###### Webpack配置文件

//

const HtmlWebpackPlugin = require('html-webpack-plugin');

// 合并第三方库

const CommonsChunkPlugin = require("webpack/lib/optimize/CommonsChunkPlugin");

// 从js文件中分离css文件

const ExtractTextPlugin = require("extract-text-webpack-plugin");

// 分别指定每个样式表的输出文件名

const extractCSS = new ExtractTextPlugin("css/styles.css");

const extractLESS = new ExtractTextPlugin('css/less.css');

module.exports = {

entry: {

// 多入口文件

bundle1: \_\_dirname + '/src/js/app.jsx',

bundle2: \_\_dirname + '/src/js/main2.jsx',

// 需要合并的第三方库在这里指定

vendor: ['react', 'react-dom', 'react-router']

},

output: {

path: \_\_dirname + '/build',

filename: 'js/[name].js'

},

module: {

rules: [

{

test: /\.js[x]?$/,

exclude: /node\_modules/,

loader: 'babel-loader',

// 有.babelrc文件则不需要在这里设置

// query: {

// presets: ['es2015', 'react'],

// }

},

{

test: /\.css$/,

use: extractCSS.extract({

fallback: "style-loader",

use: "css-loader"

})

},

{

test: /\.less$/i,

use: extractLESS.extract([ 'css-loader', 'less-loader' ])

},

{

test: /\.(png|jpg)$/,

loader: "file-loader?name=img/[hash:8].[name].[ext]"

}

]

},

plugins: [

// 将bundle1和bundle2文件的相同代码提取到bundle文件中

// new CommonsChunkPlugin({

// name: 'common',

// chunks: ['bundle1', 'bundle2'],

// filename: 'js/bundle.js',

// }),

// 将使用到的全部第三方的库提取到同一个文件vendor

// new CommonsChunkPlugin({

// name: 'vendor',

// filename: 'js/vendor.js'

// }),

new HtmlWebpackPlugin({

template: \_\_dirname + "/src/index.html"

}),

// 分离css文件

extractCSS,

extractLESS,

],

devServer: {

contentBase: "./build/", // 本地服务器所加载的页面所在的目录

historyApiFallback: true, // 不跳转

inline: true // 实时刷新

}

}

index.html文件：

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<title>生活百科</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/animate.css/3.5.2/animate.min.css">

<link rel="stylesheet" href="css/bootstrap.css">

</head>

<body>

<div id="app"></div>

<script src="js/lib/jquery.js"></script>

<script src="js/lib/bootstrap.js"></script>

<script src="js/lib/bmob.js"></script>

</body>

</html>

###### app.jax

/\*\*

\* 程序主入口，包括引用，渲染组件。

\*/

import React from 'react';

import ReactDOM from 'react-dom';

import { createStore } from 'redux';

import { connect, Provider } from 'react-redux';

import Global from './Global.jsx';

import Action from './Action.jsx';

import BmobUtils from './util/bmobUtils.jsx';

import MainRouter from './component/MainRouter.jsx';

require('../css/style.css');

require('../css/less.less');

/\*\*

\* 初始化Bmob

\*/

Bmob.initialize(Global.APPLICATION\_ID, Global.REST\_API\_KEY);

/\*\*

\* Redux部分

\*/

/\*\*

\* reducer函数

\*/

const defaultState = {

currentUser: BmobUtils.getCurrentUser(),

currentUserBookCollection: [],

currentUserMusicCollection: [],

currentUserFilmCollection: [],

}

const reducer = (state = defaultState, action) => {

let obj = new Object();

let arr = new Array();

Object.assign(obj, state);

switch (action.type) {

case Action.LOGIN\_SUCCESS :

obj.currentUser = action.payload.currentUser;

return obj;

case Action.LOGOUT :

obj.currentUser = action.payload.currentUser;

obj.currentUserBookCollection = [];

obj.currentUserMusicCollection = [];

obj.currentUserFilmCollection = [];

return obj;

case Action.ADD\_BOOK\_COLLECTION :

Object.assign(arr, obj.currentUserBookCollection);

arr.push(action.payload.bookCollection);

obj.currentUserBookCollection = arr;

return obj;

case Action.REMOVE\_BOOK\_COLLECTION :

Object.assign(arr, obj.currentUserBookCollection);

for(let i=0; i<arr.length; i++){

if(arr[i].id == action.payload.bookCollectionId){

arr.splice(i, 1);

}

}

obj.currentUserBookCollection = arr;

return obj;

case Action.ADD\_MUSIC\_COLLECTION :

Object.assign(arr, obj.currentUserMusicCollection);

arr.push(action.payload.musicCollection);

obj.currentUserMusicCollection = arr;

return obj;

case Action.REMOVE\_MUSIC\_COLLECTION :

Object.assign(arr, obj.currentUserMusicCollection);

for(let i=0; i<arr.length; i++){

if(arr[i].id == action.payload.musicCollectionId){

arr.splice(i, 1);

}

}

obj.currentUserMusicCollection = arr;

return obj;

case Action.ADD\_FILM\_COLLECTION :

Object.assign(arr, obj.currentUserFilmCollection);

arr.push(action.payload.filmCollection);

obj.currentUserFilmCollection = arr;

return obj;

case Action.REMOVE\_FILM\_COLLECTION :

Object.assign(arr, obj.currentUserFilmCollection);

for(let i=0; i<arr.length; i++){

if(arr[i].id == action.payload.filmCollectionId){

arr.splice(i, 1);

}

}

obj.currentUserFilmCollection = arr;

return obj;

case Action.INIT\_BOOK\_COLLECTION :

arr = action.payload.bookCollection;

obj.currentUserBookCollection = arr;

return obj;

case Action.INIT\_MUSIC\_COLLECTION :

arr = action.payload.musicCollection;

obj.currentUserMusicCollection = arr;

return obj;

case Action.INIT\_FILM\_COLLECTION :

arr = action.payload.filmCollection;

obj.currentUserFilmCollection = arr;

return obj;

default :

return defaultState;

}

}

/\*\*

\* Action Creator

\*/

const getAction = (type, payload) => {

return {

type,

payload,

}

}

const store = createStore(reducer);

const render = () => {

ReactDOM.render(

<Provider store={store}>

<MainRouter/>

</Provider>

,

document.getElementById('app')

)

}

render();

store.subscribe(render);

export default store;

###### Global.jsx

/\*\*

\* 全局静态变量

\*/

const APPLICATION\_ID = 'cb3ba145277bf36c0e899c432cc6cce7';

const REST\_API\_KEY = '4962df6128294bb610ccb5d8307638d9';

const BOOK = 'book';

const MUSIC = 'music';

const FILM = 'film';

const BOOK\_CATEGORY = [

{ name: '童话', category: 'fairy\_tale', baseURL: 'https://api.douban.com/v2/book/search?tag=童话&count=5&' },

{ name: '小说', category: 'novel', baseURL: 'https://api.douban.com/v2/book/search?tag=小说&count=5&' },

{ name: '散文诗歌', category: 'prose\_poetry', baseURL: 'https://api.douban.com/v2/book/search?tag=散文诗歌&count=5&' },

{ name: '名人传记', category: 'biography', baseURL: 'https://api.douban.com/v2/book/search?tag=名人传记&count=5&' },

]

const MUSIC\_CATEGORY = [

{ name: '周杰伦', category: 'Jay\_Chou', baseURL: 'https://api.douban.com/v2/music/search?tag=周杰伦&count=5&' },

{ name: '陈奕迅', category: 'Eason\_Chan', baseURL: 'https://api.douban.com/v2/music/search?tag=陈奕迅&count=5&' },

{ name: '薛之谦', category: 'Joker\_Xue', baseURL: 'https://api.douban.com/v2/music/search?tag=薛之谦&count=5&' },

{ name: '容祖儿', category: 'Joey\_Yung', baseURL: 'https://api.douban.com/v2/music/search?tag=容祖儿&count=5&' },

{ name: '杨千嬅', category: 'Miriam\_Yeung', baseURL: 'https://api.douban.com/v2/music/search?tag=杨千嬅&count=5&' },

{ name: 'Adele', category: 'Adele', baseURL: 'https://api.douban.com/v2/music/search?tag=adele&count=5&' },

]

const FILM\_CATEGORY = [

{ name: '正在热映', category: 'in\_theaters', baseURL: 'https://api.douban.com/v2/movie/in\_theaters?count=5&' },

{ name: '即将上映', category: 'coming\_soon', baseURL: 'https://api.douban.com/v2/movie/coming\_soon?count=5&' },

{ name: 'TOP250', category: 'top250', baseURL: 'https://api.douban.com/v2/movie/top250?count=5&' },

]

const BOOK\_DETAILS\_BASE\_URL = 'https://api.douban.com/v2/book/';

const MUSIC\_DETAILS\_BASE\_URL = 'https://api.douban.com/v2/music/';

const FILM\_DETAILS\_BASE\_URL = 'https://api.douban.com/v2/movie/subject/';

const BOOK\_SEARCH\_BASE\_URL = 'https://api.douban.com/v2/book/search';

const MUSIC\_SEARCH\_BASE\_URL = 'https://api.douban.com/v2/music/search';

const FILM\_SEARCH\_BASE\_URL = 'https://api.douban.com/v2/movie/search';

const SINA\_LOACTION\_URL = 'http://int.dpool.sina.com.cn/iplookup/iplookup.php?format=js';

const BAIDU\_WEATCHER\_URL = 'http://api.map.baidu.com/telematics/v3/weather';

const BAIDU\_AK = 'UUKsMTP8kZUXnjAmcBuNkx2LbhldkTd7';

const Global = {

APPLICATION\_ID,

REST\_API\_KEY,

BOOK,

MUSIC,

FILM,

BOOK\_CATEGORY,

MUSIC\_CATEGORY,

FILM\_CATEGORY,

BOOK\_DETAILS\_BASE\_URL,

MUSIC\_DETAILS\_BASE\_URL,

FILM\_DETAILS\_BASE\_URL,

BOOK\_SEARCH\_BASE\_URL,

MUSIC\_SEARCH\_BASE\_URL,

FILM\_SEARCH\_BASE\_URL,

SINA\_LOACTION\_URL,

BAIDU\_WEATCHER\_URL,

BAIDU\_AK,

}

export default Global;

###### Action.jax

/\*\*

\* 全局的所有Action便利

\*/

const LOGIN\_SUCCESS = 0x1;

const LOGOUT = 0x2;

const ADD\_BOOK\_COLLECTION = 0x3;

const ADD\_MUSIC\_COLLECTION = 0x4;

const ADD\_FILM\_COLLECTION = 0x5;

const REMOVE\_BOOK\_COLLECTION = 0x6;

const REMOVE\_MUSIC\_COLLECTION = 0x7;

const REMOVE\_FILM\_COLLECTION = 0x8;

const INIT\_BOOK\_COLLECTION = 0x9;

const INIT\_MUSIC\_COLLECTION = 0x10;

const INIT\_FILM\_COLLECTION = 0x11;

const Action = {

LOGIN\_SUCCESS,

LOGOUT,

ADD\_BOOK\_COLLECTION,

ADD\_MUSIC\_COLLECTION,

ADD\_FILM\_COLLECTION,

REMOVE\_BOOK\_COLLECTION,

REMOVE\_MUSIC\_COLLECTION,

REMOVE\_FILM\_COLLECTION,

INIT\_BOOK\_COLLECTION,

INIT\_MUSIC\_COLLECTION,

INIT\_FILM\_COLLECTION,

}

export default Action;

###### bmobUtils.jsx

/\*\*

\* 定义Bmob的相关操作的工具类

\*/

/\*\*

\* 用户注册

\*/

const register = (username, password, email, success, fail) => {

let user = new Bmob.User();

user.set('username', username);

user.set('password', password);

user.set('email', email);

user.signUp(null, {

success: function(user) {

success(user);

},

error: function(user, e) {

fail(user, e);

}

});

}

/\*\*

\* 用户登录

\*/

const login= (username, password, success, fail) => {

Bmob.User.logIn(username, password, {

success: function(user) {

success(user);

},

error: function(user, error) {

fail(user, error);

}

});

}

/\*\*

\* 得到当前成功登录的用户

\*/

const getCurrentUser = () => {

let currentUser = Bmob.User.current();

return currentUser;

}

/\*\*

\* 注销用户

\*/

const logout = () => {

console.log(Bmob.User.logOut());

}

/\*\*

\* 得到当前用户的图书收藏

\*/

const getCurrentUserBookCollection = (success) => {

let currentUser = getCurrentUser();

if(!currentUser){

success([]);

return ;

}

let BookCollection = Bmob.Object.extend("BookCollection");

let query = new Bmob.Query(BookCollection);

query.equalTo("username", currentUser.attributes.username);

query.find({

success: (results) => {

if(results){

success(results);

}else{

success([]);

}

},

error: (error) => {

console.log(error);

}

})

}

/\*\*

\* 得到当前用户的音乐收藏

\*/

const getCurrentUserMusicCollection = (success) => {

let currentUser = getCurrentUser();

if(!currentUser){

success([]);

return ;

}

let MusicCollection = Bmob.Object.extend("MusicCollection");

let query = new Bmob.Query(MusicCollection);

query.equalTo("username", currentUser.attributes.username);

query.find({

success: (results) => {

if(results){

success(results);

}else{

success([]);

}

},

error: (error) => {

console.log(error);

}

})

}

/\*\*

\* 得到当前用户的电影收藏

\*/

const getCurrentUserFilmCollection = (success) => {

let currentUser = getCurrentUser();

if(!currentUser){

success([]);

return ;

}

let FilmCollection = Bmob.Object.extend("FilmCollection");

let query = new Bmob.Query(FilmCollection);

query.equalTo("username", currentUser.attributes.username);

query.find({

success: (results) => {

if(results){

success(results);

}else{

success([]);

}

},

error: (error) => {

console.log(error);

}

})

}

/\*\*

\* 添加Book收藏

\*/

const addBookCollection = (username, bookId, book, success, fail) => {

let BookCollection = Bmob.Object.extend("BookCollection");

let promise = new Promise((resolve, reject) => {

let query = new Bmob.Query(BookCollection);

query.equalTo("username", username);

query.equalTo("bookId", bookId);

query.find({

success: (results) => {

resolve(results);

},

error: (error) => {

console.log(error);

reject(error);

}

})

})

promise.then((results) => {

if(!results.length) {

let bookCollection = new BookCollection();

bookCollection.set('bookId', bookId);

bookCollection.set('username', username);

bookCollection.set('book', JSON.stringify(book));

bookCollection.save(null, {

success: (bookCollection) => {

success(bookCollection);

},

error: (bookCollection, error) => {

console.log(error);

fail(error);

},

})

}

})

.catch((error) => {

console.log('catch error:', error);

})

}

/\*\*

\* 删除Book收藏

\*/

const removeBookCollection = (bookCollection, success, fail) => {

bookCollection.destroy({

success: () => {

success();

},

error: (error) => {

console.log(error);

}

})

}

/\*\*

\* 添加Book收藏

\*/

const addMusicCollection = (username, musicId, music, success, fail) => {

let MusicCollection = Bmob.Object.extend("MusicCollection");

let promise = new Promise((resolve, reject) => {

let query = new Bmob.Query(MusicCollection);

query.equalTo("username", username);

query.equalTo("musicId", musicId);

query.find({

success: (results) => {

resolve(results);

},

error: (error) => {

console.log(error);

reject(error);

}

})

})

promise.then((results) => {

if(!results.length) {

let musicCollection = new MusicCollection();

musicCollection.set('musicId', musicId);

musicCollection.set('username', username);

musicCollection.set('music', JSON.stringify(music));

musicCollection.save(null, {

success: (musicCollection) => {

success(musicCollection);

},

error: (musicCollection, error) => {

console.log(error);

fail(error);

},

})

}

})

.catch((error) => {

console.log('catch error:', error);

})

}

/\*\*

\* 删除Music收藏

\*/

const removeMusicCollection = (musicCollection, success, fail) => {

musicCollection.destroy({

success: () => {

console.log('remove success');

success();

},

error: (error) => {

console.log(error);

}

})

}

/\*\*

\* 添加Film收藏

\*/

const addFilmCollection = (username, filmId, film, success, fail) => {

let FilmCollection = Bmob.Object.extend("FilmCollection");

let promise = new Promise((resolve, reject) => {

let query = new Bmob.Query(FilmCollection);

query.equalTo("username", username);

query.equalTo("filmId", filmId);

query.find({

success: (results) => {

resolve(results);

},

error: (error) => {

console.log(error);

reject(error);

}

})

})

promise.then((results) => {

if(!results.length) {

let filmCollection = new FilmCollection();

filmCollection.set('filmId', filmId);

filmCollection.set('username', username);

filmCollection.set('film', JSON.stringify(film));

filmCollection.save(null, {

success: (filmCollection) => {

success(filmCollection);

},

error: (filmCollection, error) => {

console.log(error);

fail(error);

},

})

}

})

.catch((error) => {

console.log('catch error:', error);

})

}

/\*\*

\* 删除Film收藏

\*/

const removeFilmCollection = (filmCollection, success, fail) => {

filmCollection.destroy({

success: () => {

success();

},

error: (error) => {

console.log(error);

}

})

}

/\*\*

\* 获取轮播图数据

\*/

const fetchCarouselData = (success) => {

let Carousel = Bmob.Object.extend("Carousel");

let query = new Bmob.Query(Carousel);

query.find({

success: (results) => {

success(results);

},

error: (error) => {

}

})

}

const BmobUtils = {

register,

login,

logout,

getCurrentUser,

addBookCollection,

removeBookCollection,

addMusicCollection,

removeMusicCollection,

addFilmCollection,

removeFilmCollection,

getCurrentUserBookCollection,

getCurrentUserMusicCollection,

getCurrentUserFilmCollection,

fetchCarouselData,

}

export default BmobUtils;

###### utils.jsx

import $ from 'jquery';

import Global from '../Global.jsx';

const uuid = () => {

let i, random;

let uuid = '';

for (i = 0; i < 32; i++) {

random = Math.random() \* 16 | 0;

if (i === 8 || i === 12 || i === 16 || i === 20) {

uuid += '-';

}

uuid += (i === 12 ? 4 : (i === 16 ? (random & 3 | 8) : random))

.toString(16);

}

return uuid;

}

/\*转换前：{id: '001',name: 'taoyige', age: 22} (obj)

转换后：id=001&name=taoyige&age=22

\*/

const obj2uri = (obj) => {

return Object.keys(obj).map(function(k) {

return encodeURIComponent(k) + '=' + encodeURIComponent(obj[k]);

}).join('&');

}

/\*\*

\* 异步jsonp请求

\*/

const $jsonp = (url, data, callback) => {

// 生成唯一函数名

var callbackFuncName = 'MY\_JSONP\_CALLBACK' + Math.random().toString().replace('.', '');

var params = '';

// 先判断url有没有自带?

if (url.indexOf('?') == -1) {

params += '?';

}

// 拼接参数

for (var key in data) {

params += key + '=' + data[key] + '&';

}

url += params + 'callback' + '=' + callbackFuncName;

// 创建script节点

var script = document.createElement('script');

// 1. 挂载方法到window

// 每次执行完callback后都需要将该script对象删除

window[callbackFuncName] = function (data) {

callback(data);

document.body.removeChild(script);

};

// 插入数据

script.src = url;

// 插入节点

document.body.appendChild(script);

}

/\*\*

\* 根据url获取数据

\*/

const fetchData = (url, data, callback) => {

$jsonp(url, data, callback);

}

/\*\*

\* 获取当前地理位置的天气

\*/

const fetchLocalWeather = (success) => {

let promise = new Promise((resolve, reject) => {

$.getScript(Global.SINA\_LOACTION\_URL, function(\_result) {

if (remote\_ip\_info.ret == '1') {

resolve(remote\_ip\_info.city);

} else {

reject('没有找到匹配的IP地址信息！');

}

});

})

promise.then((city)=> {

fetchDestinationWeather(city, success);

}).catch((error) => {

console.log(error);

});

}

/\*\*

\* 获取指定地点天气

\*/

const fetchDestinationWeather = (city, success) => {

fetchData(Global.BAIDU\_WEATCHER\_URL,

{ location:city, output:'json', ak:Global.BAIDU\_AK },

(data) => {

console.log(data);

if(data.error === 0)

success(data);

});

}

export {uuid, obj2uri, fetchData, fetchLocalWeather, fetchDestinationWeather};

###### App.jsx

/\*\*

\* 整个应用的父组件，包括了导航条，主体内容和脚注

\*/

import React from 'react';

import MainRouter from './MainRouter.jsx';

import Footer from './Footer.jsx';

import VisibleMain from './Main.jsx';

class App extends React.Component {

render () {

return (

<VisibleMain/>

)

}

}

export default App;

###### BackToTop.jsx

import React from 'react';

class BackToTop extends React.Component {

handleClickBackToTop (e) {

document.body.scrollTop = 0;

}

render () {

return (

<div id="backToTop" className="back-to-top hidden-sm hidden-xs">

<a href="javascript:;" title="回到顶部" onClick={this.handleClickBackToTop}>

<span className="glyphicon glyphicon-chevron-up" aria-hidden="true"></span>

</a>

</div>

)

}

}

export default BackToTop;

###### Book.jsx

import React from 'react';

import { Link } from 'react-router';

import $ from 'jquery';

import SubNavigation from './SubNavigation.jsx';

import Global from '../Global.jsx';

import { fetchData } from '../util/utils.jsx';

import Footer from './Footer.jsx';

import BackToTop from './BackToTop.jsx';

import LoadMore from './LoadMore.jsx';

import book1 from '../../json/book1.json';

import book2 from '../../json/book2.json';

import book3 from '../../json/book3.json';

import book4 from '../../json/book4.json';

const data = [book1, book2, book3, book4];

class Book extends React.Component {

constructor(props) {

super(props);

this.loadMoreTop = 0;

this.start = 0;

this.state = {

books: [],

};

}

componentWillMount () {

this.initBooks();

}

componentWillReceiveProps (nextProps) {

this.initBooks();

// this.initLoadMore();

}

initBooks () {

let category = this.props.params.category;

for(let i=0; i<Global.BOOK\_CATEGORY.length; i++){

let item = Global.BOOK\_CATEGORY[i];

if(item.category == category){

// fetchData(`${item.baseURL}&start=${this.start}&`, {}, (data) => {

// this.setState({

// books: data.books

// })

// });

this.setState({

books: data[i].books

})

}

}

}

componentDidMount () {

this.initLoadMore();

}

componentDidUpdate(prevProps, prevState){

this.loadMoreTop = $('#loadMore').offset().top;

}

initLoadMore () {

let that = this;

let height = $(window).height();

let isLoadMore = true;

$(window).on('scroll', (e) => {

if(isLoadMore) {

if(this.loadMoreTop - $(window).scrollTop() <= height) {

isLoadMore = false;

setTimeout(() => {

// let category = this.props.params.category;

// for(let i=0; i<Global.BOOK\_CATEGORY.length; i++){

// let item = Global.BOOK\_CATEGORY[i];

// if(item.category == category){

// fetchData(Global.BOOK\_CATEGORY[i].baseURL, {start: this.start}, (data) => {

// let arr = that.state.books;

// arr.push(...data.books);

// this.setState({

// books: arr

// })

// this.start += 5;

// });

// }

// }

let category = this.props.params.category;

for(let i=0; i<Global.BOOK\_CATEGORY.length; i++){

let item = Global.BOOK\_CATEGORY[i];

if(item.category == category){

let arr = that.state.books;

arr.push(data[i].books[this.start++ % 4]);

this.setState({

books: arr

})

}

}

isLoadMore = true;

}, 1000)

}

}

})

}

render () {

return (

<div className="container">

<div className="row">

<SubNavigation category={ Global.BOOK } list={ Global.BOOK\_CATEGORY }/>

<div className="col-xs-12 col-sm-9 col-sm-offset-2 col-md-8 col-md-offset-1 list-content">

<ul>

{

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

let tags = '';

for(let i=0; i<item.tags.length; i++){

tags += item.tags[i].name + ' ';

}

return (

<li key={index} className="list-item">

<div className="media">

<div className="media-left">

<Link to={`/book\_details/${item.id}`}>

<img className="media-object dd" src={item.image} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/book\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">标签:{tags}</p>

<p>作者:{item.author}</p>

<p>出版社:{item.publisher}</p>

</div>

</div>

</li>

)

})

}

</ul>

<LoadMore/>

</div>

<Footer/>

<BackToTop/>

</div>

</div>

)

}

}

export default Book;

###### BookDetails.jsx

import React from 'react';

import { fetchData } from '../util/utils.jsx';

import { connect } from 'react-redux';

import BmobUtils from '../util/bmobUtils.jsx';

import Global from '../Global.jsx';

import Action from '../Action.jsx';

class BookDetails extends React.Component {

constructor(props) {

super(props);

this.state = {

book: {

rating: {

numRaters: 0,

},

title: '',

images: {

large: '',

},

author: [],

tags: [],

},

}

this.handleCollectionClick = this.handleCollectionClick.bind(this);

this.handleCancelCollectionClick = this.handleCancelCollectionClick.bind(this);

}

componentWillMount () {

let that = this;

let id = this.props.params.id;

let url = Global.BOOK\_DETAILS\_BASE\_URL + id;

fetchData(url, {}, (data) => {

if(that.refs.myRef)

that.setState({

book: data,

})

});

}

handleCollectionClick () {

this.props.onCollectionClick(this.props.currentUser, this.state.book, this.props.currentUserBookCollection);

}

handleCancelCollectionClick () {

this.props.onCancelCollectionClick(this.props.currentUser, this.props.currentUserBookCollection);

}

render () {

let book = this.state.book;

let author = '';

for(let i=0; i<book.author.length; i++){

author += book.author[i] + ' ';

}

let tags = '';

for(let i=0; i<book.tags.length; i++){

tags += book.tags[i].name + ' ';

}

let hasCollection = false;

for(let i=0; i<this.props.currentUserBookCollection.length; i++){

if(this.props.currentUserBookCollection[i].attributes.bookId == this.props.params.id){

hasCollection = true;

}

}

return (

<div className="container details" ref="myRef">

<div className="details-jumbotron">

<div className="media">

<div className="media-left">

<a href="#">

<img className="media-object" src={book.images.large} alt={book.title}></img>

</a>

</div>

<div className="media-body">

<span className="badge badge-details">

<span className="glyphicon glyphicon-thumbs-up" aria-hidden="true"></span>

<span title="收藏数">&nbsp;{book.rating.numRaters}</span>

</span>

<h1>{book.title}</h1>

<p><span className="text-bold">标签：</span>{tags}</p>

<p><span className="text-bold">作者：</span>{author}</p>

<p><span className="text-bold">出版日期：</span>{book.pubdate}</p>

<p><span className="text-bold">总页数：</span>{book.pages}</p>

<p><span className="text-bold">售价：</span>{book.price}</p>

<p><span className="text-bold">出版社：</span>{book.publisher}</p>

<p><span className="text-bold">摘要：</span>{book.summary}</p>

<button style={{display:hasCollection==false?'inline-block':'none'}} onClick={this.handleCollectionClick} className="btn btn-focus btn-lg">

收&nbsp;&nbsp;藏

</button>

<button style={{display:hasCollection==true?'inline-block':'none'}} onClick={this.handleCancelCollectionClick} className="btn btn-default btn-lg">

取消收藏

</button>

</div>

</div>

</div>

</div>

)

}

}

const mapStateToProps = (state) => {

return {

currentUser: state.currentUser,

currentUserBookCollection: state.currentUserBookCollection,

}

}

const mapDispatchToProps = (dispatch, ownProps) => {

return {

onCollectionClick: (currentUser, book, currentUserBookCollection) => {

if(currentUser){

if(currentUserBookCollection.indexOf(ownProps.params.id) == -1){

BmobUtils.addBookCollection(currentUser.attributes.username, book.id, book, (bookCollection) => {

dispatch({

type: Action.ADD\_BOOK\_COLLECTION,

payload: {

bookCollection: bookCollection

}

})

}, (error) => {

console.log(error);

});

}

}

},

onCancelCollectionClick: (currentUser, currentUserBookCollection) => {

if(currentUser){

let bookCollection = null;

for(let i=0; i<currentUserBookCollection.length; i++){

if(currentUserBookCollection[i].attributes.bookId == ownProps.params.id){

bookCollection = currentUserBookCollection[i];

}

}

BmobUtils.removeBookCollection(bookCollection, () => {

dispatch({

type: Action.REMOVE\_BOOK\_COLLECTION,

payload: {

bookCollectionId: bookCollection.id

}

})

}, (error) => {

console.log(error);

});

}

}

}

}

const VisibleBookDetails = connect(

mapStateToProps,

mapDispatchToProps

)(BookDetails);

export default VisibleBookDetails;

###### Film.jsx

import React from 'react';

import { Link } from 'react-router';

import SubNavigation from './SubNavigation.jsx';

import Global from '../Global.jsx';

import { fetchData } from '../util/utils.jsx';

import Footer from './Footer.jsx';

import BackToTop from './BackToTop.jsx';

import LoadMore from './LoadMore.jsx';

import film1 from '../../json/film1.json';

import film2 from '../../json/film2.json';

import film3 from '../../json/film3.json';

const data = [film1, film2, film3];

class Film extends React.Component {

constructor(props) {

super(props);

this.loadMoreTop = 0;

this.start = 0;

this.state = {

films: [],

};

}

componentWillMount () {

this.initFilms();

}

componentWillReceiveProps () {

this.initFilms();

}

initFilms () {

let category = this.props.params.category;

for(let i=0; i<Global.FILM\_CATEGORY.length; i++){

let item = Global.FILM\_CATEGORY[i];

if(item.category == category){

// fetchData(item.baseURL, {}, (data) => {

// console.log(data);

// this.setState({

// films: data.subjects

// })

// });

this.setState({

films: data[i].subjects

})

}

}

}

componentDidMount () {

this.initLoadMore();

}

componentDidUpdate(prevProps, prevState){

this.loadMoreTop = $('#loadMore').offset().top;

}

initLoadMore () {

let that = this;

let height = $(window).height();

let isLoadMore = true;

$(window).on('scroll', (e) => {

if(isLoadMore) {

if(this.loadMoreTop - $(window).scrollTop() <= height) {

isLoadMore = false;

setTimeout(() => {

// let category = this.props.params.category;

// for(let i=0; i<Global.FILM\_CATEGORY.length; i++){

// let item = Global.FILM\_CATEGORY[i];

// if(item.category == category){

// fetchData(Global.FILM\_CATEGORY[i].baseURL, {start: this.start}, (data) => {

// let arr = that.state.films;

// arr.push(...data.subjects);

// this.setState({

// films: arr

// })

// this.start += 5;

// });

// }

// }

let category = this.props.params.category;

for(let i=0; i<Global.FILM\_CATEGORY.length; i++){

let item = Global.FILM\_CATEGORY[i];

if(item.category == category){

let arr = that.state.films;

arr.push(data[i].subjects[this.start++ % 4]);

this.setState({

films: arr

})

}

}

isLoadMore = true;

}, 1000)

}

}

})

}

render () {

return (

<div className="container">

<div className="row">

<SubNavigation category={ Global.FILM } list={ Global.FILM\_CATEGORY }/>

<div className="col-xs-12 col-sm-9 col-sm-offset-2 col-md-8 col-md-offset-1 list-content">

<ul>

{

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

let casts = '';

for(let i=0; i<item.casts.length; i++){

casts += item.casts[i].name + ' ';

}

let directors = '';

for(let i=0; i<item.directors.length; i++){

directors += item.directors[i].name + ' ';

}

return (

<li key={index} className="list-item">

<div className="media">

<div className="media-left">

<Link to={`/film\_details/${item.id}`}>

<img className="media-object dd" src={item.images.small} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/film\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">年份:{item.year}</p>

<p>演员:{casts}</p>

<p>导演:{directors}</p>

</div>

</div>

</li>

)

})

}

</ul>

<LoadMore/>

</div>

<Footer/>

<BackToTop/>

</div>

</div>

)

}

}

export default Film;

###### FilmDetails.jsx

import React from 'react';

import { fetchData } from '../util/utils.jsx';

import { connect } from 'react-redux';

import BmobUtils from '../util/bmobUtils.jsx';

import Global from '../Global.jsx';

import Action from '../Action.jsx';

class FilmDetails extends React.Component {

constructor(props) {

super(props);

this.state = {

film: {

reviews\_count: 0,

wish\_count: 0,

year: '',

images: { large: '' },

title: '',

countries: [],

genres: [],

casts: [],

summary: '',

directors: [ { name: '' } ],

ratings\_count: 0,

},

};

this.handleCollectionClick = this.handleCollectionClick.bind(this);

this.handleCancelCollectionClick = this.handleCancelCollectionClick.bind(this);

}

componentWillMount () {

let that = this;

let id = this.props.params.id;

let url = Global.FILM\_DETAILS\_BASE\_URL + id;

fetchData(url, {}, (data) => {

if(that.refs.myRef)

that.setState({

film: data,

})

});

}

handleCollectionClick () {

this.props.onCollectionClick(this.props.currentUser, this.state.film, this.props.currentUserFilmCollection);

}

handleCancelCollectionClick () {

this.props.onCancelCollectionClick(this.props.currentUser, this.props.currentUserFilmCollection);

}

render () {

let film = this.state.film;

let casts = '';

for(let i=0; i<film.casts.length; i++){

casts += film.casts[i].name + ' ';

}

let genres = '';

for(let i=0; i<film.genres.length; i++){

genres += film.genres[i] + ' ';

}

let hasCollection = false;

for(let i=0; i<this.props.currentUserFilmCollection.length; i++){

if(this.props.currentUserFilmCollection[i].attributes.filmId == this.props.params.id){

hasCollection = true;

}

}

return (

<div className="container details" ref="myRef">

<div className="details-jumbotron">

<div className="media">

<div className="media-left">

<a href="#">

<img className="media-object" src={film.images.large} alt={film.title}></img>

</a>

</div>

<div className="media-body">

<span className="badge badge-details">

<span className="glyphicon glyphicon-thumbs-up" aria-hidden="true"></span>

<span title="收藏数">&nbsp;{film.ratings\_count}</span>

</span>

<h1>{film.title}</h1>

<p><span className="text-bold">导演：</span>{film.directors[0].name}</p>

<p><span className="text-bold">演员：</span>{casts}</p>

<p><span className="text-bold">上映日期：</span>{film.year}</p>

<p><span className="text-bold">类型：</span>{genres}</p>

<p><span className="text-bold">摘要：</span>{film.summary}</p>

<button style={{display:hasCollection==false?'inline-block':'none'}} onClick={this.handleCollectionClick} className="btn btn-focus btn-lg">

收&nbsp;&nbsp;藏

</button>

<button style={{display:hasCollection==true?'inline-block':'none'}} onClick={this.handleCancelCollectionClick} className="btn btn-default btn-lg">

取消收藏

</button>

</div>

</div>

</div>

</div>

)

}

}

const mapStateToProps = (state) => {

return {

currentUser: state.currentUser,

currentUserFilmCollection: state.currentUserFilmCollection,

}

}

const mapDispatchToProps = (dispatch, ownProps) => {

return {

onCollectionClick: (currentUser, film, currentUserFilmCollection) => {

if(currentUser){

if(currentUserFilmCollection.indexOf(ownProps.params.id) == -1){

BmobUtils.addFilmCollection(currentUser.attributes.username, film.id, film, (filmCollection) => {

dispatch({

type: Action.ADD\_FILM\_COLLECTION,

payload: {

filmCollection: filmCollection

}

})

}, (error) => {

console.log(error);

});

}

}

},

onCancelCollectionClick: (currentUser, currentUserFilmCollection) => {

if(currentUser){

let filmCollection = null;

for(let i=0; i<currentUserFilmCollection.length; i++){

if(currentUserFilmCollection[i].attributes.filmId == ownProps.params.id){

filmCollection = currentUserFilmCollection[i];

}

}

BmobUtils.removeFilmCollection(filmCollection, () => {

dispatch({

type: Action.REMOVE\_FILM\_COLLECTION,

payload: {

filmCollectionId: filmCollection.id

}

})

}, (error) => {

console.log(error);

});

}

}

}

}

const VisibleFilmDetails = connect(

mapStateToProps,

mapDispatchToProps

)(FilmDetails);

export default VisibleFilmDetails;

###### Footer.jsx

/\*\*

\* 脚注部分

\*/

import React from 'react';

class Footer extends React.Component {

render () {

return (

<div className="col-sm-3 col-md-2 list-content col-md-offset-8 footer-fixed hidden-xs hidden-sm">

<div className="friends-link module">

<div className="link-head">

友情链接

</div>

<div className="link-conent clearfix">

<ul>

<li><a href="http://www.zhku.edu.cn/">仲恺农业工程学院</a></li>

<li><a href="http://www.ruanyifeng.com/blog/2016/09/react-technology-stack.html">阮一峰-React 技术栈系列教程</a></li>

<li><a href="http://es6.ruanyifeng.com/">阮一峰-ECMAScript 6 入门</a></li>

<li><a href="https://github.com/facebook/react">React</a></li>

<li><a href="https://reacttraining.com/react-router/">React-Router</a></li>

<li><a href="http://redux.js.org/">Redux</a></li>

<li><a href="http://webpack.github.io/">Webpack</a></li>

<li><a href="http://babeljs.io/">Babel</a></li>

<li><a href="http://www.bootcss.com/">Bootstrap</a></li>

<li><a href="http://jquery.com/">JQuery</a></li>

<li><a href="http://nodejs.cn/">NodeJS</a></li>

<li><a href="https://www.npmjs.com/">NPM</a></li>

</ul>

</div>

</div>

<div className="module">

<div className="link-head">

GitHub

</div>

<div className="link-conent clearfix">

<ul>

<li><a href="https://github.com/taoyige/HouseholdEncyclopedia">GitHub项目仓库</a></li>

<li><a href="http://taoyige.github.io/HouseholdEncyclopedia/build">GitHub项目代理</a></li>

</ul>

</div>

</div>

</div>

)

}

}

export default Footer;

###### Home.jsx

import React from 'react';

import { Link } from 'react-router';

import $ from 'jquery';

import BmobUtils from '../util/bmobUtils.jsx';

import { fetchData, fetchLocalWeather, fetchDestinationWeather } from './../util/utils.jsx';

class Home extends React.Component {

constructor(props) {

super(props);

this.state = {

toggle\_on: false,

carousel\_list: [],

weather\_info: {

date: '1997-01-01',

results: [

{

currentCity: '',

index: [],

pm25: '-1',

weather\_data: [],

}

]

},

};

this.handleToggleClick = this.handleToggleClick.bind(this);

this.handleWeatherSearchSubmit = this.handleWeatherSearchSubmit.bind(this);

}

componentWillMount () {

this.initData();

}

componentWillReceiveProps () {

this.initData();

}

initData () {

BmobUtils.fetchCarouselData((results) => {

this.setState({

carousel\_list: results

})

});

fetchLocalWeather((data) => {

this.setState({

weather\_info: data,

})

});

}

handleToggleClick () {

this.setState({

toggle\_on: !this.state.toggle\_on,

})

if(this.state.toggle\_on){

$('.weather\_detail').removeClass('animated fadeInDown');

$('.weather\_detail').addClass('animated zoomOut');

}else {

$('.weather\_detail').removeClass('animated zoomOut');

$('.weather\_detail').addClass('animated fadeInDown');

}

}

handleWeatherSearchSubmit (e) {

e.preventDefault();

let city = this.refs.cityName.value;

if(city) {

fetchDestinationWeather(city, (data) => {

this.setState({

weather\_info: data,

})

this.refs.cityName.value = '';

})

}

}

render () {

let list = this.state.carousel\_list;

return (

<div className="container home">

<div className="row weather">

<div className="col-xs-1 col-md-1 weather-header">

<p className="text-center">{this.state.weather\_info.results[0].currentCity}天气</p>

<p className="text-center">PM2.5：{this.state.weather\_info.results[0].pm25}</p>

</div>

{

this.state.weather\_info.results[0].weather\_data.map((item, index) => {

return (

<div className="col-xs-2 col-md-2 blk\_fc\_c0\_i" key={index}>

<p className="wt\_fc\_c0\_i\_date">{item.date.split(' ')[0]}</p>

<p className="wt\_fc\_c0\_i\_weather">{item.weather}</p>

<p className="wt\_fc\_c0\_i\_wind">{item.wind}</p>

<p className="wt\_fc\_c0\_i\_temp">{item.temperature}</p>

</div>

)

})

}

<form onSubmit={this.handleWeatherSearchSubmit} className="weather-search">

<input ref='cityName' type="text" placeholder='国内城市'></input>

</form>

<div className="toggle" onClick={this.handleToggleClick}>

<span className={this.state.toggle\_on==false?'glyphicon glyphicon-chevron-down':'glyphicon glyphicon-chevron-up'} aria-hidden="true"></span>

</div>

</div>

<div className='row weather weather\_detail'>

<div className="col-xs-1 col-md-1"></div>

{

this.state.weather\_info.results[0].index.map((item, index) => {

return (

<div className="col-xs-2 col-md-2 blk\_fc\_c0\_i" key={index}>

<p className="wt\_fc\_c0\_i\_tipt">{item.tipt}</p>

<p className="wt\_fc\_c0\_i\_title\_zs">{item.title}：{item.zs}</p>

<p className="wt\_fc\_c0\_i\_des">{item.des}</p>

</div>

)

})

}

</div>

<div id="myCarousel" className="carousel slide row">

<ol className="carousel-indicators">

{

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

return (

<li key={index} data-target="#myCarousel" data-slide-to={index} className={index==0?'active':''}></li>

)

})

}

</ol>

<div className="carousel-inner">

{

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

return (

<div className={index==0?'item active': 'item'} key={index}>

<Link to={item.attributes.link}>

<img className="lunbo" src={item.attributes.url} alt="Third slide"></img>

</Link>

</div>

)

})

}

</div>

<a className="carousel-control left" href="#myCarousel"

data-slide="prev"><span className="glyphicon glyphicon-chevron-left" aria-hidden="true"></span>

</a>

<a className="carousel-control right" href="#myCarousel"

data-slide="next"><span className="glyphicon glyphicon-chevron-right" aria-hidden="true"></span>

</a>

</div>

</div>

)

}

}

export default Home;

###### LoadMore.jsx

import React from 'react';

class LoadMore extends React.Component {

render () {

return (

<div id="loadMore">

<div className="spinner">

<div className="rect1"></div>

<div className="rect2"></div>

<div className="rect3"></div>

<div className="rect4"></div>

<div className="rect5"></div>

</div>

</div>

)

}

}

export default LoadMore;

###### Login.jsx

import React from 'react';

import BmobUtils from '../util/bmobUtils.jsx';

import { connect } from 'react-redux';

import Action from '../Action.jsx';

class Login extends React.Component {

constructor(props) {

super(props);

this.state = {

loginInfo: '',

isLogining: false,

};

this.handleCloseClick = this.handleCloseClick.bind(this);

this.handleLoginSubmit = this.handleLoginSubmit.bind(this);

}

shouldComponentUpdate () {

this.clearInput();

return true

}

clearInput () {

this.refs.username.value = '';

this.refs.password.value = '';

this.setState({

loginInfo: ''

})

}

handleCloseClick (e) {

e.preventDefault();

this.props.handleCloseClick();

}

handleLoginSubmit (e) {

e.preventDefault();

let username = this.refs.username.value;

let password = this.refs.password.value;

let loginInfo = '';

if(this.checkInputUsername(username) && this.checkInputPassword(password)) {

loginInfo = '正在登录...';

this.setState({

isLogining: true,

})

this.login(username, password);

}else if(!this.checkInputUsername(username)) {

loginInfo = '用户名不合法！';

}else if(!this.checkInputPassword(password)) {

loginInfo = '密码不合法！';

}

this.setState({

loginInfo: loginInfo,

})

}

/\*\*

\* 检查输入的用户名合法性

\*/

checkInputUsername (username) {

if(username.match(/^[a-zA-Z][A-Za-z0-9]{2,5}$/)) {

return true;

}

return false;

}

/\*\*

\* 检查输入的密码合法性

\*/

checkInputPassword (password) {

if(password.match(/^[a-zA-Z][A-Za-z0-9]{2,5}$/)) {

return true;

}

return false;

}

login (username, password) {

const success = (user) => {

this.props.onLoginSuccess(user);

this.props.handleCloseClick();

}

const fail = (user, e) => {

this.setState({

isLogining: false,

})

let loginInfo = '';

if(e.code === 101) {

loginInfo = '用户名或密码错误！';

}

this.setState({

loginInfo: loginInfo,

})

}

BmobUtils.login(username, password, success, fail);

}

render () {

return (

<div id="login" className="login" style={{ display: this.props.showLogin?'block':'none' }}>

<div className="mask"></div>

<div className="content">

<a onClick={this.handleCloseClick} className="close-login">

<span className="glyphicon glyphicon-remove" aria-hidden="true"></span>

</a>

<form className="form-horizontal" onSubmit={this.handleLoginSubmit}>

<div className="form-group">

<label htmlFor="username" className="col-xs-2 col-sm-2 control-label">账号:</label>

<div className="col-xs-10 col-sm-10">

<input ref="username" type="text" className="form-control" id="username" placeholder="Username"></input>

</div>

</div>

<div className="form-group">

<label htmlFor="password" className="col-xs-2 col-sm-2 control-label">密码:</label>

<div className="col-xs-10 col-sm-10">

<input ref="password" type="password" className="form-control" id="password" placeholder="Password"></input>

</div>

</div>

<div className="form-group">

<div className="col-xs-offset-2 col-xs-10 col-sm-offset-2 col-sm-10">

<div className="checkbox">

<label>

<input type="checkbox"></input>记住密码

</label>

</div>

</div>

</div>

<div className="form-group">

<div className="col-xs-offset-2 col-xs-10 col-sm-offset-2 col-sm-10">

<button disabled={this.state.isLogining} type="submit" className="btn btn-primary">登&nbsp;&nbsp;录</button>

<button onClick={this.props.handleRegisterClick} className="btn btn-default">注&nbsp;&nbsp;册</button>

<span className="text-danger">{this.state.loginInfo}</span>

</div>

</div>

</form>

</div>

</div>

)

}

}

const mapStateToProps = (state) => {

return {

currentUser: state.currentUser

}

}

const mapDispatchToProps = (dispatch, ownProps) => {

return {

onLoginSuccess: (user) => {

dispatch({

type: Action.LOGOUT,

payload: {

currentUser: user

}

})

}

}

}

const VisibleLogin = connect(

mapStateToProps,

mapDispatchToProps

)(Login);

export default VisibleLogin;

###### Main.jsx

/\*\*

\* 路由入口组件

\* 包括了导航栏组件和主体组件

\*/

import React from 'react';

import Navigation from './Navigation.jsx';

import { connect, Provider } from 'react-redux';

import BmobUtils from '../util/bmobUtils.jsx';

import Action from '../Action.jsx';

class Main extends React.Component {

render () {

return (

<div>

<Navigation/>

{ this.props.children }

</div>

)

}

}

const mapStateToProps = (state) => {

return {

currentUser: state.currentUser,

}

}

const mapDispatchToProps = (dispatch, ownProps) => {

return {

onClick: () => {

dispatch({

type:0x1,

payload: {

currentUser: Bmob.User.current()

}

})

},

initBookCollectionData: () => {

BmobUtils.getCurrentUserBookCollection((results) => {

dispatch({

type: Action.INIT\_BOOK\_COLLECTION,

payload: {

bookCollection: results

}

})

});

},

initMusicCollectionData: () => {

BmobUtils.getCurrentUserMusicCollection((results) => {

dispatch({

type: Action.INIT\_MUSIC\_COLLECTION,

payload: {

musicCollection: results

}

})

});

},

initFilmCollectionData: () => {

BmobUtils.getCurrentUserFilmCollection((results) => {

dispatch({

type: Action.INIT\_FILM\_COLLECTION,

payload: {

filmCollection: results

}

})

});

}

}

}

const VisibleMain = connect(

mapStateToProps,

mapDispatchToProps

)(Main);

export default VisibleMain;

###### MainRouter.jsx

/\*\*

\* 路由

\*/

import React from 'react';

import { Router, Route, IndexRoute, hashHistory } from 'react-router';

import VisibleMain from './Main.jsx';

import Home from './Home.jsx';

import Book from './Book.jsx';

import Music from './Music.jsx';

import Film from './Film.jsx';

import Search from './Search.jsx';

import BookDetails from './BookDetails.jsx';

import MusicDetails from './MusicDetails.jsx';

import FilmDetails from './FilmDetails.jsx';

import UserCollection from './UserCollection.jsx';

class MainRouter extends React.Component {

/\*\*

\* 无论什么时候，顶层路由都不应该发生改变

\*/

shouldComponentUpdate(){

return false;

}

render () {

const routes = <Route path="/" component={VisibleMain}>

<IndexRoute component={Home}/>

<Route path="user\_collection" component={UserCollection}/>

<Route path="book(/:category)" component={Book}/>

<Route path="music(/:category)" component={Music}/>

<Route path="film(/:category)" component={Film}/>

<Route path="book\_details/:id" component={BookDetails}/>

<Route path="music\_details/:id" component={MusicDetails}/>

<Route path="film\_details/:id" component={FilmDetails}/>

<Route path="search/:key" component={Search}/>

</Route>;

return <Router routes={routes} history={hashHistory}/>;

}

}

export default MainRouter;

###### Music.jsx

import React from 'react';

import { Link } from 'react-router';

import SubNavigation from './SubNavigation.jsx';

import Global from '../Global.jsx';

import { fetchData } from '../util/utils.jsx';

import Footer from './Footer.jsx';

import BackToTop from './BackToTop.jsx';

import LoadMore from './LoadMore.jsx';

import music1 from '../../json/music1.json';

import music2 from '../../json/music2.json';

import music3 from '../../json/music3.json';

import music4 from '../../json/music4.json';

import music5 from '../../json/music5.json';

import music6 from '../../json/music6.json';

const data = [music1, music2, music3, music4, music5, music6];

class Music extends React.Component {

constructor(props) {

super(props);

this.loadMoreTop = 0;

this.start = 0;

this.state = {

musics: [],

};

}

componentWillMount () {

this.initMusics();

}

componentWillReceiveProps () {

this.initMusics();

}

initMusics () {

let category = this.props.params.category;

for(let i=0; i<Global.MUSIC\_CATEGORY.length; i++){

let item = Global.MUSIC\_CATEGORY[i];

if(item.category == category){

// fetchData(item.baseURL, {}, (data) => {

// console.log(data);

// this.setState({

// musics: data.musics

// })

// });

this.setState({

musics: data[i].musics

})

}

}

}

componentDidMount () {

this.initLoadMore();

}

componentDidUpdate(prevProps, prevState){

this.loadMoreTop = $('#loadMore').offset().top;

}

initLoadMore () {

let that = this;

let height = $(window).height();

let isLoadMore = true;

$(window).on('scroll', (e) => {

if(isLoadMore) {

if(this.loadMoreTop - $(window).scrollTop() <= height) {

isLoadMore = false;

setTimeout(() => {

// let category = this.props.params.category;

// for(let i=0; i<Global.MUSIC\_CATEGORY.length; i++){

// let item = Global.MUSIC\_CATEGORY[i];

// if(item.category == category){

// fetchData(Global.MUSIC\_CATEGORY[i].baseURL, {start: this.start}, (data) => {

// let arr = that.state.musics;

// arr.push(...data.musics);

// this.setState({

// musics: arr

// })

// this.start += 5;

// });

// }

// }

let category = this.props.params.category;

for(let i=0; i<Global.MUSIC\_CATEGORY.length; i++){

let item = Global.MUSIC\_CATEGORY[i];

if(item.category == category){

let arr = that.state.musics;

arr.push(data[i].musics[this.start++ % 4]);

this.setState({

musics: arr

})

}

}

isLoadMore = true;

}, 1000)

}

}

})

}

render () {

return (

<div className="container">

<div className="row">

<SubNavigation category={ Global.MUSIC } list={ Global.MUSIC\_CATEGORY }/>

<div className="col-xs-12 col-sm-9 col-sm-offset-2 col-md-8 col-md-offset-1 list-content">

<ul>

{

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

let tags = '';

for(let i=0; i<item.tags.length; i++){

tags += item.tags[i].name + ' ';

}

return (

<li key={index} className="list-item">

<div className="media">

<div className="media-left">

<Link to={`/music\_details/${item.id}`}>

<img className="media-object dd" src={item.image} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/music\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">标签:{tags}</p>

<p>发行年份:{item.attrs.pubdate}</p>

<p className="ellipsis">专辑音乐:{item.attrs.tracks?item.attrs.tracks[0]:''}</p>

</div>

</div>

</li>

)

})

}

</ul>

<LoadMore/>

</div>

<Footer/>

<BackToTop/>

</div>

</div>

)

}

}

export default Music;

###### MusicDetails.jsx

import React from 'react';

import { fetchData } from '../util/utils.jsx';

import { connect } from 'react-redux';

import BmobUtils from '../util/bmobUtils.jsx';

import Global from '../Global.jsx';

import Action from '../Action.jsx';

class MusicDetails extends React.Component {

constructor(props) {

super(props);

this.state = {

music: {

rating: {

numRaters: 0,

},

title: '',

image: '',

summary: '',

author: [],

tags: [],

attrs: {

pubdate: [ '' ],

tracks: [ '' ],

publisher: [ '' ],

},

},

};

this.handleCollectionClick = this.handleCollectionClick.bind(this);

this.handleCancelCollectionClick = this.handleCancelCollectionClick.bind(this);

}

handleCollectionClick () {

this.props.onCollectionClick(this.props.currentUser, this.state.music, this.props.currentUserMusicCollection);

}

handleCancelCollectionClick () {

this.props.onCancelCollectionClick(this.props.currentUser, this.props.currentUserMusicCollection);

}

componentWillMount () {

let that = this;

let id = this.props.params.id;

let url = Global.MUSIC\_DETAILS\_BASE\_URL + id;

fetchData(url, {}, (data) => {

if(that.refs.myRef)

that.setState({

music: data,

})

});

}

render () {

let music = this.state.music;

let author = '';

for(let i=0; i<music.author.length; i++){

author += music.author[i].name + ' ';

}

let tags = '';

for(let i=0; i<music.tags.length; i++){

tags += music.tags[i].name + ' ';

}

let hasCollection = false;

for(let i=0; i<this.props.currentUserMusicCollection.length; i++){

if(this.props.currentUserMusicCollection[i].attributes.musicId == this.props.params.id){

hasCollection = true;

}

}

return (

<div className="container details" ref="myRef">

<div className="details-jumbotron">

<div className="media">

<div className="media-left">

<a href="#">

<img className="media-object" src={music.image} alt={music.title}></img>

</a>

</div>

<div className="media-body">

<span className="badge badge-details">

<span className="glyphicon glyphicon-thumbs-up" aria-hidden="true"></span>

<span title="收藏数">&nbsp;{music.rating.numRaters}</span>

</span>

<h1>{music.title}</h1>

<p><span className="text-bold">标签：</span>{tags}</p>

<p><span className="text-bold">作者：</span>{author}</p>

<p><span className="text-bold">出版日期：</span>{music.attrs.pubdate[0]}</p>

<p><span className="text-bold">出版商：</span>{music.attrs.publisher[0]}</p>

<p><span className="text-bold">摘要：</span>{music.summary}</p>

<p><span className="text-bold">专辑音乐：</span>{music.attrs.tracks[0]}</p>

<button style={{display:hasCollection==false?'inline-block':'none'}} onClick={this.handleCollectionClick} className="btn btn-focus btn-lg">

收&nbsp;&nbsp;藏

</button>

<button style={{display:hasCollection==true?'inline-block':'none'}} onClick={this.handleCancelCollectionClick} className="btn btn-default btn-lg">

取消收藏

</button>

</div>

</div>

</div>

</div>

)

}

}

const mapStateToProps = (state) => {

return {

currentUser: state.currentUser,

currentUserMusicCollection: state.currentUserMusicCollection,

}

}

const mapDispatchToProps = (dispatch, ownProps) => {

return {

onCollectionClick: (currentUser, music, currentUserMusicCollection) => {

if(currentUser){

if(currentUserMusicCollection.indexOf(ownProps.params.id) == -1){

BmobUtils.addMusicCollection(currentUser.attributes.username, music.id, music, (musicCollection) => {

dispatch({

type: Action.ADD\_MUSIC\_COLLECTION,

payload: {

musicCollection: musicCollection

}

})

}, (error) => {

console.log(error);

});

}

}

},

onCancelCollectionClick: (currentUser, currentUserMusicCollection) => {

if(currentUser){

let musicCollection = null;

for(let i=0; i<currentUserMusicCollection.length; i++){

if(currentUserMusicCollection[i].attributes.musicId == ownProps.params.id){

musicCollection = currentUserMusicCollection[i];

}

}

BmobUtils.removeMusicCollection(musicCollection, () => {

dispatch({

type: Action.REMOVE\_MUSIC\_COLLECTION,

payload: {

musicCollectionId: musicCollection.id

}

})

}, (error) => {

console.log(error);

});

}

}

}

}

const VisibleMusicDetails = connect(

mapStateToProps,

mapDispatchToProps

)(MusicDetails);

export default VisibleMusicDetails;

###### Navigation.jsx

/\*\*

\* 导航条组件

\*/

import React from 'react';

import { IndexLink, Link } from 'react-router';

import Global from '../Global.jsx';

import { hashHistory } from 'react-router'

import { connect } from 'react-redux';

import Action from '../Action.jsx';

import BmobUtils from '../util/bmobUtils.jsx';

import Login from './Login.jsx';

import Register from './Register.jsx';

class Navigation extends React.Component {

constructor(props) {

super(props);

this.state = {

showLogin: false,

showRegister: false,

};

this.handleLoginClick = this.handleLoginClick.bind(this);

this.handleRegisterClick = this.handleRegisterClick.bind(this);

this.handleCloseClick = this.handleCloseClick.bind(this);

this.handleLogoutClick = this.handleLogoutClick.bind(this);

}

componentWillMount () {

this.props.initBookCollectionData();

this.props.initMusicCollectionData();

this.props.initFilmCollectionData();

}

shouldComponentUpdate () {

this.props.initBookCollectionData();

this.props.initMusicCollectionData();

this.props.initFilmCollectionData();

return true;

}

handleSearchSubmit (e) {

e.preventDefault();

let key = e.target.childNodes[0].value;

if(key) {

const path = `search/${key}`;

hashHistory.push(path);

e.target.childNodes[0].value = '';

}

}

handleLoginClick (e) {

e ? e.preventDefault() : null;

this.setState({

showLogin: true,

showRegister: false,

})

}

handleRegisterClick (e) {

e.preventDefault();

this.setState({

showLogin: false,

showRegister: true,

})

}

handleCloseClick (e) {

this.setState({

showLogin: false,

showRegister: false,

})

}

handleLogoutClick (e) {

e.preventDefault();

this.props.onLogoutClick();

BmobUtils.logout();

}

render () {

let currentUser = this.props.currentUser;

return (

<nav className="navbar navbar-default navbar-fixed-top">

<div className="container">

<div className="row">

<div className="navbar-header">

<button type="button" className="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar" aria-expanded="false" aria-controls="navbar">

<span className="sr-only">Toggle navigation</span>

<span className="icon-bar"></span>

<span className="icon-bar"></span>

<span className="icon-bar"></span>

</button>

<a className="navbar-brand" href="#">生活百科</a>

</div>

<div id="navbar" className="navbar-collapse collapse">

<ul className="nav navbar-nav">

<li>

<IndexLink to="/" activeClassName="active">

<span className="glyphicon glyphicon-home" aria-hidden="true"></span>

&nbsp;主页

</IndexLink>

</li>

<li>

<Link to={`/${Global.BOOK}/${Global.BOOK\_CATEGORY[0].category}`}>

<span className="glyphicon glyphicon-book" aria-hidden="true"></span>

&nbsp;图书

</Link>

</li>

<li>

<Link to={`/${Global.MUSIC}/${Global.MUSIC\_CATEGORY[0].category}`}>

<span className="glyphicon glyphicon-music" aria-hidden="true"></span>

&nbsp;音乐

</Link>

</li>

<li>

<Link to={`/${Global.FILM}/${Global.FILM\_CATEGORY[0].category}`}>

<span className="glyphicon glyphicon-film" aria-hidden="true"></span>

&nbsp;电影

</Link>

</li>

</ul>

<ul style={{display:currentUser==null?'block':'none'}} className="nav navbar-nav navbar-right">

<li><a onClick={this.handleLoginClick}>登录</a></li>

<li><a onClick={this.handleRegisterClick}>注册</a></li>

</ul>

<ul style={{display:currentUser==null?'none':'block'}} className="nav navbar-nav navbar-right">

<li><Link to={`/user\_collection`}>{currentUser!=null?currentUser.attributes.username:''}</Link></li>

<li><a onClick={this.handleLogoutClick}>注销</a></li>

</ul>

<form className="navbar-form navbar-right" onSubmit={this.handleSearchSubmit}>

<input type="text" className="form-control" placeholder="搜索..."></input>

</form>

</div>

</div>

</div>

<Login showLogin={this.state.showLogin} handleCloseClick={this.handleCloseClick} handleRegisterClick={this.handleRegisterClick}/>

<Register handleLoginClick={this.handleLoginClick} showRegister={this.state.showRegister} handleCloseClick={this.handleCloseClick}/>

</nav>

)

}

}

const mapStateToProps = (state) => {

return {

currentUser: state.currentUser,

}

}

const mapDispatchToProps = (dispatch, ownProps) => {

return {

onLogoutClick: () => {

dispatch({

type:0x2,

payload: {

currentUser: null,

currentUserBookCollection: [],

currentUserMusicCollection: [],

currentUserFilmCollection: [],

}

})

},

initBookCollectionData: () => {

BmobUtils.getCurrentUserBookCollection((results) => {

dispatch({

type: Action.INIT\_BOOK\_COLLECTION,

payload: {

bookCollection: results

}

})

});

},

initMusicCollectionData: () => {

BmobUtils.getCurrentUserMusicCollection((results) => {

dispatch({

type: Action.INIT\_MUSIC\_COLLECTION,

payload: {

musicCollection: results

}

})

});

},

initFilmCollectionData: () => {

BmobUtils.getCurrentUserFilmCollection((results) => {

dispatch({

type: Action.INIT\_FILM\_COLLECTION,

payload: {

filmCollection: results

}

})

});

}

}

}

const VisibleNavigation = connect(

mapStateToProps,

mapDispatchToProps

)(Navigation);

export default VisibleNavigation;

###### Register.jsx

import React from 'react';

import BmobUtils from '../util/bmobUtils.jsx';

class Register extends React.Component {

constructor(props) {

super(props);

this.state = {

registerInfo: '',

isRegisting: false,

isRegistedSuccessful: false,

};

this.handleCloseClick = this.handleCloseClick.bind(this);

this.handleRegisterSubmit = this.handleRegisterSubmit.bind(this);

this.handleToLogin = this.handleToLogin.bind(this);

}

shouldComponentUpdate () {

this.clearInput();

return true

}

clearInput () {

this.refs.username.value = '';

this.refs.password.value = '';

this.refs.passwordConfirm.value = '';

this.refs.email.value = '';

this.setState({

registerInfo: '',

})

}

handleCloseClick (e) {

e.preventDefault();

this.props.handleCloseClick();

}

handleRegisterSubmit (e) {

e.preventDefault();

let username = this.refs.username.value;

let password = this.refs.password.value;

let passwordConfirm = this.refs.passwordConfirm.value;

let email = this.refs.email.value;

let registerInfo = '';

if(this.checkInputUsername(username) && this.checkInputPassword(password) && this.checkInputPassword(passwordConfirm) && password == passwordConfirm && email) {

registerInfo = '正在注册...';

this.setState({

isRegisting: true,

})

this.register(username, password, email);

}else if(!this.checkInputUsername(username)) {

registerInfo = '用户名不合法！';

}else if(!this.checkInputPassword(password)) {

registerInfo = '密码不合法！';

}else if(!this.checkInputPassword(passwordConfirm)) {

registerInfo = '确认密码不合法！';

}else if(password !== passwordConfirm) {

registerInfo = '两次输入的密码不匹配！';

}else if(!email) {

registerInfo = '邮箱不合法';

}

this.setState({

registerInfo: registerInfo,

})

}

/\*\*

\* 注册用户方法

\*/

register (username, password, email) {

const success = (user) => {

this.setState({

isRegistedSuccessful: true,

})

}

const fail = (user, e) => {

console.log('注册失败', e);

}

BmobUtils.register(username, password, email, success, fail);

}

/\*\*

\* 检查输入的用户名合法性

\*/

checkInputUsername (username) {

if(username.match(/^[a-zA-Z][A-Za-z0-9]{2,8}$/)) {

return true;

}

return false;

}

/\*\*

\* 检查输入的密码合法性

\*/

checkInputPassword (password) {

if(password.match(/^[a-zA-Z][A-Za-z0-9]{2,8}$/)) {

return true;

}

return false;

}

handleToLogin () {

this.setState({

isRegistedSuccessful: false,

})

this.props.handleLoginClick();

}

render () {

return (

<div className="register" style={{ display: this.props.showRegister?'block':'none' }}>

<div className="mask"></div>

<div className="content">

<a onClick={this.handleCloseClick} className="close-register">

<span className="glyphicon glyphicon-remove" aria-hidden="true"></span>

</a>

<form style={{ display: this.state.isRegistedSuccessful ? "none" : 'block'}} className="form-horizontal" onSubmit={this.handleRegisterSubmit}>

<div className="form-group">

<label htmlFor="username" className="col-xs-3 col-sm-3 control-label">账号:</label>

<div className="col-xs-9 col-sm-9">

<input ref="username" type="text" className="form-control" id="username" placeholder="Username"></input>

</div>

</div>

<div className="form-group">

<label htmlFor="password" className="col-xs-3 col-sm-3 control-label">密码:</label>

<div className="col-xs-9 col-sm-9">

<input ref="password" type="password" className="form-control" id="password" placeholder="Password"></input>

</div>

</div>

<div className="form-group">

<label htmlFor="passwordConfirm" className="col-xs-3 col-sm-3 control-label">确认密码:</label>

<div className="col-xs-9 col-sm-9">

<input ref="passwordConfirm" type="password" className="form-control" id="passwordConfirm" placeholder="Password Confirm"></input>

</div>

</div>

<div className="form-group">

<label htmlFor="eamil" className="col-xs-3 col-sm-3 control-label">邮箱:</label>

<div className="col-xs-9 col-sm-9">

<input ref="email" type="email" className="form-control" id="eamil" placeholder="Email"></input>

</div>

</div>

<div className="form-group">

<div className="col-sm-offset-3 col-sm-10">

<button type="submit" className="btn btn-default" disabled={this.state.isRegisting}>注&nbsp;&nbsp;册</button>

<span className="text-danger">{this.state.registerInfo}</span>

</div>

</div>

</form>

<div style={{ display: this.state.isRegistedSuccessful ? "block" : 'none'}} className="registed-successful">

<p className="logo text-center text-success"><span className="glyphicon glyphicon-ok-circle"></span></p>

<p className="info text-center text-success">恭喜！注册成功。</p>

<button onClick={this.handleToLogin} className="to-login btn btn-default btn-block">去登录</button>

</div>

</div>

</div>

)

}

}

export default Register;

###### Search.jsx

import React from 'react';

import { Link } from 'react-router';

import $ from 'jquery';

import Global from '../Global.jsx';

import { obj2uri, fetchData } from '../util/utils.jsx';

class Search extends React.Component {

constructor(props) {

super(props);

this.state = {

books: [],

musics: [],

films: [],

books\_count: 0,

musics\_count: 0,

films\_count: 0,

};

}

componentWillMount () {

this.initData();

}

componentWillReceiveProps (nextProps) {

this.initData();

}

/\*\*

\* 初始化数据方法

\*/

initData () {

let key = this.props.params.key;

let params = obj2uri({ tag: key, count: 10 });

fetchData(`${Global.BOOK\_SEARCH\_BASE\_URL}?${params}&`, {}, (data) => {

this.setState({

books: data.books,

books\_count: data.count,

})

});

fetchData(`${Global.MUSIC\_SEARCH\_BASE\_URL}?${params}&`, {}, (data) => {

this.setState({

musics: data.musics,

musics\_count: data.count,

})

});

fetchData(`${Global.FILM\_SEARCH\_BASE\_URL}?${params}&`, {}, (data) => {

this.setState({

films: data.subjects,

films\_count: data.count,

})

});

}

render () {

return (

<div className="container search">

<div className="row">

<div className="search-header col-md-12">

<div className="search-info">

<p>

<span>关键字：{this.props.params.key}</span>

<span>总条目：{this.state.books\_count + this.state.musics\_count + this.state.films\_count}</span>

<br/>

<span>图书：{this.state.books\_count}</span>

<span>音乐：{this.state.musics\_count}</span>

<span>电影：{this.state.films\_count}</span>

</p>

</div>

<ul id="nav\_search" className="nav nav-tabs nav-justified">

<li className="active"><a href="#tab1" data-toggle="tab">Book</a>

</li>

<li><a href="#tab2" data-toggle="tab">Music</a>

</li>

<li><a href="#tab3" data-toggle="tab">Film</a>

</li>

</ul>

</div>

<div className="search-body col-md-12 container">

<div className="tab-content">

<div className="tab-pane active row" id="tab1">

<p className="text-center" style={{display: this.state.books.length!=0?'none':'block'}}>

没找到"{this.props.params.key}"的相关图书信息!

</p>

<ul>

{

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

let tags = '';

for(let i=0; i<item.tags.length; i++){

tags += item.tags[i].name + ' ';

}

return (

<li key={index} className="search-item col-md-6">

<div className="media">

<div className="media-left">

<Link to={`/book\_details/${item.id}`}>

<img className="media-object dd" src={item.image} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/book\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">标签:{tags}</p>

<p>作者:{item.author}</p>

<p>出版社:{item.publisher}</p>

</div>

</div>

</li>

)

})

}

</ul>

</div>

<div className="tab-pane row" id="tab2">

<p className="text-center" style={{display: this.state.musics.length!=0?'none':'block'}}>

没找到"{this.props.params.key}"的相关音乐信息!

</p>

<ul>

{

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

let tags = '';

for(let i=0; i<item.tags.length; i++){

tags += item.tags[i].name + ' ';

}

return (

<li key={index} className="search-item col-md-6">

<div className="media">

<div className="media-left">

<Link to={`/music\_details/${item.id}`}>

<img className="media-object dd" src={item.image} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/music\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">标签:{tags}</p>

<p>发行年份:{item.attrs.pubdate}</p>

<p className="ellipsis">专辑音乐:{item.attrs.tracks!==undefined ? item.attrs.tracks[0] : '无'}</p>

</div>

</div>

</li>

)

})

}

</ul>

</div>

<div className="tab-pane row" id="tab3">

<p className="text-center" style={{display: this.state.films.length!=0?'none':'block'}}>

没找到"{this.props.params.key}"的相关电影信息!

</p>

<ul>

{

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

let casts = '';

for(let i=0; i<item.casts.length; i++){

casts += item.casts[i].name + ' ';

}

let directors = '';

for(let i=0; i<item.directors.length; i++){

directors += item.directors[i].name + ' ';

}

return (

<li key={index} className="search-item col-md-6">

<div className="media">

<div className="media-left">

<Link to={`/film\_details/${item.id}`}>

<img className="media-object dd" src={item.images.small} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/film\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">年份:{item.year}</p>

<p>演员:{casts}</p>

<p>导演:{directors}</p>

</div>

</div>

</li>

)

})

}

</ul>

</div>

</div>

</div>

</div>

</div>

)

}

}

export default Search;

###### SubNavigation.jsx

import React from 'react';

import { IndexLink, Link } from 'react-router';

class SubNavigation extends React.Component {

render () {

let category = this.props.category;

return (

<div className="col-sm-2 col-md-1 sidebar">

<ul className="nav nav-sidebar">

{

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

return (

<li key={index}><Link to={`/${category}/${item.category}`} activeClassName="active">{item.name}</Link></li>

)

})

}

</ul>

</div>

)

}

}

export default SubNavigation;

###### UserCollection.jsx

import React from 'react';

import { Link } from 'react-router';

import { connect } from 'react-redux';

import BmobUtils from '../util/bmobUtils.jsx';

class UserCollection extends React.Component {

constructor(props) {

super(props);

this.state = {

books: [],

musics: [],

films: [],

books\_count: 0,

musics\_count: 0,

films\_count: 0,

};

}

// componentWillMount () {

// this.initData();

// }

componentWillReceiveProps (nextProps) {

this.initData();

}

/\*\*

\* 初始化数据

\*/

initData () {

BmobUtils.getCurrentUserBookCollection((results) => {

let books = [];

for(let i=0; i<results.length; i++){

let json = JSON.parse(results[i].attributes.book);

books.push(json)

}

this.setState({

books: books,

books\_count: books.length,

})

})

BmobUtils.getCurrentUserMusicCollection((results) => {

let musics = [];

for(let i=0; i<results.length; i++){

let json = JSON.parse(results[i].attributes.music);

musics.push(json)

}

this.setState({

musics: musics,

musics\_count: musics.length,

})

})

BmobUtils.getCurrentUserFilmCollection((results) => {

let films = [];

for(let i=0; i<results.length; i++){

let json = JSON.parse(results[i].attributes.film);

films.push(json)

}

this.setState({

films: films,

films\_count: films.length,

})

})

}

render () {

return (

<div className="container search">

<div className="row">

<div className="search-header col-md-12">

<div className="search-info">

<p style={{display:this.props.currentUser!=null?'block':'none'}}>

尊敬的：

<span className="text-important">

{this.props.currentUser!=null?this.props.currentUser.attributes.username:''}

</span>

<span>您一共收藏了：{this.state.books\_count + this.state.musics\_count + this.state.films\_count}个条目</span>

<br/>

<span>图书：{this.state.books\_count}</span>

<span>音乐：{this.state.musics\_count}</span>

<span>电影：{this.state.films\_count}</span>

</p>

<p style={{display:this.props.currentUser==null?'block':'none'}} className="text-important">

你还没登录呢！

</p>

</div>

<ul id="nav\_search" className="nav nav-tabs nav-justified">

<li className="active"><a href="#tab1" data-toggle="tab">Book</a>

</li>

<li><a href="#tab2" data-toggle="tab">Music</a>

</li>

<li><a href="#tab3" data-toggle="tab">Film</a>

</li>

</ul>

</div>

<div className="search-body col-md-12 container">

<div className="tab-content">

<div className="tab-pane active row" id="tab1">

<p className="text-center" style={{display: this.state.books.length!=0?'none':'block'}}>

你还没有收藏一本图书呢！

</p>

<ul>

{

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

let tags = '';

for(let i=0; i<item.tags.length; i++){

tags += item.tags[i].name + ' ';

}

return (

<li key={index} className="search-item col-md-6">

<div className="media">

<div className="media-left">

<Link to={`/book\_details/${item.id}`}>

<img className="media-object dd" src={item.image} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/book\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">标签:{tags}</p>

<p>作者:{item.author}</p>

<p>出版社:{item.publisher}</p>

</div>

</div>

</li>

)

})

}

</ul>

</div>

<div className="tab-pane row" id="tab2">

<p className="text-center" style={{display: this.state.musics.length!=0?'none':'block'}}>

你还没有收藏一首音乐呢！

</p>

<ul>

{

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

let tags = '';

for(let i=0; i<item.tags.length; i++){

tags += item.tags[i].name + ' ';

}

return (

<li key={index} className="search-item col-md-6">

<div className="media">

<div className="media-left">

<Link to={`/music\_details/${item.id}`}>

<img className="media-object dd" src={item.image} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/music\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">标签:{tags}</p>

<p>发行年份:{item.attrs.pubdate}</p>

<p className="ellipsis">专辑音乐:{item.attrs.tracks!==undefined ? item.attrs.tracks[0] : '无'}</p>

</div>

</div>

</li>

)

})

}

</ul>

</div>

<div className="tab-pane row" id="tab3">

<p className="text-center" style={{display: this.state.films.length!=0?'none':'block'}}>

你还没有收藏一部电影呢！

</p>

<ul>

{

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

let casts = '';

for(let i=0; i<item.casts.length; i++){

casts += item.casts[i].name + ' ';

}

let directors = '';

for(let i=0; i<item.directors.length; i++){

directors += item.directors[i].name + ' ';

}

return (

<li key={index} className="search-item col-md-6">

<div className="media">

<div className="media-left">

<Link to={`/film\_details/${item.id}`}>

<img className="media-object dd" src={item.images.small} alt={item.title}>

</img>

</Link>

</div>

<div className="media-body item-body">

<Link to={`/film\_details/${item.id}`} className="title">{item.title}</Link>

<p className="ellipsis">年份:{item.year}</p>

<p>演员:{casts}</p>

<p>导演:{directors}</p>

</div>

</div>

</li>

)

})

}

</ul>

</div>

</div>

</div>

</div>

</div>

)

}

}

const mapStateToProps = (state) => {

return {

currentUser: state.currentUser,

currentUserBookCollection: state.currentUserBookCollection,

currentUserMusicCollection: state.currentUserMusicCollection,

currentUserFilmCollection: state.currentUserFilmCollection,

}

}

const mapDispatchToProps = (dispatch, ownProps) => {

return {

}

}

const VisibleUserCollection = connect(

mapStateToProps,

mapDispatchToProps

)(UserCollection);

export default VisibleUserCollection;