to="/counter2">counter2Link>
                <Route path="/counter1" component={Counter1} />
                  <Route path="/counter2" component={Counter2}/>
         React.Fragment>
        Router>

    Provider>
),document.getElementById('root'));
```

9. connected-react-router

src/components/Counter1.tsx

```
import * as React from 'react';
import { connect } from 'react-redux';
import * as types from '../store/types';
import * as actions from '../store/actions/counter1';
export interface Props{
    number: number,
    increment1: any,
    decrement1: any,
    goCounter2: any
}
class Counter1 extends React.Component<Props>{
    render() {
        const {number,increment1,decrement1,goCounter2}=this.props;
        return (
            <div>
                <p>{number}p>
                <button onClick={increment1}>+button>
                <button onClick={decrement1}>-button>
                <button onClick={goCounter2}>goCounter2button>
            div>
        )
    }
}

let mapStateToProps=function (state:types.Store):types.Counter1 {
    return state.counter1;
}

export default connect(mapStateToProps,actions)(Counter1);
```

src/index.tsx

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import Counter1 from './components/Counter1';
import Counter2 from './components/Counter2';
import { Provider } from 'react-redux';
import store from './store';
import {Route,Link } from 'react-router-dom';
import { ConnectedRouter } from 'connected-react-router'
import history from './store/history';
ReactDOM.render((
    <Provider store={store}>
        <ConnectedRouter history={history}>
            <React.Fragment>
                <Link to="/counter1">counter1Link>
                <Link to="/counter2">counter2Link>
            <Route path="/counter1" component={Counter1} />
              <Route path="/counter2" component={Counter2}/>
         React.Fragment>
        ConnectedRouter>
    Provider>
),document.getElementById('root'));
```

src/store/actions/counter1.tsx

```
import {INCREMENT1,DECREMENT1} from '../action-types';
import { push } from 'connected-react-router';

export interface Increment1{
    type:typeof INCREMENT1
}
export interface Decrement1{
    type:typeof DECREMENT1
}
export type Action=Increment1|Decrement1;

export function increment1(): any {
    return function (dispatch:any,getState:any) {
        setTimeout(function () {
            dispatch({
                type:INCREMENT1
            })
        },1000);
    }
}
export function decrement1():Decrement1 {
    return { type: DECREMENT1 };
}
export function goCounter2():any {
    return push('/counter2');
}
```

src/store/index.tsx

```
import {createStore,applyMiddleware} from 'redux'
import reducers from './reducers';
import { routerMiddleware } from 'connected-react-router'
import history from './history';
import thunk from 'redux-thunk';
import logger from 'redux-logger';
let router = routerMiddleware(history);
let store=createStore(reducers,applyMiddleware(router,thunk,logger));
export default store;
```

src/store/reducers/index.tsx

```
import counter1 from './counter1';
import counter2 from './counter2';
import { combineReducers } from 'redux';
import history from '../history';
import { connectRouter } from 'connected-react-router'
let reducers=combineReducers({
    counter1,
    counter2,
    router: connectRouter(history)
});
export default reducers;
```

src/store/history.tsx

```
import {createBrowserHistory} from 'history'
const history=createBrowserHistory()
export default history;
```

10.仓库