# 1.业务合约编写说明

有三点需要注意的地方：

1.1 必须包含文件OwnerNamed.sol：

import "sysbase/OwnerNamed.sol";

1.2 必须继承自OwnerNamed

contract Demo is OwnerNamed {

1.3 构造函数中必须调用注册函数register

function Demo() {

register("DemoModule", "0.0.1.0", "Demo", "0.0.1.0");

}

# 2.模块合约编写说明

模块合约编写步骤：

2.1 修改模块模板文件名DemoModule.sol -> xxxModule.sol

2.2 打开xxxModule.sol文件，全局替换DemoModule -> xxxModule，勾选匹配大小写，不能勾选全词匹配！！！

2.3 修改模块配置

tmpModule.moduleText = "权限配置测试模块";//中文模块名【需要修改】

2.4 修改合约配置

tmpContract.cctName = "Demo";//设置合约名称【需要修改】

tmpContract.enable = 1;//合约开关：0-放权；1-控权【需要修改】

增加合约的办法：拷贝粘帖以下完整的部分，并修改上面的两个字段：

//1 添加合约：Demo

tmpContract.cctName = "Demo";//设置合约名称【需要修改】

tmpContract.enable = 1;//合约开关：0-放权；1-控权【需要修改】

log(moduleName\_moduleVersion, "addContract json:", tmpContract.toJson());

ret = addContract(tmpContract.toJson());//添加合约

if (0 == ret)

log(moduleName\_moduleVersion, "addContract contractName: success.", tmpContract.cctName);

else {

log(moduleName\_moduleVersion, "addContract contractName: failed.", tmpContract.cctName);

return ret;

}

2.5 修改菜单配置

示例中包含1级和2级菜单的设置方法，拷贝一段完整的示例，修改即可。

即修改以下字段：

tmpAction.id = "DemoModule\_0.0.1.0\_Menu0001";//菜单id，修改说明：1.同步修改模块名；2.同步修改模块版本号；3.末尾4位数字从0001开始+1递增【需要修改】

tmpAction.name = "测试用顶级菜单";//菜单名称【需要修改】

tmpAction.level = 1;//菜单级别：1-顶级菜单；2-子菜单【需要修改】

tmpAction.parentId = "0";//顶级菜单无父填0，2级菜单填写父菜单id【需要修改】

tmpAction.url = "";//顶级菜单为空，2级菜单填写控制台操作方法，如user/list.do【需要修改】

2.6 修改接口配置

拷贝示例段，修改以下字段：

tmpAction.id = "DemoModule\_0.0.1.0\_Action0001";//接口id，修改说明：1.同步修改模块名；2.同步修改模块版本号；3.末尾4位数字从0001开始+1递增【需要修改】

tmpAction.name = "getData";//接口名称【需要修改】

tmpAction.enable = 0;//接口开关：0-放权；1-控权【需要修改】

tmpAction.parentId = "DemoModule\_0.0.1.0\_Menu0002";//接口所属2级菜单id【需要修改】

tmpAction.resKey = "Demo";//所属合约名称【需要修改】

tmpAction.opKey = "getData()";//接口声明【需要修改】

添加完所有合约的所有对外接口。

2.7 修改角色配置

tmpRole.id = "DemoModule\_0.0.1.0\_Role0001";//角色id，修改说明：1.同步修改模块名；2.同步修改模块版本号；3.末尾4位数字从0001开始+1递增【需要修改】

tmpRole.name = "测试角色";//角色名称【需要修改】

//设置当前角色下的接口id【需要修改】

//tmpRole.actionIdList.push("DemoModule\_0.0.1.0\_Action0001");

tmpRole.actionIdList.push("DemoModule\_0.0.1.0\_Action0002");

添加完所有角色。

# 3.业务合约和模块发布示例

3.1 首次发布：

deployer.deploy(Demo);

deployer.deploy(DemoModule);

即先发布业务合约，最后发布模块合约

3.2 非首次发布

如果修改了业务合约的接口的具体实现，并没有修改/增加/删除接口声明，那么可以单独直接发布这样的业务合约，而没必要发布对应的模块。

如果修改了接口声明，必须先修改好对应的模块代码，再一起发布。

不光业务合约可以单独发布，模块合约也可以单独发布，如修改模块或合约开关等，就可以直接只发模块。

3.3 目前模块和业务重发只支持首次发布用户的重发，如果换了钱包，需要重新发布合约或模块，则必须先调用以下接口修改模块和及其所属合约的拥有者。

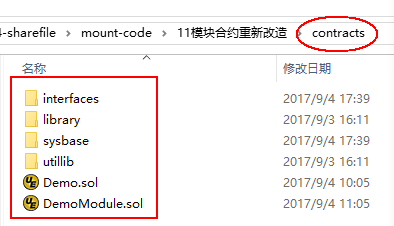
var m = RoleFilterManager.at('0x1000000000000000000000000000000000000024');

m.changeModuleOwner("DemoModule","0.0.1.0","0x00eed3de6867547d0e8073d96e024dcd1a6b11ea",{from:"0x0005838753242ae1c28e7edc6f65ccedf2b1de2a"});

参数依次为：目标模块名，目标模块版本号，新的拥有者地址，最后的from表示是管理员地址，即创建链时的第一个用户地址。

成功修改完模块的拥有者之后，就可以使用新的地址重新发布模块了。

# 4.编写业务合约的最小目录



contracts目录位于truffle工程中，其他的内置合约都不需要。

# 5.编写示例

示例业务合约：



示例模块合约：

