**注册url**

import PageProvider from 'Core/Page/PageProvider'

import Page from 'Core/Page/Page'

// 导出模块

export default class Module extends BaseModule

{

    initialize(){

        // 注册一个 url

        PageProvider.register(new Page("MyModule", "/MyModule", (props) => <p>Hello World</p>));

    }

}

**注册后台菜单**



代码：

import React from 'react'

import BaseModule from 'Core/Modules/BaseModule'

import ModuleFactory from 'Core/Modules/ModuleFactory'

import CoreModule from 'Core/Module';

import { Switch, Route } from "react-router-dom";

import MenuProvider from 'Core/Menu/MenuProvider'

import AccessScope, { ApiScopeNodeType } from "Core/ApiScopeAuthority/AccessScope";

// 导出模块

export default class Module extends BaseModule {

    initialize() {

        // 注册后台菜单

        MenuProvider.registerMenu(

            {

                // 菜单id

                id: "MyModule",

                // 菜单名称

                text: "我的后台菜单",

                // 菜单图标

                icon: "oi-document",

                // 菜单url（注：应以/ManageHome开头）

                url: "/ManageHome/MyModule",

                // 菜单下的子菜单

                menuItems: [

                    {

                        id: "MyModuleChild1",

                        text: "子菜单",

                        icon: "oi-menu",

                        // 菜单url（应以父菜单的url开头，即（/ManageHome/MyModule））

                        url: "/ManageHome/MyModule/MyModuleChild1",

                        // 菜单需要访问的后台Api域

                        // 如果不清楚，不要乱写

                        accessScope:

                            [

                                // 指定该菜单需要访问Menu域的manage节点的api

                                AccessScope.Menu(ApiScopeNodeType.manage)

                            ]

                    }

                ]

            },

            // 与菜单的url一致

            "/ManageHome/MyModule",

            // 当访问/ManageHome/MyModule，将显示该组件

            (props) => (<div className="w-100 h-100 overflow-auto-y">

                <Switch>

                    <Route path="/ManageHome/MyModule/MyModuleChild1" component={(props)=>(<div>显示的内容</div>)} />

                </Switch>

            </div>),

            // 菜单的排序索引

            5

        );

    }

}

效果图：



**注册设置字段**

如果模块需要一些自定义的设置，可以注册设置字段

import React from 'react'

import BaseModule from 'Core/Modules/BaseModule'

import ModuleFactory from 'Core/Modules/ModuleFactory'

import CoreModule from 'Core/Module';

import SiteSettingManager from 'Core/SiteSettings/SiteSettingManager';

import SiteSettingGroupConfig from 'Core/SiteSettings/SiteSettingGroupConfig';

import SiteSettingConfig from 'Core/SiteSettings/SiteSettingConfig';

// 导出模块

export default class Module extends BaseModule {

    initialize() {

        // 注册设置字段

        SiteSettingManager.registerSettingConfig(new SiteSettingGroupConfig("MyModuelSetting", "我的设置字段", [

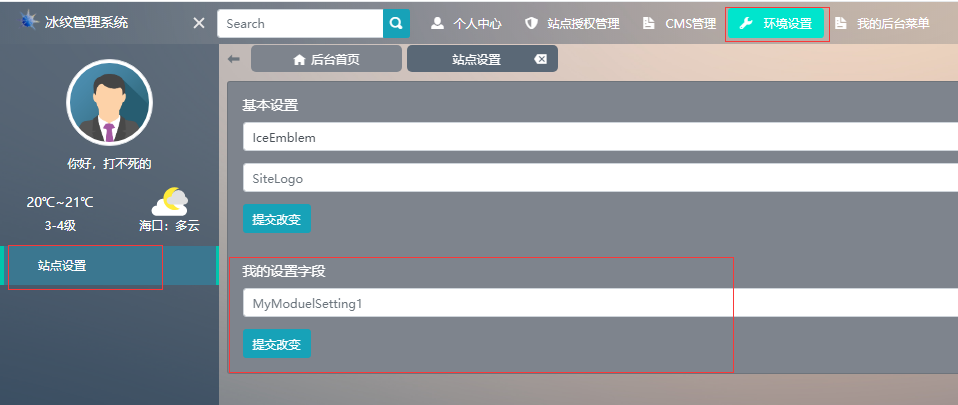
            new SiteSettingConfig("MyModuelSetting1", "我的设置字段1"),

        ]));

    }

}

效果图：



使用设置，设置放在redux的state中，如下代码取出设置：

import RootRedux from 'Core/IEReduxs/RootRedux';

const mapStateToProps = (state, ownProps) => { // ownProps为当前组件的props

    return {

        // 查找设置

        siteSetting: state.siteSettings.find(item => item.group == "MyModuelSetting" && item.key == "MyModuelSetting1")

    }

}

const mapDispatchToProps = (dispatch, ownProps) => {

    return {

    }

}

// 生成redux组件

const Container = RootRedux.connect(

    mapStateToProps, // 关于state

    mapDispatchToProps

)((props) => (<div>

    {/\* 使用设置 \*/}

    {props.siteSetting ? props.siteSetting.value : "用户没有设置"}</div>)

);