import axios from 'axios'

import { Message } from 'element-ui';

// 创建基本的请求实例

const service = axios.create({

// `baseURL` 将自动加在 `url` 前面，除非 `url` 是一个绝对 URL。

// 它可以通过设置一个 `baseURL` 便于为 axios 实例的方法传递相对 URL

baseURL:process.env.BASE\_API, // api 的 base\_url

// `timeout` 指定请求超时的毫秒数(0 表示无超时时间)

// 如果请求话费了超过 `timeout` 的时间，请求将被中断

timeout:20000

})

/\*\* 给请求实例添加拦截器\*/

//添加请求拦截器

service.interceptors.request.use(function(config){

// 在发送请求之前做些什么

if(config.noUrlPrefix){

return config

}else{

config.url = '' + config.url

return config

}

},function(error){

// 对请求错误做些什么

return Promise.reject(error);

})

//添加响应拦截器

service.interceptors.response.use(function(res){

// 对响应数据做点什么

// return res;

if(res.status !== 200){ // 响应状态不是200是做的判断

Message({

message: res.statustext,

type: 'error',

duration: 5 \* 1000

})

return Promise.reject(error)

}else{ // 响应状态是200是做的判断

if (!(res.data.code == 0 || res.data.code == 200 || res.data.error == "0")) { // 后台返回的code状态不是0、200或者error的字符串是0做出判断

if (res.data.code == 403) {

return Promise.reject(error)

}

if (res.data.code == 401) {

if (location.pathname != '/login') {

// store.dispatch('LogOut');

}

}

Message({

message: res.data.msg,

type: 'error',

duration: 5 \* 1000

});

return Promise.reject(res.data.message);

} else { // 后台返回的code状态是0、200或者error的字符串是0做出判断

return res.data

}

}

},function(error){

// 对响应错误做点什么

console.log(error)

let res = error.response

console.log(res);

if (res.status === 401) {

if (location.pathname != '/login') {

// store.dispatch('LogOut');

}

} else {

let msg = error;

if (res.statusText) {

msg = res.statusText

}

if (error.response && error.response.data && error.response.data.message) {

msg = error.response.data.message;

}

Message({

message: msg,

type: 'error',

duration: 5 \* 1000

})

return Promise.reject(error)

}

})

export default service