link: null
title: 珠峰架构师成长计划
description: render\client.js
keywords: null
author: null
date: null
publisher: 珠峰架构师成长计划
stats: paragraph=304 sentences=776, words=5657

1.渲染模式

1.1 服务器渲染

- 页面上的内容是由服务器生产的

```
npm install express --save
```

render\client.js

```
let express = require('express');
let app = express();
app.get('/', (req, res) => {
  res.send(`

            hello

    `);
});
app.listen(8080);
```

1.2 客户端渲染

- 页面上的内容由于浏览器运行JS脚本而渲染到页面上的
    - 浏览器访问服务器
    - 服务器返回一个空的HTML页面，里面有一个JS资源链接，比如 client
    - 浏览器下载JS代码并在浏览器中运行
    - 内容呈现在页面上

```
let express = require('express');
let app = express();
app.get('/', (req, res) => {
  res.send(`

            root.innerHTML = 'hello'

    `);
});
app.listen(8090);
```

1.3 为什么SSR

- 首屏等待 在客户端渲染的模式下，所有的数据请求和DOM渲染都在浏览器端完成,所以第一次访问页面时，可能会出现白屏，而服务器端渲染会在服务器端进行数据请求和DOM渲染，浏览器收到的完整的内容，可以渲染页面
- SEO SPA对搜索引擎不够友好

2.SSR+SPA同构

- 第一次访问页面是SSR，后面的访问是SPA，而且支持SEO
- 客户端和服务器端同构可以实现(尽可能复用代码)
- 工作流程
    - 服务器端运行React代码渲染出HTML字符串
    - 服务器把渲染出的HTML页面发送给了浏览器
    - 浏览器接受到HTML会渲染到页面上
    - 浏览器发现页面引用的 client.js文件会去下载
    - 浏览器下载得到的 client.js文件并在浏览器端执行
    - 浏览器中的代码接管了页面的所有内容，后面和客户端渲染是一样的

2.1 安装

```
npm install react react-dom --save
npm install webpack webpack-cli source-map-loader babel-loader @babel/preset-env @babel/preset-react webpack-merge webpack-node-externals npm-run-all nodemon  -
-save-dev
```

2.2 webpack.config.base.js

webpack.config.base.js

```
module.exports = {
    mode: 'development',
    resolve: {
        alias: {
            '@': path.resolve(__dirname, 'src')
        }
    },
    module: {
        rules: [
            {
                test: /\.js$/,
                enforce: 'pre',
                use: ['source-map-loader']
            },
            {
                test: /\.js/,
                use: {
                    loader: 'babel-loader',
                    options: {
                        presets: [
                            "@babel/preset-env",
                            "@babel/preset-react"
                        ]
                    }
                },
                exclude: /node_modules/,

            }
        ]
    }
}
```

2.3 webpack.config.client.js

webpack.config.client.js

```
const path = require('path');
const { merge } = require('webpack-merge');
const base = require('./webpack.config.base');
const config = merge(base, {
    target: 'web',
    entry: './src/client/index.js',
    output: {
        path: path.resolve('public'),
        filename: 'client.js'
    }
});
module.exports = config;
```

2.4 webpack.config.server.js

webpack.config.server.js

```
const path = require('path');
const { merge } = require('webpack-merge');
const webpackNodeExternals = require('webpack-node-externals');
const base = require('./webpack.config.base');
module.exports = merge(base, {
    target: 'node',
    entry: './src/server/index.js',
    output: {
        path: path.resolve('build'),
        filename: 'server.js'
    },
    externals: [webpackNodeExternals()]
});
```

2.5 Counter.js

src\routes\Counter.js

```
import React, { useState } from 'react';
function Counter() {
    const [number, setNumber] = useState(0);
    return (
        <div>
            <p>{number}p>
            <button onClick={() => setNumber(number + 1)}>+button>
        div>
    )
}
export default Counter;
```

2.6 src\server\index.js

src\server\index.js

```
import React from 'react';
import { renderToString } from 'react-dom/server';
import Counter from '../routes/Counter';
const express = require('express');
const app = express();
app.use(express.static('public'));
app.get('*', (req, res) => {
  const html = renderToString(
    <Counter />
  );
  res.send(`

          ssr

        ${html}

  `);
});
app.listen(3000, () => console.log("server started on 3000"));
```

2.7 src\client\index.js

src\client\index.js

```
import React from 'react';
import { hydrateRoot } from 'react-dom/client';
import Counter from '../routes/Counter';
const root = document.getElementById('root');
hydrateRoot(root, );
```

2.8 package.json

package.json

```
{
  "scripts": {
    "start": "nodemon build/server.js",
    "build": "npm-run-all --parallel build:**",
    "build:server": "webpack --config webpack.config.server.js --watch",
    "build:client": "webpack --config webpack.config.client.js --watch"
  },
}
```

3.使用路由

3.1 安装

```
npm install react-router-dom --save
```

3.2 客户端路由

- 客户端请求服务器
- 服务器返回HTML给浏览器，浏览器渲染显示页面
- 浏览器发现需要外链JS资源，加载JS资源
- 加载好的JS资源在浏览器端执行
- JS中的React代码开始实现路由功能
- 路由代码首先获取地址栏中的地址，然后根据不同的地址根据路由配置渲染对应内容

3.3 routesConfig.js

src\routesConfig.js

```
import React from 'react';
import Home from './routes/Home';
import Counter from './routes/Counter';
export default [
  {
    path: '/',
    element: <Home />,
    index: true
  },
  {
    path: '/counter',
    element: <Counter />
  }
]
```

3.4 App.js

src\App.js

```
import React from 'react';
import { useRoutes } from 'react-router-dom';
import routesConfig from './routesConfig';
function App() {
  return (
    useRoutes(routesConfig)
  )
}
export default App;
```

3.5 server\index.js

src\server\index.js

```
import React from 'react';
import { renderToString } from 'react-dom/server';
+import routesConfig from '../routesConfig';
+import { StaticRouter } from "react-router-dom/server";
+import { matchRoutes } from 'react-router-dom';
+import App from '../App';
const express = require('express');
const app = express();
app.use(express.static('public'));
app.get('*', (req, res) => {
   const html = renderToString(
+
+


   );
    res.send(`

          ssr

        ${html}

  `);
});
app.listen(3000, () => console.log("server started on 3000"));
```

3.6 src\client\index.js

src\client\index.js

```
import React from 'react';
import { hydrateRoot } from 'react-dom/client';
+import { BrowserRouter } from 'react-router-dom';
+import App from '../App';
const root = document.getElementById('root');
hydrateRoot(root,
+
+
+  );
```

3.7 src\routes\Home.js

src\routes\Home.js

```
import React from 'react';
function Home() {
  return (
    <div>
      Home
    div>
  )
}
export default Home;
```

4. 头部导航

src\components\Header\index.js

```
import React from 'react';
import { Link } from 'react-router-dom';
function Header() {
  return (
    <ul>
      <li><Link to="/">HomeLink>li>
      <li><Link to="/counter">CounterLink>li>
    ul>
  )
}
export default Header
```

4.2 App.js

src\App.js

```
import React from 'react';
import { useRoutes } from 'react-router-dom';
import routesConfig from './routesConfig';
+import Header from './components/Header';
function App() {
  return (
+   <>
+
     {useRoutes(routesConfig)}
+   </>
  )
}
export default App;
```

5. 集成redux

5.1 安装

```
npm install redux react-redux redux-thunk redux-promise redux-logger --save
```

5.2 store\index.js

src\store\index.js

```
import { createStore, combineReducers, applyMiddleware } from 'redux'
import thunk from 'redux-thunk';
import promise from 'redux-promise';
import logger from 'redux-logger';
import counter from './reducers/counter';
export function getStore() {
    const reducers = { counter }
    const combinedReducer = combineReducers(reducers);
    const store = applyMiddleware(thunk, promise,logger)(createStore)(combinedReducer);
    return store
}
```

5.3 action-types.js

src\store\action-types.js

```
export const ADD = 'ADD';
```

5.4 counter.js

src\store\reducers\counter.js

```
import { ADD } from '../action-types';
const initialState = { number: 0 };
function counter(state = initialState, action) {
  switch (action.type) {
    case ADD:
      return { number: state.number + 1 }
    default:
      return state;
  }
}
export default counter;
```

5.5 counter.js

src\store\actionCreators\counter.js

```
import { ADD } from '@/store/action-types';
const actionCreators = {
  add() {
    return { type: ADD };
  }
}
export default actionCreators;
```

5.6 src\routes\Counter.js

src\routes\Counter.js

```
import React from 'react';
+import { useDispatch, useSelector } from 'react-redux';
+import actionCreators from '@/store/actionCreators/counter';
function Counter() {
+    const number = useSelector(state => state.counter.number);
+    const dispatch = useDispatch();
    return (

            {number}
+            dispatch(actionCreators.add())}>+

    )
}
export default Counter;
```

5.7 src\App.js

src\App.js

```
import React from 'react';
import { useRoutes } from 'react-router-dom';
import routesConfig from './routesConfig';
import Header from './components/Header';
+import { Provider } from 'react-redux';
+import { getStore } from './store';
const store = getStore();
function App() {
  return (
+

      {useRoutes(routesConfig)}
+
  )
}
export default App;
```

6. 子路由并调用接口

6.1 安装

```
npm install cors axios --save-dev
```

6.2 api.js

api.js

```
const express = require('express')
const cors = require('cors');
const app = express();
app.use(cors());
const users = [{ id: 1, name: 'zhufeng1' }, { id: 2, name: 'zhufeng2' }, { id: 3, name: 'zhufeng3' }];
app.get('/api/users', (req, res) => {
  res.json({
    success: true,
    data: users
  });
});
app.listen(5000, () => console.log('api server started on port 5000'));
```

6.3 action-types.js

src\store\action-types.js

```
export const ADD = 'ADD';
+export const SET_USER_LIST = 'SET_USER_LIST';
+export const ADD_USER = 'ADD_USER';
```

6.4 user.js

src\store\reducers\user.js

```
import { ADD_USER, SET_USER_LIST } from '../action-types';
const initialState = { list: [] };
function counter(state = initialState, action) {
  switch (action.type) {
    case SET_USER_LIST:
      return { list: action.payload }
    case ADD_USER:
      return { list: [...state.list, action.payload] }
    default:
      return state;
  }
}
export default counter;
```

6.5 user.js

src\store\actionCreators\user.js

```
import { SET_USER_LIST, ADD_USER } from '../action-types';
import axios from 'axios';
const actions = {
  getUserList() {
    return function (dispatch, getState) {
      return axios.get('http://localhost:5000/api/users').then((response) => {
        const { data } = response.data;
        dispatch({
          type: SET_USER_LIST,
          payload: data
        });
      });
    }
  },
  addUser(user) {
    return { type: ADD_USER, payload: user }
  }
}
export default actions;
```

6.6 src\store\index.js

src\store\index.js

```
import { createStore, combineReducers, applyMiddleware } from 'redux'
import thunk from 'redux-thunk';
import promise from 'redux-promise';
import logger from 'redux-logger';
import counter from './reducers/counter';
+import user from './reducers/user';
export function getStore() {
+   const reducers = { counter, user }
    const combinedReducer = combineReducers(reducers);
    const store = applyMiddleware(thunk, promise, logger)(createStore)(combinedReducer);
    return store
}
```

6.7 src\routesConfig.js

src\routesConfig.js

```
import React from 'react';
import Home from './routes/Home';
import Counter from './routes/Counter';
+import User from './routes/User';
+import UserAdd from './routes/UserAdd';
+import UserList from './routes/UserList';
export default [
  {
    path: '/',
    element: ,
    index: true
  },
  {
    path: '/counter',
    element:
  },
+ {
+   path: '/user',
+   element: ,
+   children: [
+     {
+       path: '/user/List',
+       element: ,
+       index: true
+     },
+     {
+       path: '/user/Add',
+       element:
+     }
+   ]
+ }
]
```

6.8 User.js

src\routes\User.js

```
import React from 'react';
import { Link, Outlet } from 'react-router-dom';
function User() {
    return (
        <>
            <ul>
                <li><Link to="/user/add">UserAddLink>li>
                <li><Link to="/user/list">UserListLink>li>
            ul>
            <Outlet />
        </>
    )
}
export default User;
```

6.9 UserAdd.js

src\routes\UserAdd.js

```
import React, { useRef } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import { useNavigate } from 'react-router-dom';
import actionCreators from '@/store/actionCreators/user';
function UserAdd() {
    const list = useSelector(state => state.user.list);
    const nameRef = useRef();
    const navigate = useNavigate();
    const dispatch = useDispatch();
    const handleSubmit = (event) => {
        event.preventDefault();
        const name = nameRef.current.value;
        dispatch(actionCreators.addUser({ id: Date.now(), name }));
        navigate('/User/List');
    }
    return (
        <form onSubmit={handleSubmit}>
            用户名 <input ref={nameRef} />
            <input type="submit">input>
        form>
    )
}
export default UserAdd;
```

6.10 src\routes\UserList.js

src\routes\UserList.js

```
import React, { useEffect } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import actionCreators from '@/store/actionCreators/user';
function UserList() {
    const list = useSelector(state => state.user.list);
    const dispatch = useDispatch();
    useEffect(() => {
        if (list.length === 0) {
            dispatch(actionCreators.getUserList());
        }
    }, [])
    return (
        <ul>
            {
                list.map(user => <li key={user.id}>{user.name}li>)
            }
        ul>
    )
}
export default UserList;
```

src\components\Header\index.js

```
import React from 'react';
import { Link } from 'react-router-dom';
function Header() {
  return (

      Home
      Counter
+     User

  )
}
export default Header
```

7. 代理接口和服务器加载数据

- createStore.ts (https://github.com/reduxjs/redux/blob/master/src/createStore.ts)
- redux-thunk (https://github.com/reduxjs/redux-thunk/blob/master/src/index.ts)

7.1 安装

```
npm install express-http-proxy --save
```

7.2 src\server\index.js

src\server\index.js

```
import React from 'react';
import { renderToString } from 'react-dom/server';
import { StaticRouter } from "react-router-dom/server";
+import proxy from 'express-http-proxy';
import App from '../App';
+import { getServerStore } from '../store';
+import { matchRoutes } from 'react-router-dom';
+import routesConfig from '../routesConfig';
const express = require('express');
const app = express();
app.use(express.static('public'));
+app.use('/api', proxy('http://localhost:5000', {
+  proxyReqPathResolver(req) {
+    return `/api${req.url}`;
+  }
+}));
app.get('*', (req, res) => {
+ const routeMatches = matchRoutes(routesConfig, { pathname: req.url });
+  if (routeMatches) {
+   const store = getServerStore();
+   const promises = routeMatches
+     .map(({ route }) => route.element.type.loadData && route.element.type.loadData(store).then(data => data, error => error))
+     .concat(App.loadData && App.loadData(store))
+     .filter(Boolean)
+   Promise.all(promises).then(() => {
+     const html = renderToString(
+
+
+
+     );
      res.send(`

          ssr

        ${html}
+       </span>
<span class="hljs-addition">+         var context = {</span>
<span class="hljs-addition">+          state:${JSON.stringify(store.getState())}</span>
<span class="hljs-addition">+         }</span>
<span class="hljs-addition">+

  `);
    })
+ } else {
+   res.sendStatus(404);
+ }
});
app.listen(3000, () => console.log("server started on 3000"));
```

7.3 src\client\index.js

src\client\index.js

```
import React from 'react';
import { hydrateRoot } from 'react-dom/client';
import { BrowserRouter } from 'react-router-dom';
import { getClientStore } from '../store';
import App from '../App';
const root = document.getElementById('root');
+const store = getClientStore();
hydrateRoot(root,

+
  );
```

7.4 request.js

src\server\request.js

```
import axios from 'axios'
const request = axios.create({
  baseURL: 'http://localhost:5000/'
});
export default request
```

7.5 request.js

src\client\request.js

```
import axios from 'axios'
const request = axios.create({
  baseURI: '/'
});
export default request
```

7.6 UserList.js

src\routes\UserList.js

```
import React, { useEffect } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import actionCreators from '@/store/actionCreators/user';
function UserList() {
    const list = useSelector(state => state.user.list);
    const dispatch = useDispatch();
    useEffect(() => {
        if (list.length
            dispatch(actionCreators.getUserList());
        }
    }, [])
    return (

            {
                list.map(user => {user.name})
            }

    )
}
+UserList.loadData = (store) => {
+    return store.dispatch(actionCreators.getUserList());
+}
export default UserList;
```

7.7 src\store\actionCreators\user.js

src\store\actionCreators\user.js

```
import { SET_USER_LIST, ADD_USER } from '../action-types';
const actions = {
  getUserList() {
+   return function (dispatch, getState, request) {
+      return request.get('/api/users').then((response) => {
        const { data } = response.data;
        dispatch({
          type: SET_USER_LIST,
          payload: data
        });
      });
    }
  },
  addUser(user) {
    return { type: ADD_USER, payload: user }
  }
}
export default actions;
```

7.8 src\App.js

src\App.js

```
import React from 'react';
import { useRoutes } from 'react-router-dom';
import routesConfig from './routesConfig';
import Header from './components/Header';
import { Provider } from 'react-redux';
+function App({ store }) {
  return (

      {useRoutes(routesConfig)}

  )
}
export default App;
```

7.9 src\store\index.js

src\store\index.js

```
import { createStore, combineReducers, applyMiddleware } from 'redux'
import thunk from 'redux-thunk';
import promise from 'redux-promise';
import logger from 'redux-logger';
import counter from './reducers/counter';
import user from './reducers/user';
+import clientRequest from '@/client/request';
+import serverRequest from '@/server/request';
+const clientThunk = thunk.withExtraArgument(clientRequest);
+const serverThunk = thunk.withExtraArgument(serverRequest);
+const reducers = { counter, user }
+const combinedReducer = combineReducers(reducers);
+export function getClientStore() {
+    const initialState = window.context.state;
+    return applyMiddleware(clientThunk, promise, logger)(createStore)(combinedReducer, initialState);
+}
+export function getServerStore() {
+    return applyMiddleware(serverThunk, promise, logger)(createStore)(combinedReducer);
+}
```

8. 登录和权限

8.1 安装

```
npm install express-session --save
```

8.2 api.js

```
const express = require('express')
const cors = require('cors');
+const session = require('express-session');
+const app = express();
+app.use(cors());
+app.use(session({
+  saveUninitialized: true,
+  resave: true,
+  secret: 'zhufeng'
+}))
+app.use(express.json());
+app.use(express.urlencoded({ extended: true }));
const users = [{ id: 1, name: 'zhufeng1' }, { id: 2, name: 'zhufeng2' }, { id: 3, name: 'zhufeng3' }];
app.get('/api/users', (req, res) => {
  res.json({
    success: true,
    data: users
  });
});
+app.post('/api/login', (req, res) => {
+  const user = req.body;
+  req.session.user = user;
+  res.json({
+    success: true,
+    data: user
+  });
+});
+app.get('/api/logout', (req, res) => {
+  req.session.user = null;
+  res.json({
+    success: true
+  });
+});
+app.get('/api/user', (req, res) => {
+  const user = req.session.user;
+  if (user) {
+    res.json({
+      success: true,
+      data: user
+    });
+  } else {
+    res.json({
+      success: false,
+      error: '用户未登录'
+    });
+  }
+});
app.listen(5000, () => console.log('api server started on port 5000'));
```

8.3 src\routesConfig.js

src\routesConfig.js

```
import React from 'react';
import Home from './routes/Home';
import Counter from './routes/Counter';
import User from './routes/User';
import UserAdd from './routes/UserAdd';
import UserList from './routes/UserList';
+import Login from './routes/Login';
+import Logout from './routes/Logout';
+import Profile from './routes/Profile';
export default [
  {
    path: '/',
    element: ,
    index: true
  },
  {
    path: '/counter',
    element:
  },
  {
    path: '/user',
    element: ,
    children: [
      {
        path: '/user/List',
        element: ,
        index: true
      },
      {
        path: '/user/Add',
        element:
      }
    ]
  },
+ {
+   path: '/login',
+   element:
+ },
+ {
+   path: '/logout',
+   element:
+ },
+ {
+   path: '/profile',
+   element:
+ },
]
```

8.4 Login.js

src\routes\Login.js

```
import React, { useRef } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import actionCreators from '@/store/actionCreators/auth';
function Login() {
  const list = useSelector(state => state.user.list);
  const dispatch = useDispatch();
  const nameRef = useRef();
  const handleSubmit = (event) => {
    event.preventDefault();
    const name = nameRef.current.value;
    dispatch(actionCreators.login({ name }));
  }
  return (
    <form onSubmit={handleSubmit}>
      用户名 <input ref={nameRef} />
      <input type="submit">input>
    form>
  )
}
export default Login;
```

8.5 Logout.js

src\routes\Logout.js

```
import React from 'react';
import { useDispatch } from 'react-redux';
import actionCreators from '@/store/actionCreators/auth';
function Logout() {
  const dispatch = useDispatch();
  return (
    <button onClick={() => dispatch(actionCreators.logout())}>退出button>
  )
}
export default Logout;
```

8.6 Profile.js

src\routes\Profile.js

```
import React, { useEffect } from 'react';
import { useSelector } from 'react-redux';
import { useNavigate } from 'react-router-dom';
function Profile() {
  const user = useSelector(state => state.auth.user);
  const navigate = useNavigate();
  useEffect(() => {
    if (!user) {
      navigate('/login');
    }
  },[]);
  return <div>用户名:{user && user.name}div>
}
export default Profile;
```

8.7 action-types.js

src\store\action-types.js

```
export const ADD = 'ADD';

export const SET_USER_LIST = 'SET_USER_LIST';
export const ADD_USER = 'ADD_USER';

+export const LOGIN_SUCCESS = 'LOGIN_SUCCESS';
+export const LOGIN_ERROR = 'LOGIN_ERROR';
+export const LOGOUT_SUCCESS = 'LOGOUT_SUCCESS';
```

8.8 auth.js

src\store\reducers\auth.js

```
import { LOGIN_ERROR, LOGIN_SUCCESS, LOGOUT_SUCCESS } from '../action-types';
const initialState = { user: null, error: null }
function auth(state = initialState, action) {
  switch (action.type) {
    case LOGIN_SUCCESS:
      return { user: action.payload, error: null };
    case LOGIN_ERROR:
      return { user: null, error: action.payload };
    case LOGOUT_SUCCESS:
      return { user: null, error: null };
    default:
      return state;
  }
}
export default auth;
```

8.9 auth.js

src\store\actionCreators\auth.js

```

import { LOGIN_ERROR, LOGIN_SUCCESS, LOGOUT_SUCCESS } from '../action-types';
import { push } from 'redux-first-history';
const actionCreators = {
  login(user) {
    return function (dispatch, getState, request) {
      return request.post('/api/login', user).then(res => {
        const { success, data, error } = res.data;
        if (success) {
          dispatch({ type: LOGIN_SUCCESS, payload: data });
          dispatch(push('/profile'));
        } else {
          dispatch({ type: LOGIN_ERROR, payload: error });
        }
      });
    }
  },
  logout() {
    return function (dispatch, getState, request) {
      return request.get('/api/logout').then(res => {
        const { success } = res.data;
        if (success) {
          dispatch({ type: LOGOUT_SUCCESS });
          dispatch(push('/login'));
        }
      });
    }
  },
  validate() {
    return function (dispatch, getState, request) {
      return request.get('/api/validate').then(res => {
        const { success, data } = res.data;
        if (success) {
          dispatch({ type: LOGIN_SUCCESS, payload: data });
        }
      });
    }
  }
}

export default actionCreators;
```

8.10 src\store\index.js

src\store\index.js

```
import { createStore, combineReducers, applyMiddleware } from 'redux';
import thunk from 'redux-thunk';
import promise from 'redux-promise';
import logger from 'redux-logger'
import counter from './reducers/counter';
import user from './reducers/user';
+import auth from './reducers/auth';
import clientRequest from '@/client/request';
import serverRequest from '@/server/request';
+import { createBrowserHistory, createMemoryHistory } from 'history'
+import { createReduxHistoryContext } from 'redux-first-history';
export function getClientStore() {
  const initialState = window.context.state;
+  const { createReduxHistory, routerMiddleware, routerReducer } = createReduxHistoryContext({
+    history: createBrowserHistory()
+  });
+  const reducers = { counter, user, auth, router: routerReducer };
+  const combinedReducer = combineReducers(reducers);
+  const store = applyMiddleware(thunk.withExtraArgument(clientRequest), promise, routerMiddleware, logger)
+    (createStore)
+    (combinedReducer, initialState);
+  const history = createReduxHistory(store);
+  return { store, history }
}
export function getServerStore(req) {
+ const { createReduxHistory, routerMiddleware, routerReducer } = createReduxHistoryContext({
+   history: createMemoryHistory()
+ });
+ const reducers = { counter, user, auth, router: routerReducer };
+ const combinedReducer = combineReducers(reducers);
+ const store = applyMiddleware(thunk.withExtraArgument(serverRequest(req)), promise, routerMiddleware, logger)(createStore)(combinedReducer);
+  const history = createReduxHistory(store);
+  return { store, history }
}
```

8.11 src\App.js

src\App.js

```
import React from 'react';
import { useRoutes } from 'react-router-dom';
import routesConfig from './routesConfig';
import Header from './components/Header';
import { Provider } from 'react-redux';
+import actionCreators from './store/actionCreators/auth';
function App({ store }) {
  return (

      {useRoutes(routesConfig)}

  )
}
+App.loadData = (store) => {
+  return store.dispatch(actionCreators.validate())
+}
export default App;
```

8.12 server\index.js

src\server\index.js

```
app.get('*', (req, res) => {
  const routeMatches = matchRoutes(routesConfig, { pathname: req.url });
  if (routeMatches) {
+   const store = getServerStore(req);
    const promises = routeMatches
      .map(({ route }) => route.element.type.loadData && route.element.type.loadData(store).then(data => data, error => error))
      .concat(App.loadData && App.loadData(store))
      .filter(Boolean)
  }
}
```

8.13 request.js

src\server\request.js

```
import axios from 'axios'
const request = (req) => axios.create({
  baseURL: 'http://localhost:5000/',
+ headers: {
+   cookie: req.get('cookie') || ''
+ }
});
export default request
```

src\components\Header\index.js

```
import React from 'react';
import { Link } from 'react-router-dom';
+import { useSelector } from 'react-redux';
function Header() {
+ const { user } = useSelector(state => state.auth)
  return (

      Home
      Counter
      User
+     {
+       user ? (
+         <>
+           个人中心
+           退出
+         </>
+       ) : 登录
+     }

  )
}
export default Header
```

8.15 client\index.js

src\client\index.js

```
import React from 'react';
import { hydrateRoot } from 'react-dom/client';
+import { HistoryRouter as Router } from "redux-first-history/rr6";
import App from '@/App';
import { getClientStore } from '../store';
const root = document.getElementById('root');
const { store, history } = getClientStore();
hydrateRoot(root,
+

+
);
```

9. 状态码301和404

9.1 NotFound.js

src\routes\NotFound.js

```
import React from 'react';
function NotFound(props) {
    return (
        <div>NotFounddiv>
    )
}
export default NotFound;
```

9.2 src\routesConfig.js

src\routesConfig.js

```
import React from 'react';
import Home from './routes/Home';
import Counter from './routes/Counter';
import User from './routes/User';
import UserAdd from './routes/UserAdd';
import UserList from './routes/UserList';
import Login from './routes/Login';
import Logout from './routes/Logout';
import Profile from './routes/Profile';
+import NotFound from './routes/NotFound';
export default [
  {
    path: '/',
    element: ,
    index: true
  },
  {
    path: '/counter',
    element:
  },
  {
    path: '/user',
    element: ,
    children: [
      {
        path: '/user/List',
        element: ,
        index: true
      },
      {
        path: '/user/Add',
        element:
      }
    ]
  },
  {
    path: '/login',
    element:
  },
  {
    path: '/logout',
    element:
  },
  {
    path: '/profile',
    element:
  },
+ {
+   path: '*',
+   element:
+ }
]
```

9.3 src\server\index.js

src\server\index.js

```
import React from 'react';
import { renderToString } from 'react-dom/server';
import { StaticRouter } from "react-router-dom/server";
import proxy from 'express-http-proxy';
import App from '../App';
import { getServerStore } from '../store';
import { matchRoutes } from 'react-router-dom';
import routesConfig from '../routesConfig';
const express = require('express');
const app = express();
app.use(express.static('public'));
app.use('/api', proxy('http://localhost:5000', {
  proxyReqPathResolver(req) {
    return `/api${req.url}`;
  }
}));
app.get('*', (req, res) => {
  const routeMatches = matchRoutes(routesConfig, { pathname: req.url });
  if (routeMatches) {
    const store = getServerStore(req);
    const promises = routeMatches
      .map(({ route }) => route.element.type.loadData && route.element.type.loadData(store).then(data => data, error => error))
      .concat(App.loadData && App.loadData(store))
      .filter(Boolean)
    Promise.all(promises).then(() => {
+     if (req.url === '/profile' && (!(store.getState().auth.user))) {
+       return res.redirect('/login');
+     } else if (routeMatches[routeMatches.length - 1].route.path === '*') {
+       res.statusCode = 404;
+     }
      const html = renderToString(

      );
      res.send(`

          ssr

        ${html}

          var context = {
           state:${JSON.stringify(store.getState())}
          }

  `);
    })
  } else {
    res.sendStatus(404);
  }
});
app.listen(3000, () => console.log("server started on 3000"));
```

10. 支持CSS

10.1 安装

```
npm install css-loader isomorphic-style-loader-react18 --save
```

10.2 src\App.css

src\App.css

```
.color {
  color: red
}
```

10.3 src\App.js

src\App.js

```
import React from 'react';
import { useRoutes } from 'react-router-dom';
import routesConfig from './routesConfig';
import Header from './components/Header';
import { Provider } from 'react-redux';
import actionCreators from './store/actionCreators/auth';
+import useStyles from 'isomorphic-style-loader-react18/useStyles'
+import styles from './App.css'
function App({ store }) {
+ useStyles(styles);
  return (

      {useRoutes(routesConfig)}
+     red

  )
}
App.loadData = (store) => {
  return store.dispatch(actionCreators.validate())
}
export default App;
```

10.4 webpack.config.base.js

webpack.config.base.js

```
const path = require('path');
module.exports = {
    mode: 'development',
    resolve: {
        alias: {
            '@': path.resolve(__dirname, 'src')
        }
    },
    module: {
        rules: [
            {
                test: /\.js$/,
                enforce: 'pre',
                use: ['source-map-loader']
            },
            {
                test: /\.js/,
                use: {
                    loader: 'babel-loader',
                    options: {
                        presets: [
                            "@babel/preset-env",
                            "@babel/preset-react"
                        ]
                    }
                },
                exclude: /node_modules/
            },
+           {
+               test: /\.css$/,
+               use: [
+                   {
+                       loader: 'isomorphic-style-loader-react18'
+                   },
+                   {
+                       loader: 'css-loader',
+                       options: {
+                           modules: true
+                       }
+                   }
+               ]
+           }
+       ]
    }
}
```

10.5 src\server\index.js

src\server\index.js

```
import React from 'react';
import { renderToString } from 'react-dom/server';
import { StaticRouter } from "react-router-dom/server";
import proxy from 'express-http-proxy';
+import StyleContext from 'isomorphic-style-loader-react18/StyleContext'
import App from '../App';
import { getServerStore } from '../store';
import { matchRoutes } from 'react-router-dom';
import routesConfig from '../routesConfig';

const express = require('express');
const app = express();
app.use(express.static('public'));
app.use('/api', proxy('http://localhost:5000', {
  proxyReqPathResolver(req) {
    return `/api${req.url}`;
  }
}));
app.get('*', (req, res) => {
  const routeMatches = matchRoutes(routesConfig, { pathname: req.url });
  if (routeMatches) {
    const store = getServerStore(req);
    const promises = routeMatches
      .map(({ route }) => route.element.type.loadData && route.element.type.loadData(store).then(data => data, error => error))
      .concat(App.loadData && App.loadData(store))
      .filter(Boolean)
    Promise.all(promises).then(() => {
      if (req.url
        return res.redirect('/login');
      } else if (routeMatches[routeMatches.length - 1].route.path
        res.statusCode = 404;
      }
+     const css = new Set()
+     const insertCss = (...styles) => styles.forEach(style => {
+       css.add(style._getCss())
+     })
      const html = renderToString(

+


+


      );
      res.send(`

          ssr
+         ${[...css].join('')}

        ${html}

          var context = {
           state:${JSON.stringify(store.getState())}
          }

  `);
    })
  } else {
    res.sendStatus(404);
  }
});
app.listen(3000, () => console.log("server started on 3000"));
```

10.6 src\client\index.js

src\client\index.js

```
import React from 'react';
import { hydrateRoot } from 'react-dom/client';
import { BrowserRouter } from 'react-router-dom';
+import StyleContext from 'isomorphic-style-loader-react18/StyleContext'
import { getClientStore } from '../store';
import App from '../App';
const root = document.getElementById('root');
const store = getClientStore();
+const insertCss = (...styles) => {
+  const removeCss = styles.map(style => style._insertCss())
+  return () => removeCss.forEach(dispose => dispose())
+}
hydrateRoot(root,

+
+
+
  );
```

11. SEO

11.1 安装

```
npm install react-helmet --save
```

11.2 src\routes\Home.js

src\routes\Home.js

```
import React from 'react';
import { Helmet } from 'react-helmet';
function Home() {
  return (
    <>
      <Helmet>
        <title>首页标题title>
        <meta name="description" content="首页描述">meta>
      Helmet>
      <div>
        Home
      div>
    </>
  )
}
export default Home;
```

11.3 src\server\index.js

src\server\index.js

```
import React from 'react';
import { renderToString } from 'react-dom/server';
import { StaticRouter } from "react-router-dom/server";
import proxy from 'express-http-proxy';
import StyleContext from 'isomorphic-style-loader-react18/StyleContext'
+import { Helmet } from 'react-helmet';
import App from '../App';
import { getServerStore } from '../store';
import { matchRoutes } from 'react-router-dom';
import routesConfig from '../routesConfig';
const express = require('express');
const app = express();
app.use(express.static('public'));
app.use('/api', proxy('http://localhost:5000', {
  proxyReqPathResolver(req) {
    return `/api${req.url}`;
  }
}));
app.get('*', (req, res) => {
  const routeMatches = matchRoutes(routesConfig, { pathname: req.url });
  if (routeMatches) {
    const store = getServerStore(req);
    const promises = routeMatches
      .map(({ route }) => route.element.type.loadData && route.element.type.loadData(store).then(data => data, error => error))
      .concat(App.loadData && App.loadData(store))
      .filter(Boolean)
    Promise.all(promises).then(() => {
      if (req.url
        return res.redirect('/login');
      } else if (routeMatches[routeMatches.length - 1].route.path
        res.statusCode = 404;
      }
      const css = new Set()
      const insertCss = (...styles) => styles.forEach(style => {
        css.add(style._getCss())
      })
+     let helmet = Helmet.renderStatic();
      const html = renderToString(

      );

      res.send(`

+         ${helmet.title.toString()}
+         ${helmet.meta.toString()}
          ${[...css].join('')}

        ${html}

          var context = {
           state:${JSON.stringify(store.getState())}
          }

  `);
    })
  } else {
    res.sendStatus(404);
  }
});
app.listen(3000, () => console.log("server started on 3000"));
```

12. 流式SSR

12.1 user.js

src\store\actionCreators\user.js

```
import { SET_USER_LIST, ADD_USER } from '../action-types';
const actions = {
  getUserList() {
    return function (dispatch, getState, request) {
      return request.get('/api/users').then((response) => {
        const { data } = response.data;
        dispatch({
          type: SET_USER_LIST,
          payload: data
        });
+       return getState().user.list;
      });
    }
  },
  addUser(user) {
    return { type: ADD_USER, payload: user }
  }
}
export default actions;
```

12.2 server\index.js

src\server\index.js

```
import React from 'react';
+import { renderToPipeableStream } from 'react-dom/server';
import { StaticRouter } from "react-router-dom/server";
import proxy from 'express-http-proxy';
import StyleContext from 'isomorphic-style-loader-react18/StyleContext'
import { Helmet } from 'react-helmet';
import App from '../App';
import { getServerStore } from '../store';
import { matchRoutes } from 'react-router-dom';
import routesConfig from '../routesConfig';

const express = require('express');
const app = express();
app.use(express.static('public'));
app.use('/api', proxy('http://localhost:5000', {
  proxyReqPathResolver(req) {
    return `/api${req.url}`;
  }
}));
app.get('*', (req, res) => {
  const routeMatches = matchRoutes(routesConfig, { pathname: req.url });
  if (routeMatches) {
    const store = getServerStore(req);
    const promises = routeMatches
      .map(({ route }) => route.element.type.loadData && route.element.type.loadData(store).then(data => data, error => error))
      .concat(App.loadData && App.loadData(store))
      .filter(Boolean)
    Promise.all(promises).then(() => {
      if (req.url
        return res.redirect('/login');
      } else if (routeMatches[routeMatches.length - 1].route.path
        res.statusCode = 404;
      }
      const css = new Set()
      const insertCss = (...styles) => styles.forEach(style => {
        css.add(style._getCss())
      })
      let helmet = Helmet.renderStatic();
+     const { pipe } = renderToPipeableStream(

        ,
+        {
+          onShellReady() {
+            res.statusCode = 200;
+            res.setHeader('Content-type', 'text/html;charset=utf8');
+            res.write(`
+
+
+              ssr
+                ${helmet.title.toString()}
+                ${helmet.meta.toString()}
+              ${[...css].join('')}
+
+
+            `);
+            pipe(res);
+            res.write(`
+            </span>
<span class="hljs-addition">+              var context = {</span>
<span class="hljs-addition">+              state:${JSON.stringify(store.getState())}</span>
<span class="hljs-addition">+              }</span>
<span class="hljs-addition">+
+
+
+          `);
          }
        }
      );
    })
  } else {
    res.sendStatus(404);
  }
});
app.listen(3000, () => console.log("server started on 3000"));
```

12.3 src\routes\UserList.js

src\routes\UserList.js

```
+import React, { Suspense, useRef } from 'react';
import { useDispatch, useSelector } from 'react-redux';
import actionCreators from '@/store/actionCreators/user';
function UserList() {
+   const dispatch = useDispatch();
+   const resourceRef = useRef();
+   if (!resourceRef.current) {
+       const promise = dispatch(actionCreators.getUserList());
+       const resource = wrapPromise(promise);
+       resourceRef.current = resource;
+   }
+   return (
+       loading...}>
+
+
+   )
}
+function LazyList({ resource }) {
+    const userList = resource.read();
+    return (
+
+           {
+               userList.map(item => {item.name})
+           }
+
+    )
+}
+/*
+const promise = getUserList()
+const resource = wrapPromise(promise);
+function getUserList() {
+    return new Promise((resolve) => {
+        setTimeout(() => {
+            resolve([
+                { id: 1, name: 'zhufeng1' },
+                { id: 2, name: 'zhufeng2' },
+                { id: 3, name: 'zhufeng3' }
+            ])
+        }, 5000)
+    });
+}
+*/
+function wrapPromise(promise) {
+    let status = "pending";
+    let result;
+    let suspender = promise.then(
+        (r) => {
+            status = "success";
+            result = r;
+        },
+        (e) => {
+            status = "error";
+            result = e;
+        }
+    );
+    return {
+        read() {
+            if (status === "pending") {
+                throw suspender;
+            } else if (status === "error") {
+                throw result;
+            } else if (status === "success") {
+                return result;
+            }
+        }
+    };
+}
export default UserList;
```

12.4 api.js

api.js

```
const express = require('express')
const cors = require('cors');
const session = require('express-session');
const app = express();
app.use(cors());
app.use(session({
  saveUninitialized: true,
  resave: true,
  secret: 'zhufeng'
}))
app.use(express.json());
app.use(express.urlencoded({ extended: true }));
const users = [{ id: 1, name: 'zhufeng1' }, { id: 2, name: 'zhufeng2' }, { id: 3, name: 'zhufeng3' }];
app.get('/api/users', (req, res) => {
+ setTimeout(() => {
+   res.json({
+     success: true,
+     data: users
+   });
+ }, 5000);
});
app.post('/api/login', (req, res) => {
  const user = req.body;
  req.session.user = user;
  res.json({
    success: true,
    data: user
  });
});
app.get('/api/logout', (req, res) => {
  req.session.user = null;
  res.json({
    success: true
  });
});
app.get('/api/user', (req, res) => {
  const user = req.session.user;
  if (user) {
    res.json({
      success: true,
      data: user
    });
  } else {
    res.json({
      success: false,
      error: '用户未登录'
    });
  }
});
app.listen(5000, () => console.log('api server started on port 5000'));
```

13.参考

13.1 源码参考

- react-dom-server (https://zh-hans.reactjs.org/docs/react-dom-server.html)
- createStore.ts (https://github.com/reduxjs/redux/blob/master/src/createStore.ts)
- redux-thunk (https://github.com/reduxjs/redux-thunk/blob/master/src/index.ts)

13.2 水合

- 水合反应（ hydrated reaction），也叫作水化
- 是指物质溶解在水里时，与水发生的化学作用,水合分子的过程
- 组件在服务器端拉取数据(水)，并在服务器端首次渲染
- 脱水: 对组件进行脱水，变成HTML字符串，脱去动态数据，成为风干标本快照
- 注水：发送到客户端后，重新注入数据(水)，重新变成可交互组件


这个是晒干之后的银耳
呈自然的淡黄色