生成模块接口过程：

1. 数据库配置

A、打开hy\_frame\_rule 表 ，添加前端路由及后端路由规则：

id为当前路由的id，

pid为父路由的id，如果为最高级pid = 0

type 有 admin,nav,menu,url,detail 五中类型 ，其中 nav menu detail 为前台路由渲染的路由类型，url admin 为后台api 接口的路由类型。

Icon 为 前台路由的图标

Model 为前台路由类型配置，表示前台路由隶属于哪个model

Route 为前台路由的路径名

Path 为 前台路由的路径形式 ，以id-id-id… 这种形式连接，越低级的路由id在越前面

Source 为后端路由接口对应原始路径，

distance 为 重定向后实际的路径名

Method代表该路径以什么样的方式请求 其中 1是 get,2是post,3是put,4是delete,0是前端路由

B、路由与权限的关系配置:

hy\_frame\_access表配置权限

role\_id 为角色的id，rule\_id 为 hy\_frame\_rule 表 的字段id

C、指定管理员所属角色

打开hy\_frame\_manager 表

Id 为管理员账号

Name 为管理员名

Password为管理员密码

Role\_id 为管理员所属角色

比如我前台有一个User页要进行curd操作，他的api有

GET http://域名/users/lists?currPage=1&pageSize=10

GET http://域名/users/123/read

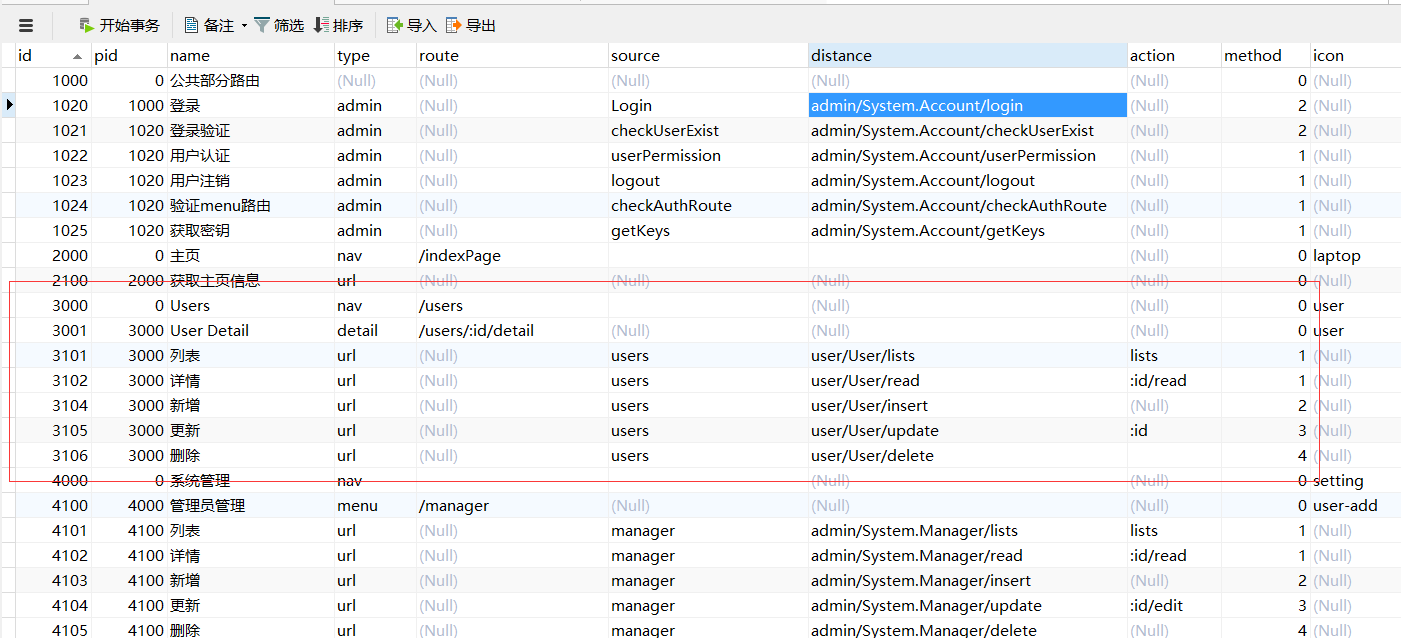
POST http://域名/users

PUT http://域名/users/123

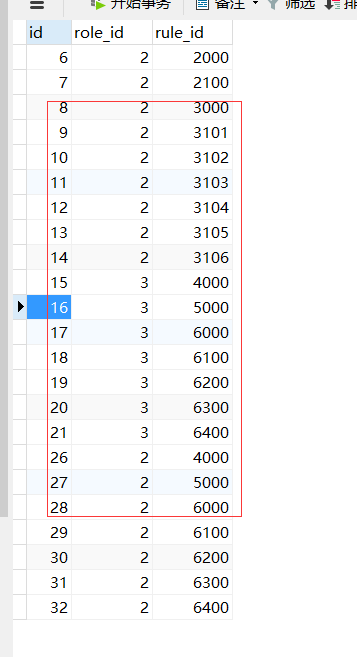
DELETE http://域名/users/123

类似这样的curd操作访问后台获取数据

数据库可进行如下配置



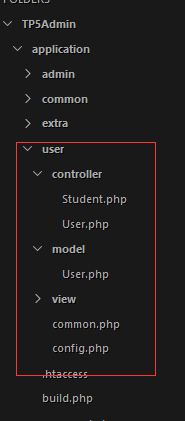
然后再在frame\_access 填好相应的角色权限信息即可配置好



1. 代码层面配置

新建一个User模块

目录结构类似

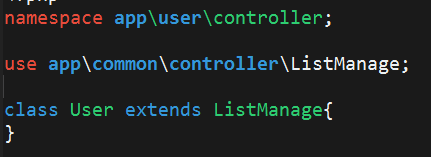


在Controller里面创建一个同名文件User.php

配置如下

Namespace 位置对应

引入并继承 ListManage 这个类即可



接着配置model

Model 引入 HyList 并继承

主要有下面几个方法

首先指定 public $table = DTP.'user'; 该模型对应表

initPk() 初始化模型对应主键

initSqlType() 返回一个字符串指定sql的查询规则，默认使用内置的getDtData

initSqlOptions() 返回一个数组，字段如下

[

'alias' => 'u', 当前表别名

//配置了就开启连表操作

'associate' => [

[

'alias' => 'a',

'table' => 'article',//关联表名

'foreignKey' => 'user\_id',//当前表连接的关联表的外键 a.user\_id

'key' => 'id',//当前表的主键 u.id

'limit'=>'a.status = 1',

'searchType'=>'eq',

'type' => 'inner'

],

[

'alias' => 'fr',

'table' => 'frame\_role',

'foreignKey' => 'id',//fr.id

'key' => 'role\_id',//u.role\_id

'searchType'=>'in'

// 'type' => false

]

],

/\*where 查询，传参形式支持ThinkPHP5的传参形式\*/

'where' => [

'u.status' => array('eq',1),

// 'a.user\_id'=>array('lt',3)

],

//field选择，表示选择哪些字段

'field' => [

// 'a.title AS article\_title',

// 'fr.name AS frame\_role\_name',

// 'u.username','u.password','u.gender AS user\_gender,u.create\_time,u.email'

'id',

'username',

'age',

'phone',

'address',

'email',

'gender',

'create\_time'

]

];

initFieldsOptions() 配置前后台字段的显示规则

[

'table' => 'user', 该字段所属哪张表，如果为当前模块的字段可忽略

'name'=>'username', 字段名 以及 唯一标示字段的id

'label' => '名字', 字段标题

'render'=>[ 是否开启前台渲染功能，前台会在渲染之前配置一个render函数 在字段渲染之前做处理

'show'=>true

],

//表单配置项

'form'=>[

'type'=>['add','edit'], add表示新增表单有此字段，edit为编辑表单，both为两者都有

//前台的表单验证规则 参照rc-form.js

'rules'=>[

'type'=>'string',

'required'=>true,

],

'callback'=>['setVal']

],

/\*搜索框验证规则\*/

'search'=>[

'open'=>true, 是否打开搜索功能

/\*前台的验证规则\*/

'rules'=>[

'type'=>'string',

'required'=>false,

'message'=>'请输入姓名！',

],

/\*后台的验证规则\*/

'backend'=>[

'type'=>'like',//模糊查询

// 'callback'=>['']

]

]

],

此时后台部分的API接口配置完成！