

# ThinkPHP 中的 RBAC






TP 在包中提供了一个叫 RBAC 的类，这个类中就是将来检查权限的代码。

TP 中提供了一种 RBAC 的实现思路和部分代码，我们项目还需要自己完成另一部分。

TP 中提供的部分是：

## 1. 建表的思路





权限表：

```
CREATE TABLE IF NOT EXISTS `think_node` (  
  `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,  
  `name` varchar(20) NOT NULL,  权限的名称  
  `title` varchar(50) DEFAULT NULL,  
  `status` tinyint(1) DEFAULT '0',  是否启用，1：启用 0：不启用  
  `remark` varchar(255) DEFAULT NULL,  描述  
  `sort` smallint(6) unsigned DEFAULT NULL,  
  `pid` smallint(6) unsigned NOT NULL,  上级权限的ID，0：顶级  
  `level` tinyint(1) unsigned NOT NULL,  权限是第几级，只能是：1,2,3  
  PRIMARY KEY (`id`),  
  KEY `level` (`level`),  
  KEY `pid` (`pid`),  
  KEY `status` (`status`),  
  KEY `name` (`name`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

例子：有一个权限：添加商品：Home/Goods/add，需要向这个表中插入三条记录：

id	name	status	pid	level
1	Home	1	0	1
2	Goods	1	1	2
3	add	1	2	3

角色表

```
CREATE TABLE IF NOT EXISTS `think_role` (  
  `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,  
  `name` varchar(20) NOT NULL,  角色名称  
  `pid` smallint(6) DEFAULT NULL,  上级角色  
  `status` tinyint(1) unsigned DEFAULT NULL,  是否启用，1：启用 0：不启用  
  `remark` varchar(255) DEFAULT NULL,  描述  
  PRIMARY KEY (`id`),  
  KEY `pid` (`pid`),  
  KEY `status` (`status`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8 ;
```

角色权限表（一个角色所拥有的权限存在这个表中）：比如说 1 这个角色拥有 1,2,3 这三个权限：

role_id	node_id	level
1	1	1
1	2	2
1	3	3

```

CREATE TABLE IF NOT EXISTS `think_access` (
  `role_id` smallint(6) unsigned NOT NULL,
  `node_id` smallint(6) unsigned NOT NULL,
  `level` tinyint(1) NOT NULL,
  `module` varchar(50) DEFAULT NULL,
  KEY `groupId` (`role_id`),
  KEY `nodeId` (`node_id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

```

管理员表（自己建）

```

CREATE TABLE IF NOT EXISTS `sh_user` (
  `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,
  `username` varchar(30) NOT NULL comment '用户名',
  `realname` varchar(30) NOT NULL comment '真实姓名',
  `password` char(32) not null comment '密码',
  `status` tinyint(1) unsigned default '1' comment '是否启用：1：启用0：禁用',
  PRIMARY KEY (`id`),
  KEY `status` (`status`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 ;

```

管理员所在角色表（一个管理员可以同时属于多个角色，如：1 管理员同时属于 1,2 两个角色：

role_id	user_id
1	1
2	1

```

CREATE TABLE IF NOT EXISTS `think_role_user` (
  `role_id` mediumint(9) unsigned DEFAULT NULL,
  `user_id` char(32) DEFAULT NULL,
  KEY `group_id` (`role_id`),
  KEY `user_id` (`user_id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

```

## 2. 检查一个管理员是否有某个权限的代码（读这些表的代码）

TP 中的 rbac 类中提供了哪几个方法：

说明：这个类中所有的方法都是静态的，那么就是说将来用时不需要 new 这个类的对象，如果要调用一个方法直接：\Think\Rbac:方法名();

- authenticate(\$where): 查询一个管理员的详细信息
- saveAccessList(\$userId): 取出一个管理员所有的权限并存到 SESSION 中
- checkAccess: 当前这个操作是否需要验证权限，因为我们可以设置哪些模块不需要验证
- checkLogin: 判断当前管理员有没有登录
- AccessDecision (\*): 根据数据库中的数据验证当前用户有没有操作当前方法的权限，执行任何操作之前先调用这个方法
- getAccessList: 根据管理员的 ID 查询数据库取出所有的权限
  - 原理：1.先取出一个管理员能访问的所有的模块是什么
    - 再循环所有的模块，取出每个模块下有权限访问的控制器

- b) 再循环所有可以访问的控制器：判断如果控制器叫 **PUBLIC** 那么所有的直接可以访问
- c) 再循环所有非 **PUBLIC** 的控制器，取出这个控制器中所有可以访问的方法

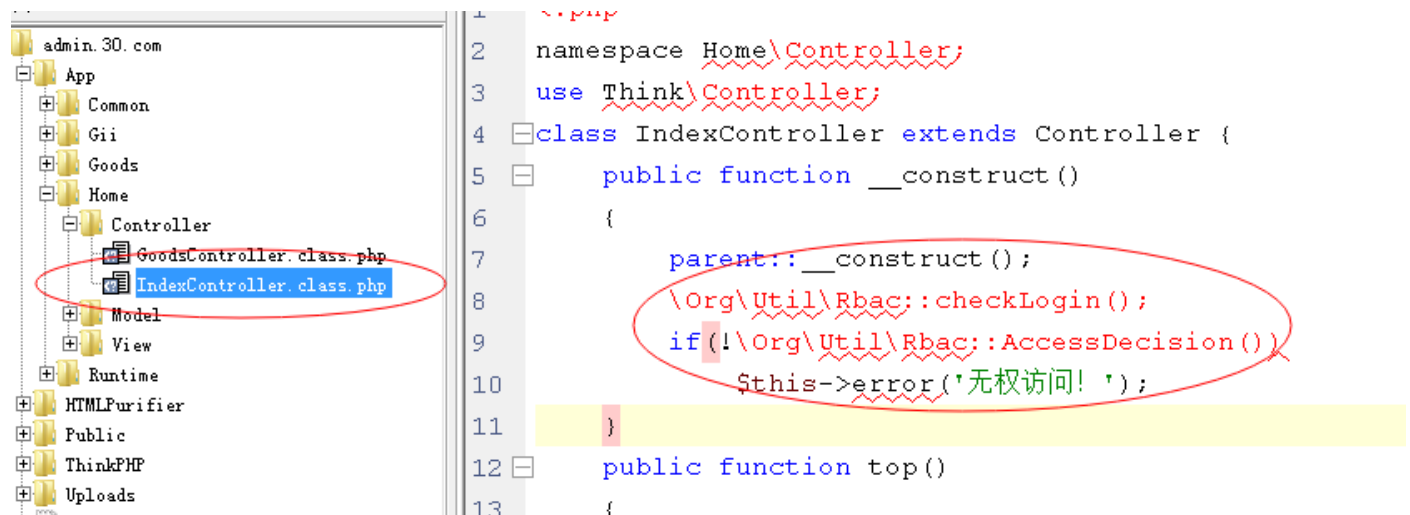
说明 2：如果要用这个类还需要在配置文件中添加几个配置项：

a. **USER\_AUTH\_MODEL**：管理员模块的名字，如：User

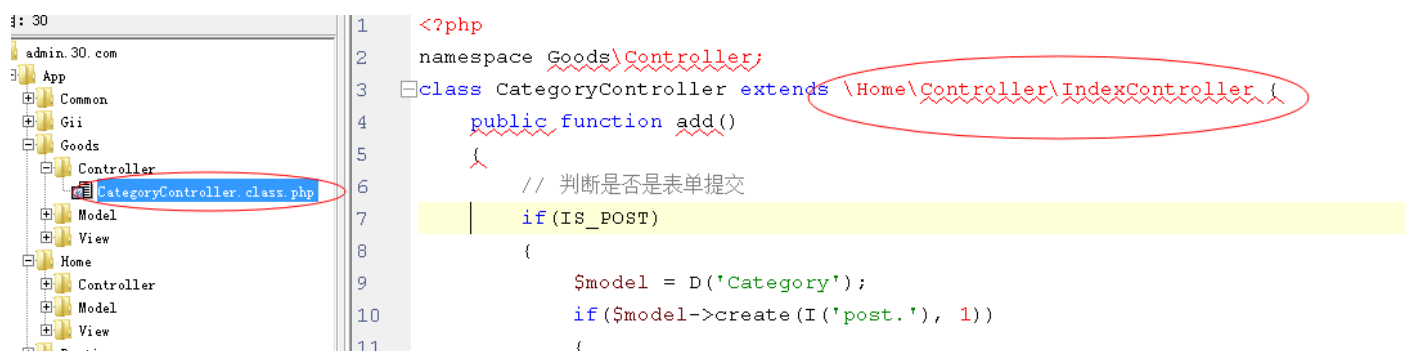
实际操作：如何实现 TP 中的验证：只需在任何操作之前先调用 **AccessDecision** 方法即可。

我们需要在一个公共的地方（每个方法执行之前都会执行的）。

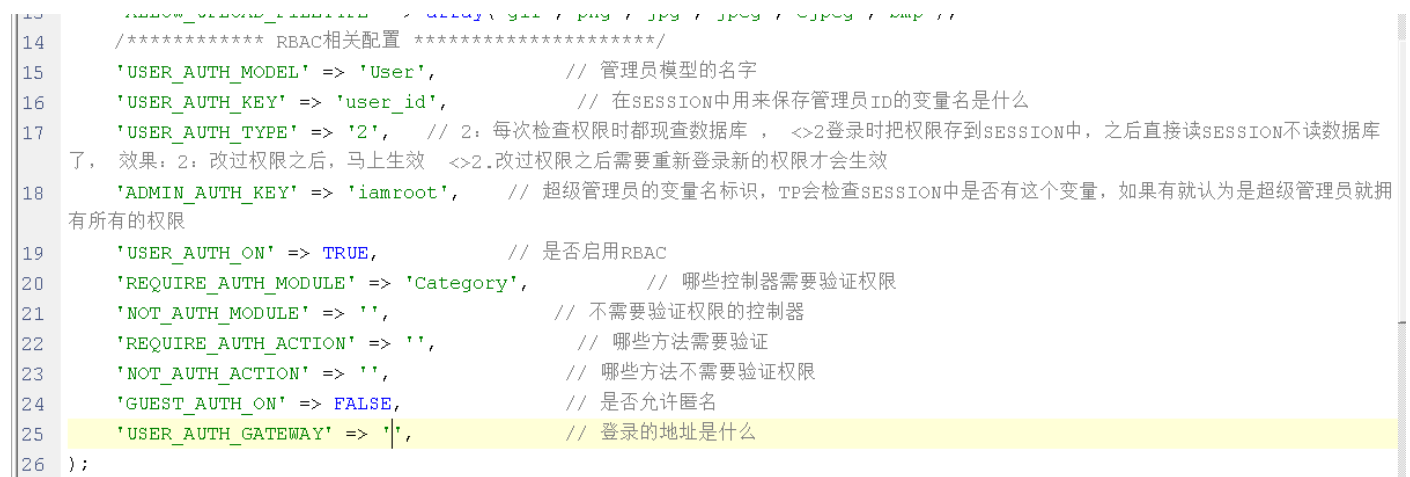
1. 在公共控制器中：



2. 让所有控制器都继承自这个控制器：



RBAC 相关的配置项：



注：其他的如角色管理、权限管理、管理员管理、权限的分配这些功能需要我们自己写。

# 权限管理、角色管理

TP 只提供表结构，具体操作需要我们自己写。

## 一、权限管理

说明：TP 中的权限只能做三级，虽然表结构中的 PID 可以实现无限级，但 TP 的代码中只读了前三级的权限，所以我们在做添加时也只能添加到三级权限

表结构：

```
CREATE TABLE IF NOT EXISTS `sh_node` (  
  `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,  
  `name` varchar(20) NOT NULL,  
  `title` varchar(50) DEFAULT NULL,  
  `status` tinyint(1) DEFAULT '0',  
  `remark` varchar(255) DEFAULT NULL,  
  `sort` smallint(6) unsigned DEFAULT NULL,  
  `pid` smallint(6) unsigned NOT NULL,  
  `level` tinyint(1) unsigned NOT NULL,  
  PRIMARY KEY (`id`),  
  KEY `level` (`level`),  
  KEY `pid` (`pid`),  
  KEY `status` (`status`),  
  KEY `name` (`name`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

实际操作

1. 使用代码生成器生成 sh\_node（权限表）表的代码，代码生成器的配置文件如下：

```
<?php  
return array(  
  'tableName' => 'sh_node', // 数据库中的表名  
  'moduleName' => 'Rbac', // 代码生成到的模块  
  'tableCnName' => '节点',  
  'insertFields' => "array('name','status','pid')", // 允许添加的表单字段  
  'updateFields' => "array('id','name','status','pid')", // 允许修改的表单中的字段  
  'validate' => "  
    array('name', 'require', '不能为空!', 1, 'regex', 3),  
    array('status', '/^\d+$/ ', '必须是一个整数!', 2, 'regex', 3),  
    array('pid', 'require', '不能为空!', 1, 'regex', 3),  
    array('pid', '/^\d+$/ ', '必须是一个整数!', 2, 'regex', 3),  
    ", // 模型中的表单验证规则
```

```

14 'fields' => array( // 表单中显示的字段
15     'pid' => array(
16         'text' => '上级节点',
17         'type' => 'select',
18         'dataSource' => array(
19             '$_selData=M(\`Node\`)->where("level<=2")->select();',
20             'id',
21             'name'
22         ),
23         'tip' => '',
24         'class' => 'required',
25     ),
26     'name' => array(
27         'text' => '节点名称',
28         'type' => 'text',
29         'tip' => '请输入',
30         'class' => 'required',
31     ),
32     'status' => array(
33         'text' => '是否启用',
34         'type' => 'radio',
35         'radioOptionValue' => array(
36             '1' => '启用',
37             '0' => '不启用',
38         ),
39         'tip' => '',
40         'class' => 'required',
41         'bizRule' => '$v[status]==1?"启用":"不启用";',
42     ),
43 ),
44 'search' => array(),
45 );

```

2. 修改生成的节点模型，添加表单验证（修改时要验证修改之后的权限级数不能越过3级）：

```

use Think\Model;
class NodeModel extends Model
{
    protected $insertFields = array('name','status','pid');
    protected $updateFields = array('id','name','status','pid');
    protected $_validate = array(
        array('name', 'require', '不能为空!', 1, 'regex', 3),
        array('status', '/^\d+$/ ', '必须是一个整数!', 2, 'regex', 3),
        array('pid', 'require', '不能为空!', 1, 'regex', 3),
        array('pid', '/^\d+$/ ', '必须是一个整数!', 2, 'regex', 3),
        array('pid', 'chkPid', '节点级数最多3级!', 1, 'callback', 3),
    );
}

```

```

public function chkPid($pid)
{
    if($pid > 0)
    {
        // 上级权限level
        $l = $this->field('level')->find($pid);
        // 当前权限级别
        $cur_level = $l['level']+1;
        if($cur_level == 2) // 如果当前是第二级，那么最多只能层子级，子级不能再有子级，如果有就是第四级了
        {
            $children = $this->where('pid='.I('post.id'))->select();
            foreach ($children as $k => $v)
            {
                if($this->where('pid='.$v['id'])->count() > 0)
                {
                    return FALSE;
                }
            }
            return TRUE;
        }
        elseif ($cur_level == 3) // 如果当前是第三级，那么就能有子级，否则就是第四级了越过了三级
        {
            if($this->where('pid='.I('post.id'))->count() > 0)
            {
                return FALSE;
            }
        }
        else
        {
            return FALSE;
        }
    }
    return TRUE;
}

```

3. 权限要做成无限级的功能，可以直接复制之前做好的无限级商品分类模型中的两个递归函数并做几处修改：

```

public function getChildren($catId)
{
    $data = $this->select();
    return $this->_getChildren($data, $catId, TRUE);
}

private function _getChildren($data, $parent_id=0, $isClear = FALSE)
{
    static $_ret = array();
    if($isClear)
    {
        $_ret = array();
    }
    foreach ($data as $k => $v)
    {
        if($v['pid'] == $parent_id)
        {
            $_ret[] = $v['id'];
            $this->_getChildren($data, $v['id']);
        }
    }
    return $_ret;
}

```

```

public function getNodeTree($level = 999)
{
    $data = $this->select();
    return $this->_reSort($data, 0, 1, $level);
}
private function _reSort($data, $parent_id=0, $level=1, $needLevel=3)
{
    if($level > $needLevel)
        return ;
    static $_ret = array();
    foreach ($data as $k => $v)
    {
        if($v['pid'] == $parent_id)
        {
            $_ret[] = $v;
            $this->_reSort($data, $v['id'], $level+1, $needLevel);
        }
    }
    return $_ret;
}

```

递归时取几级的数据，  
这个参数是在表单时用的，  
因为表单只能取出前  
两级的权限，所以加这个  
参数限制一下取的级数

4. 添加两个钩子函数在添加和修改之前先计算节点的 level 值:

```

public function _before_insert(&$data, $option)
{
    // 计算当前这个节点的level是多少
    if($data['pid'] == 0)
        $data['level'] = 1;
    else
    {
        // 先取出上级的level值
        $l = $this->field('level')->find($data['pid']);
        $data['level'] = $l['level']+1;
    }
}
public function _before_update(&$data, $option)
{
    // 计算当前这个节点的level是多少
    if($data['pid'] == 0)
        $data['level'] = 1;
    else
    {
        // 先取出上级的level值
        $l = $this->field('level')->find($data['pid']);
        $data['level'] = $l['level']+1;
    }
    // 当前节点所有子节点的level也要都跟着变
    $this->_dg_update(I('post.id'), $data['level']);
}

```

```

private function _dg_update($parent_id, $level)
{
    $children = $this->where('pid='.$parent_id)->select();
    foreach ($children as $k => $v)
    {
        $this->execute("UPDATE sh_node SET level=$level+1 WHERE id=$v[id]");
        $this->_dg_update($v['id'], $level+1);
    }
}

```

注意：修改时还要考虑到递归修改子节点的 level 值

5. 最后添加删除 钩子函数，在删除一个节点时也要删除所有子节点

```

protected function _before_delete($data)
{
    if(is_array($data['where']['id']))
    {
        $attr = explode(',', $data['where']['id'][1]); // 1,2,3,4,5
        // 循环每一个要删除的分类找出子分类
        foreach ($attr as $v)
        {
            $children = $this->getChildren($v);
            $children[] = $v;
            $children = implode(',', $children);
            $this->execute("DELETE FROM sh_node WHERE id IN($children)");
        }
    }
    else
    {
        // 先找出这个分类所有子分类的id
        $children = $this->getChildren($data['where']['id']);
        $children[] = $data['where']['id'];
        $children = implode(',', $children);
        $this->execute("DELETE FROM sh_node WHERE id IN($children)");
    }
}

```

6. 修改 add.html 添加的表单中上级节点只能是前两级的节点：

```

<td class="label">上级节点</td>
<td>
    <?php $selData=D('Node')->getNodeTree(2); ?>
    <select name="pid" class="required">
        <option value="0">请选择</option>
        <?php foreach ($selData as $k => $v): ?>
            <option value="<?php echo $v['id']; ?>"><?php echo str_repeat('-', ($v['le
;v['name']; ?></option>
        <?php endforeach; ?>
    </select>
    <span class="require-field">*</span>
</td>
</tr>
<tr>
    <td class="label">节点名称</td>
    <td>
        <input type="text" value="" />
    </td>
</tr>

```

7. 修改 save.html 修改的表单：



```

<input type="hidden" name="id" value="<?php echo $data['id']; ?>" />
<table cellpadding="1" cellspacing="3" width="100%">
    <tr>
        <td class="label">上级节点</td>
        <td>
            <?php $nodeModel = D('Node');$_selData=$nodeModel->getNodeTree(2); ?>
            <select name="pid"
            <option value="0">请选择</option>
            <?php
                $children = $nodeModel->getChildren($data['id']);
                foreach ($_selData as $k => $v):
                    if($data['id'] == $v['id'] || in_array($v['id'], $children))
                        continue ;
                    <option <?php if($data['pid']==$v['id']) echo 'selected="selected"' ?> value="<?php echo
eat('-', ($v['level']-1)*8).$v['name']; ?></option>
                <?php endforeach; ?>
            </select>

```

修改时上级节点不能是当前节点也不能是子节点

8. 在页面左侧添加权限管理的按钮：修改 Home/Index/menu.html:

```

<li class="explode" key="08_members" name="menu">
    权限管理
    <ul>
        <li class="menu-item"><a href="__APP__/Rbac/Node/1st" target="main-frame">节点列表</a></li>
        <li class="menu-item"><a href="__APP__/Rbac/Role/1st" target="main-frame">角色列表</a></li>
        <li class="menu-item"><a href="userMessage.html" target="main-frame">会员留言</a></li>
    </ul>
</li>

```

权限管理完成：效果：

## 二、角色管理

说明这个功能制作起来比较简单因为目前要改的东西不多，直接使用代码生成器生成即可：

表结构：

```

CREATE TABLE IF NOT EXISTS `sh_role` (
  `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(20) NOT NULL,
  `pid` smallint(6) DEFAULT NULL,
  `status` tinyint(1) unsigned DEFAULT NULL,
  `remark` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `pid` (`pid`),
  KEY `status` (`status`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8 ;

```

实际操作：1. 制作代码生成器的配置文件

```

return array(
    'tableName' => 'sh_role', // 数据库中的表名
    'moduleName' => 'Rbac', // 代码生成到的模块
    'tableCnName' => '角色',
    'insertFields' => "array('name','status')", // 允许添加的表单字段
    'updateFields' => "array('id','name','status')", // 允许修改的表单中的字段
    'validate' => "
        array('name', 'require', '角色名称不能为空!', 1, 'regex', 3),
        array('name', '', '角色名称已经存在!', 1, 'unique', 3),
        array('status', '/^\d+$/ ', '必须是一个整数!', 2, 'regex', 3),
        ", // 模型中的表单验证规则
    'fields' => array( // 表单中显示的字段
        'name' => array(
            'text' => '角色名称',
            'type' => 'text',
            'tip' => '请输入',
            'class' => 'required',
        ),
        'status' => array(
            'text' => '是否启用',
            'type' => 'radio',
            'radioOptionValue' => array(
                '1' => '启用',
                '0' => '不启用',
            ),
            'tip' => '',
            'class' => 'required',
            'bizRule' => '$v[status]==1?"启用":"不启用";',
        ),
    ),
    'search' => array(
        array('name', 'normal', '', 'like'),
    ),
);

```

2.直接代码生成器生成即可:

商品管理

商品列表

添加新商品

商品分类

商品品牌

商品回收站

订单管理

订单列表

订单查询

添加订单

发货单列表

退货单列表

权限管理

节点列表

角色列表

会员留言

管理中心-角色列表

添加角色

角色名称:

排序方式: ☐ 升序 ☒ 降序

搜索

<input type="checkbox"/>	角色名称	是否启用	操作
<input type="checkbox"/>	经理	不启用	编辑   移除
<input type="checkbox"/>	总监	启用	编辑   移除

删除所选

扩展: 列表页添加一个高亮显示的功能

1. 添加高亮的样式:

```
861
862 /** 一行高亮的样式 **/
863 tr.mouseon td{background:#EEE;}

861
862 /** 一行高亮的样式 **/
863 tr.mouseon td{background:#BBDDE5;}
```

2. 在 lst.html 中添加 JS 代码:

```
}
$( "tr.datatr" ).mouseover(function(){
    $( this ).addClass( "mouseon" );
});
$( "tr.datatr" ).mouseout(function(){
    $( this ).removeClass( "mouseon" );
});
});
```

完成效果:

<input type="checkbox"/>	节点名称	是否启用	操作
<input type="checkbox"/>	商品管理	1	编辑   移除
<input type="checkbox"/>	-----商品类型列表	1	编辑   移除
<input type="checkbox"/>	-----订单列表	1	编辑   移除
<input type="checkbox"/>	-----添加	1	编辑   移除
<input type="checkbox"/>	订单管理	1	编辑   移除

鼠标放上一行时背景色变灰。

# 权限的分配

现在角色管理和权限管理已经完成了，现在要做的是在添加和修改一个角色时可以为这个角色分配权限：做完之后的效果：

角色名称

经理

请输入 \*

是否启用

☐ 启用 ☒ 不启用 \*

选择权限

☐ 商品管理

☐ -----商品类型列表

☐ -----订单列表

☐ -----添加

☐ -----商品列表

☐ 订单管理

☐ -----订单列表

☐ -----修改

☐ -----添加

一、添加角色时分配权限

实际操作:

1. 修改添加和修改角色的表单, 列出所有的权限的树形结构并在前面制作复选框:

```
7 ~~~~~
8 <tr>
9
10 <td class="label">选择权限</td>
11
12 <td>
13     <?php $nodeModel = D('Node');
14     $data = $nodeModel->getNodeTree();
15     foreach ($data as $k => $v): ?>
16         <input class="nodechk" level="<?php echo $v['level']; ?>" type="checkbox" name="node_id[<?php echo $v['level']; ?>][<?php echo $v['id']; ?>]"
17         value="<?php echo $v['id']; ?>" /><?php echo str_repeat('-', ($v['level']-1)*10).$v['name']; ?><br />
18     <?php endforeach; ?>
19 </td>
20 </tr>
21
22 <tr>
23 <td colspan="2" align="center"><br />
24 <input type="submit" class="button" value=" 确定 " />
25 <input type="reset" class="button" value=" 重置 " />
26 </td>
27 </tr>
```

把权限的level值放到复选框上, 为了之后的JS代码用

给个样式, 为了JQ代码选中这些框

把level值放到name上这样提交表单时, 可以把level值和权限的id都提交上去

效果:

角色名称  请输入 \*

是否启用 ☒ 启用 ☐ 不启用 \*

选择权限 ☐ 商品管理

☐ -----商品类型列表

☐ -----订单列表

☐ -----添加

☐ 订单管理

2. 在页后面添加 JS 实现功能:
  - a. 当选中一个权限时, 这个权限所有的上级权限也选中
  - b. 当取消一个权限时, 这个权限所有下级权限也取消

```
// 选中所有的复选框
var allChkbox = $(".nodechk");
allChkbox.click(function(){
    // 当前按钮的level值
    var cur_level = $(this).attr('level');
    if($(this).attr('checked')) // 如果是选中一个权限
    {
        // 选出所有上级的框设置为选中 (上级权限: 1所有前面的权限, 2level值小于当前这个框的level值)
        $(this).prevAll(":checkbox").each(function(){ // 取出所有前面的复选框
            var level = $(this).attr('level'); // 取出前面每个的level值
            if(level < cur_level) // 如果level值小于当前的level值说明是上一级
            {
                $(this).attr("checked", "checked");
                if(level == 1) // 如果前面遇到一个顶级的那么就不用再向前找了, 说明已经到顶级了再向上就是别的权限了
                {
                    return false;
                }
            }
        });
    }
    else // 如果要取消一个权限
    {
        $(this).nextAll(":checkbox").each(function(){
            var level = $(this).attr('level');
            if(level > cur_level)
            {
                $(this).removeAttr("checked");
                if(level == 1)
                {
                    return false;
                }
            }
        });
    }
});
```

3. 处理表单把数据保存到数据库中:

一个角色拥有的权限是存在这个表中的（这个表是 TP 中 RBAC 类中给我们设计的）：

```
13 / ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ RBAC ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ /
14 # 一个角色所拥有的权限
15 CREATE TABLE IF NOT EXISTS `sh_access` (
16     `role_id` smallint(6) unsigned NOT NULL,
17     `node_id` smallint(6) unsigned NOT NULL,
18     `level` tinyint(1) NOT NULL,
19     `module` varchar(50) DEFAULT NULL,
20     KEY `groupId` (`role_id`),
21     KEY `nodeId` (`node_id`)
22 ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

在角色模型中添加一个添加之后钩子函数，在角色添加成功之后有了角色 ID 之后再接收这个角色的权限数据插入到中间表：

表单提交之后表单中数据结构如下：

```
'name' => string '33' (length=2)
'status' => string '1' (length=1)
'node_id' =>
    array
    1 =>
        array
        0 => string '4' (length=1)
    3 =>
        array
        0 => string '13' (length=2)
        1 => string '12' (length=2)
        2 => string '16' (length=2)
```

node\_id 中存的是表单选中的所有的权限的 ID，其中的键（1,3）存的是权限的 level 值，根据这个数组结构写代码循环每个权限插入到 access 表：

```

protected function _after_insert($data)
{
    $nodeId = I('post.node_id');
    if($nodeId)
    {
        $accModel = M('Access');
        // 循环每一个权限插入到角色权限表
        foreach ($nodeId as $level => $v)
        {
            foreach ($v as $k1 => $v1)
            {
                $accModel->add(array(
                    'role_id' => $data['id'],
                    'node_id' => $v1,
                    'level' => $level,
                ));
            }
        }
    }
}

```

到此添加角色时分配权限功能完成，三张表间关系：

```
mysql> SELECT * FROM sh_role;
+-----+-----+-----+-----+-----+
| id | name | pid | status | remark |
+-----+-----+-----+-----+
| 1 | 经理 | NULL | 0 | NULL |
| 2 | 总监 | NULL | 1 | NULL |
| 3 | 经理1 | NULL | 1 | NULL |
+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM sh_access;
+-----+-----+-----+-----+
| role_id | node_id | level | module |
+-----+-----+-----+-----+
| 3 | 4 | 1 | NULL |
| 3 | 10 | 1 | NULL |
| 3 | 11 | 2 | NULL |
| 3 | 14 | 2 | NULL |
| 3 | 12 | 3 | NULL |
| 3 | 15 | 3 | NULL |
+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM sh_node;
+-----+-----+-----+-----+-----+-----+-----+
| id | name | title | status | remark | sort | pid | level |
+-----+-----+-----+-----+-----+-----+-----+
| 16 | 修改 | NULL | 1 | NULL | NULL | 14 | 3 |
| 13 | 订单列表 | NULL | 1 | NULL | NULL | 11 | 3 |
| 4 | 商品管理 | NULL | 1 | NULL | NULL | 0 | 1 |
| 15 | 添加 | NULL | 1 | NULL | NULL | 14 | 3 |
| 14 | 订单列表 | NULL | 1 | NULL | NULL | 10 | 2 |
| 12 | 添加 | NULL | 1 | NULL | NULL | 11 | 3 |
+-----+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

## 二、角色的删除

删除一个角色时应该把 access 表中这个角色对应的权限也一直删除掉，所以修改角色模型添加删除的钩子函数：

```
protected function _before_delete($data)
{
    $acc = M('Access');
    // 如果是批量删除
    if(is_array($data['where']['id']))
        $acc->where("role_id IN({$data['where']['id'][1]})")->delete();
    else
        $acc->where('role_id='.$data['where']['id'])->delete();
}
```

### 三、修改角色信息时的权限分配

1. 修改角色和添加角色思路相同都是要先制作表单，在表单中取出所有的权限复制复选框，只是在修改的表单时要让当前角色所拥有的权限的复选框默认为选中状态：

```
44         <tr>
45             <td class="label">选择权限</td>
46             <td>
47                 <?php
48                 $accModel = M('Access');
49                 // 先取出当前角色所拥有的权限的id并转成一维数组
50                 $roleData = $accModel->field('GROUP_CONCAT(node_id) nid')->where('role_id='.$data['id'])->find();
51                 $roleData = explode(',', $roleData['nid']);
52
53                 $nodeModel = D('Node');
54                 $data = $nodeModel->getNodeTree();
55                 foreach ($data as $k => $v):
56                     // 判断这个角色有没有这个权限
57                     if(in_array($v['id'], $roleData))
58                         $check = 'checked="checked"';
59                     else
60                         $check = '';
61                 ?>
62                 <input <?php echo $check; ?> class="nodechk" level="<?php echo $v['level']; ?>" type="checkbox" name="n
ode_id[<?php echo $v['level']; ?>]" value="<?php echo $v['id']; ?>" /><?php echo str_repeat('-', ($v['level']-1)*10).$v['name'
]; ?><br />
63                 <?php endforeach; ?>
64             </td>
65         </tr>
```

先取出当前角色现在拥有的权限

判断这个角色有没有这个权限

完成，现在点击修改角色时，当前角色拥有的权限默认是选中的状态。

2. 最后提交表单时把权限的修改更新到数据库中

思路：修改权限时先把当前角色之前所拥有的权限数据删除掉，然后把修改之后的权限当作新的权限重新添加一份即可：



```

public function _before_update(&$data, $option)
{
    // 删除当前角色所有的权限
    $accModel = M('Access');
    $accModel->where('role_id='.I('post.id'))->delete();
    // 重新添加新权限
    $nodeId = I('post.node_id');
    if($nodeId)
    {
        // 循环每一个权限插入到角色权限表
        foreach ($nodeId as $level => $v)
        {
            foreach ($v as $k1 => $v1)
            {
                $accModel->add(array(
                    'role_id' => I('post.id'),
                    'node_id' => $v1,
                    'level' => $level,
                ));
            }
        }
    }
}

```

完成！

## 管理员管理

这本课实现的功能：

1. 管理员的 CRUD
  - a) 超级管理员 (ID=1)的账号不能被删除，也不能被禁用
  - b) 修改账号时如果密码为空，代表不修改密码
2. 后台的登录功能

实际操作

### 一、超级管理员的 CRUD

---

表结构：

```

99  ) ENGINE=MyISAM  DEFAULT CHARSET=utf8 ;
100 # 一个管理员属于哪个角色
101 CREATE TABLE IF NOT EXISTS `sh_role_user` (
102   `role_id` mediumint(9) unsigned DEFAULT NULL,
103   `user_id` char(32) DEFAULT NULL,
104   KEY `group_id` (`role_id`),
105   KEY `user_id` (`user_id`)
106 ) ENGINE=MyISAM  DEFAULT CHARSET=utf8;
107 # 管理员
108 CREATE TABLE IF NOT EXISTS `sh_user` (
109   `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,
110   `username` varchar(30) NOT NULL comment '用户名',
111   `realname` varchar(30) NOT NULL comment '真实姓名',
112   `password` char(32) not null comment '密码',
113   `status` tinyint(1) unsigned default '1' comment '是否启用: 1: 启用0: 禁用',
114   `is_supervisor` enum("是","否") not null default '否' comment '是否超级管理员',
115   PRIMARY KEY (`id`),
116   KEY `status` (`status`)
117 ) ENGINE=MyISAM  DEFAULT CHARSET=utf8 ;
118 INSERT INTO sh_user values(1,'admin','admin','16c450b7f77e73e74deef16a7f5d91f1',1,'是');
119

```

这里直接向管理员表中初始化了一个 ID=1 的超级管理员（用户名密码都是 admin）这个账号是不能被删除和禁用的，这个密码是使用的 MD5 加密，因为密码如果简单在 MD5 加密之后是可以被破解的，所以我们为了让密码更安全我们在加密时会在原密码上加一个密钥，来增密码的复杂程度：

在配置文件中定义加密的密钥：

```

22  'REQUIRE_AUTH_ACTION' => '', // 哪些方法需要验证权限
23  'NOT_AUTH_ACTION' => '', // 哪些方法不需要验证权限
24  'GUEST_AUTH_ON' => FALSE, // 是否允许匿名
25  'USER_AUTH_GATEWAY' => '/Home/Login/login', // 登录的地址是什么
26  /***** md5加密的密钥 *****/
27  'MD5_KEY' => '!343!129fd$fd_fds=+43>?lg',
28 );

```

1. 管理员的 CRUD--》直接代码生成器生成即可  
生成代码的配置文件如下：

```

<?php
return array(
    'tableName' => 'sh_user', // 数据库中的表名
    'moduleName' => 'Rbac', // 代码生成到的模块
    'tableCnName' => '管理员',
    'insertFields' => "array('username','realname','password','status')", // 允许添加的表单字段
    'updateFields' => "array('id','username','realname','password','status')", // 允许修改的表单中的字段
    'validate' => "
        array('username','require','用户名不能为空!', 1, 'regex', 3),
        array('username','','用户名已经存在!', 1, 'unique', 3),
        array('realname','require','真实姓名不能为空!', 1, 'regex', 3),
        array('password','require','密码不能为空!', 1, 'regex', 3),
        array('status','/^\\d+$/','是否启用: 1: 启用0: 禁用必须是一个整数!', 2, 'regex', 3),
    ", // 模型中的表单验证规则

```

```

15     'fields' => array( // 表单中显示的字段
16         'username' => array(
17             'text' => '用户名',
18             'type' => 'text',
19             'tip' => '请输入',
20             'class' => 'required',
21         ),
22         'realname' => array(
23             'text' => '真实姓名',
24             'type' => 'text',
25             'tip' => '请输入',
26             'class' => 'required',
27         ),
28         'password' => array(
29             'text' => '密码',
30             'type' => 'password',
31             'tip' => '请输入',
32             'class' => 'required',
33         ),
34         'status' => array(
35             'text' => '是否启用',
36             'type' => 'radio',
37             'radioOptionValue' => array(
38                 '1' => '启用',
39                 '0' => '不启用',
40             ),
41             'tip' => '',
42             'class' => 'required',
43             'bizRule' => '$v[status]==1?"启用":"不启用";',
44         ),
45         'is_supervisor' => array(
46             'text' => '是否超级管理员',
47             'type' => 'radio',
48             'radioOptionValue' => array(
49                 '否' => '否',
50                 '是' => '是',
51             ),
52             'tip' => '',
53             'class' => 'required',
54         ),
55     ),

```

```

56     'search' => array(
57         array('username', 'normal', '', 'like'),
58         array('realname', 'normal', '', 'like'),
59     ),
60 };

```

## 2. 生成代码之后如修改

### a) 修改管理员模型的表单验证功能，修改密码时可以为空：

```

namespace Adminuser\Model;
use Think\Model;
class UserModel extends Model
{
    protected $insertFields = array('username','realname','password','status','is_supervisor');
    protected $updateFields = array('id','username','realname','password','status','is_supervisor');
    protected $_validate = array(
        array('username', 'require', '用户名不能为空!', 1, 'regex', 3),
        array('username', '', '用户名已经存在!', 1, 'unique', 3),
        array('realname', 'require', '真实姓名不能为空!', 1, 'regex', 3),
        array('password', 'require', '密码不能为空!', 1, 'regex', 1),
        array('status', '/^\d+$/ ', '是否启用: 1: 启用 0: 禁用必须是一个整数!', 2, 'regex', 3),
    );
}

public function search()

```

1代表只有添加时才验证，修改时不验证，因为修改时可以不修改密码

### b) 添加钩子函数：把管理员的密码加密：

```

public function _before_insert(&$data, $option)
{
    $data['password'] = md5($data['password'] . C('MD5_KEY'));
}

```

### c) 修改时如果是超级管理员不允许被禁用：

```

public function _before_update(&$data, $option)
{
    // 判断如果修改超级管理员，不能修改降级和状态
    if(I('post.id') == 1)
    {
        unset($data['is_supervisor']);
        unset($data['status']);
    }
    if($data['password'])
        $data['password'] = md5($data['password'] . C('MD5_KEY'));
    else
        unset($data['password']);
}

```

## 3. 修改管理员控制器中的删除方法：不能删除超级管理员

```

,
public function del($id)
{
    if($id > 1)
    {
        $model = D('User');
        $model->delete($id);
        $this->success('删除成功!', U('lst'));
        exit;
    }
    else
        $this->error('无法删除超级管理员!');

}
public function bdel()
{
    $delid = I('post.delid');
    if($delid)
    {
        // 判断管理员中有没有1这个管理员
        foreach ($delid as $id)
        {
            if($id == 1)
                $this->error('超级管理员不能被删除!');
        }
        $delid = implode(',', $delid);
        $model = D('User');
    }
}

```

到此管理员的 CRUD 完成!

## 二、制作登录功能

### 1. 修改管理员模型：添加一个 login 方法：

```

public function login()
{
    $username = I('post.username');
    $password = I('post.password');
    $chkCode = I('post.chkCode');
    if($username && $password && $chkCode)
    {
        $verify = new \Think\Verify();
        if(!$verify->check($chkCode))
        {
            $this->error = '验证码不正确!';
            return FALSE;
        }
        // 验证用户名是否存在
        $user = $this->where("username='$username'" )->find();
        if($user)
        {
            if($user['status'] == 0)
            {
                $this->error = '该账号被禁用!';
                return FALSE;
            }
            if($user['password'] != md5($password . C('MD5_KEY')))
            {
                $this->error = '密码不正确!';
                return FALSE;
            }

            /***** 登录成功 *****/
            // 以下两行要和配置文件中变量名匹配，否则 TP中的RBAC无法验证权限
            session('user_id', $user['id']); // 管理员的id
            // 是否超级管理员
            if($user['is_supervisor'] == '是')
                session('iamroot', 1);
            session('username', $user['username']);
            session('realname', $user['realname']);
            return TRUE;
        }
        else
        {
            $this->error = '用户名不存在!';
            return FALSE;
        }
    }
    else
    {
        $this->error = '用户名密码验证码不能为空!';
        return FALSE;
    }
}
}

```

注意：这里登录成功之后要向 SESSION 中存几个变量其中两个变量要和配置文件中的两个变量对应，否则 TP 中 RBAC 会失效：

```

13 'ALLOW_UPLOAD_FILETYPE' => array('gif','png','jpg','jpeg','ejpeg','bmp'),
14 /***** RBAC相关配置 *****/
15 'USER_AUTH_MODEL' => 'User', // 管理员模型的名字
16 'USER_AUTH_KEY' => 'user_id', // 在SESSION中用来保存管理员ID的变量名是什么
17 'USER_AUTH_TYPE' => '2', // 2: 每次检查权限时都现查数据库, <2. 登录时把权限存到SESSION
    了, 效果: 2: 改过权限之后, 马上生效 <2. 改过权限之后需要重新登录新的权限才会生效
18 'ADMIN_AUTH_KEY' => 'iamroot', // 超级管理员的变量名标识, TP会检查SESSION中是否有这个变
    有所有的权限
19 'USER_AUTH_ON' => TRUE, // 是否启用RBAC
20 'REQUIRE_AUTH_MODULE' => '', // 哪些控制器需要验证权限
21 'NOT_AUTH_MODULE' => '', // 不需要验证权限的控制器
22 'REQUIRE_AUTH_ACTION' => '', // 哪些方法需要验证
23 'NOT_AUTH_ACTION' => '', // 哪些方法不需要验证权限
24 'GUEST_AUTH_ON' => FALSE, // 是否允许匿名
25 'USER_AUTH_GATEWAY' => '/Home/Login/login', // 登录的地址是什么
26 /***** md5加密的密钥 *****/
27 'MD5_KEY' => '!343!129fd$fd fds=+43>?1q',

```

## 2. 创建一个新的 login 的控制器实现登录功能

项目: 30

- admin.30.com
  - App
    - Adminuser
    - Common
    - Gii
    - Goods
    - Home
      - Controller
        - IndexController.class.php
        - LoginController.class.php
      - View
    - Rbac
    - Runtime
  - HTMLPurifier
  - Public
  - ThinkPHP
  - Uploads
    - db.sql
    - db1.sql
    - index.php
  - rbac

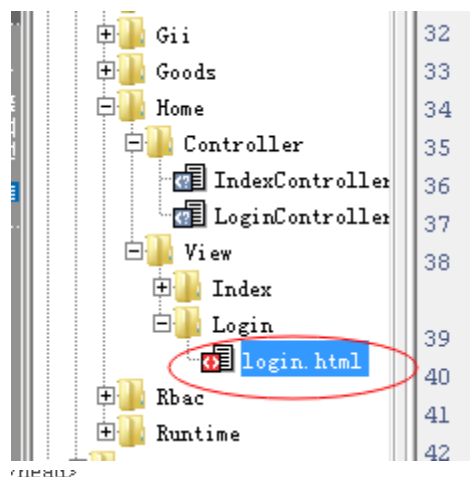
```

1 <?php
2 namespace Home\Controller;
3 use Think\Controller;
4 class LoginController extends Controller {
5     public function login()
6     {
7         if(IS_POST)
8         {
9             $model = D('Adminuser/User');
10            if($model->login() == TRUE)
11            {
12                $this->success('登录成功!', U('Home/Index/index'));
13                exit;
14            }
15            else
16            {
17                $error = $model->getError();
18                $this->error($error);
19            }
20        }
21        $this->display();
22    }
23    // 生成验证码的图片
24    public function getImg()
25    {
26        $Verify = new \Think\Verify();
27        $Verify->entry();
28    }
29 }

```

获取登录失败的原因

## 3. 复制模板中的 login.html 模板到项目中做为登录表单, 并修改几处:



```

</menu>
body style="background: #278296;color:white">
<form method="post" action="__SELF__">
  <table cellspacing="0" cellpadding="0" style="margin-top:100px" align="center">
    <tr>
      <td>
        
      </td>
      <td style="padding-left: 50px">
        <table>
          <tr>
            <td>管理员姓名: </td>
            <td>
              <input type="text" name="username" />
            </td>
          </tr>
          <tr>
            <td>管理员密码: </td>
            <td>
              <input type="password" name="password" />
            </td>
          </tr>
          <tr>
            <td>验证码: </td>
            <td>
              <input type="text" name="chkCode" class="capital" />
            </td>
          </tr>
          <tr>
            <td colspan="2" align="right">
              
            </td>
          </tr>
          <tr>
            <td colspan="2">
              <input type="submit" value="进入管理中心" class="button" />
            </td>
          </tr>
        </table>
      </td>
    </tr>
  </table>
</form>
</body>

```

到此登录功能完成！



功能说明:

1. 现在可以在配置文件中开启权限验证的功能并设置登录的网关地址:

```

3      'USER_AUTH_ON' => TRUE,           // 是否启用RBAC
4      'REQUIRE_AUTH_MODULE' => '',     // 哪些控制器需要验证权限
5      'NOT_AUTH_MODULE' => '',         // 不需要验证权限的控制器
6      'REQUIRE_AUTH_ACTION' => '',     // 哪些方法需要验证
7      'NOT_AUTH_ACTION' => '',         // 哪些方法不需要验证权限
8      'GUEST_AUTH_ON' => FALSE,        // 是否允许匿名
9      'USER_AUTH_GATEWAY' => '/Home/Login/login', // 登录的地址是什么
10     /***** md5加密的密钥 *****/
11     'MD5_KEY' => '!343!129fd$fd_fds=+43>?lg',
12 };

```

2. 开启之后如果没有登录是无法登录到后台的, 现在可以使用 admin/admin 这个超级管理员账号登录后台
3. 因为现在还没有做账号分配指定角色的功能, 所以现在只有超级管理员才可以登录后台, 其他添加的管理员无法登录后台, 因为没有分配角色就没有任何权限。
4. 可以在配置文件中设置不需要权限验证的控制器和必须要验证权限的控制器:

```

1      'USER_AUTH_ON' => TRUE,           // 是否启用RBAC
2      'REQUIRE_AUTH_MODULE' => '',     // 哪些控制器需要验证权限
3      'NOT_AUTH_MODULE' => '',         // 不需要验证权限的控制器
4      'REQUIRE_AUTH_ACTION' => '',     // 哪些方法需要验证
5      'NOT_AUTH_ACTION' => '',         // 哪些方法不需要验证权限
6      'GUEST_AUTH_ON' => FALSE,        // 是否允许匿名

```

现在 RBAC 完成了 80%了, 还差最后一个功能就是指定一个管理员属于哪个角色。这个下次课再做。  
这个功能同学们也可以自己在代码上写写看看能不能自己加上这个功能。

## 管理员所在的角色

现在权限功能我们还差一个重要的功能, 就是指定一个管理员属于哪些个角色的功能。

实际操作:

1. 在添加和修改管理员的表单中取出所有的角色制作一个复选框, 可以选择这个角色所属于的角色 (一个管理员可以属于多个角色)

```

<table cellpadding="3" cellspacing="1" width="100%">
<tr>
<td class="label">所在角色</td>
<td>
<?php $roleModel = M('Role');
$rolData = $roleModel->select(); ?>
<?php foreach ($rolData as $k => $v): ?>
<input type="checkbox" name="role_id[]" value="<?php echo $v['id']; ?>" /><?php echo $v['name']; ?>
<?php endforeach; ?>
<span class="require-field">*</span>
</td>
</tr>
</table>

```

效果:

---

所在角色

☐ 经理 ☒ 总监 \*

用户名

请输入 \*

真实姓名

请输入 \*

密码

请输入 \*

是否启用

☒ 启用 ☐ 不启用 \*

是否超级管理员

☒ 否 ☐ 是 \*

2 点击提交之后，接收选中的角色，插入到下面这个管理所在角色表，一个角色对应一条记录

说明：一个管理员可以同时属于多个角色，一个角色也可以同时有多个管理员，所以角色表和管理员表是多对多的关系，所以我们建了以下的中间表用来保存一个管理员所属于的角色：

```
1 # 一个管理员属于哪些角色
2 CREATE TABLE IF NOT EXISTS `sh_role_user` (
3     `role_id` mediumint(9) unsigned DEFAULT NULL,
4     `user_id` char(32) DEFAULT NULL,
5     KEY `group_id` (`role_id`),
6     KEY `user_id` (`user_id`)
7 ) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

向这个表插入记录时需要先有 `user_id` 管理员的 ID，所以要在添加了管理员之后有了管理员 ID 之后再处理所在的角色数据，所以要在后置钩子函数中：

```
,
protected function _after_insert($data, $option)
{
    $roleId = I('post.role_id');
    if($roleId)
    {
        $ruModel = M('RoleUser');
        // 循环表单中提交的多个角色的ID，循环插入到表中
        foreach ($roleId as $k => $v)
        {
            $ruModel->add(array(
                'role_id' => $v,
                'user_id' => $data['id'],
            ));
        }
    }
}
```

这样就可以在添加管理员时指定管理员所在的角色了。

3. 同样的道理修改一下修改管理员的表单：

```

30 <td class="label">所在角色</td>
31 <td>
32     <?php
33     // 先取出当前管理员现在属于哪个角色
34     $ruModel = M('RoleUser');
35     $_data = $ruModel->field('GROUP_CONCAT(role_id) rid')->where('user_id='.$data['id'])->find();
36     $_data = explode(',', $_data['rid']);
37
38     $roleModel = M('Role');
39     $rolData = $roleModel->select(); ?>
40     <?php foreach ($rolData as $k => $v):
41         if(in_array($v['id'], $_data))
42             $check = 'checked="checked"';
43         else
44             $check = '';
45     ?>
46     <input <?php echo $check; ?> type="checkbox" name="role_id[]" value="<?php echo $v['id']; ?>" /><?p
47 hp echo $v['name']; ?>
48     <?php endforeach; ?>
49     <span class="require-field">*</span>
50 </td>
</tr>

```

判断如果当前修改的管理员属于这个角色，那么这个角色的复选框默认是选中的状态

4. 在管理员模型中添加修改之前的钩子函数:

```

public function _before_update(&$data, $option)
{
    // 判断如果修改超级管理员，不能修改降级和状态
    if(I('post.id') == 1)
    {
        unset($data['is_supervisor']);
        unset($data['status']);
    }
    if($data['password'])
        $data['password'] = md5($data['password'] . C('MD5_KEY'));
    else
        unset($data['password']);
    /***** 处理角色数据 *****/
    $roleId = I('post.role_id');
    $ruModel = M('RoleUser');
    // 先删除之前的数据
    $ruModel->where('user_id='.I('post.id'))->delete();
    if($roleId)
    {
        foreach ($roleId as $k => $v)
        {
            $ruModel->add(array(
                'role_id' => $v,
                'user_id' => I('post.id'),
            ));
        }
    }
}

```

5. 在删除一个管理员时，把这个管理员对应的所在角色的数据也删除掉:

```
protected function _before_delete($data)
{
    $ruModel = M('RoleUser');
    // 如果是批量删除
    if(is_array($data['where']['id']))
        $ruModel->where("user_id IN({$data['where']['id'][1]})")->delete();
    else
        $ruModel->where('user_id='.$data['where']['id'])->delete();
}
```

到此，管理员指定角色的功能完成！

现在权限有 BUG：上次课讲权限管理时，有个地方我讲错了，我上次以为 title 字段没用，只要有个 name 就行了，但实际上 name 字段和 title 都是必要的。

name：存对应的 TP 中节点的名字（模块、控制器、方法名）

title：存节点的中文名：

```
CREATE TABLE IF NOT EXISTS `sh_node` (
  `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(20) NOT NULL,
  `title` varchar(50) DEFAULT NULL,
  `status` tinyint(1) DEFAULT '0',
  `remark` varchar(255) DEFAULT NULL,
  `sort` smallint(6) unsigned DEFAULT NULL,
  `pid` smallint(6) unsigned NOT NULL,
  `level` tinyint(1) unsigned NOT NULL,
  PRIMARY KEY (`id`),
  KEY `level` (`level`),
  KEY `pid` (`pid`),
  KEY `status` (`status`),
  KEY `name` (`name`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

对应节点的名称（TP中的名称）如程序上的  
模块名称或者控制器名或者方法名，如：  
如果这个权限level=1, name应该是模块名  
level=2, name应该是控制器名  
level=3, name应该是方法名

权限的中文名

所以我们要把之前的功能重新修改一下才可以：

1. 修改权限节点的添加的表单，把标题字段和名字字段都加上：



2. 同样的道理把修改的表单也改一下改成两个字段：



3. 修改节点列表页中列表页中显示出标题和名称：

```
<table cellpadding="3" cellspacing="1">
<tr>
<th width="5"><input id="selAll" type="checkbox" /></th>
<th width="150">模块名/控制器名/方法名</th>
<th>节点标题</th>
<th>是否启用</th>
<th>操作</th>
</tr>
<?php foreach ($data as $k => $v): ?>
<tr class="datatr">
<td align="center"><input type="checkbox" name="delid[]" value="<?php echo $v['id']; ?>" /></td>
<td align="center"><?php echo $v['name']; ?></td>
<td align="center"><?php echo str_repeat('-', ($v['level']-1) * 8).$v['title']; ?></td>
<td align="center"><?php echo $v['status'] == 1 ? '启用' : '不启用'; ?></td>
<td align="center">
<a href="__CONTROLLER__/edit/id/<?php echo $v['id']; ?>" title="编辑">编辑</a> |
<a href="__CONTROLLER__/del/id/<?php echo $v['id']; ?>" onclick="return confirm('确定要删除吗?');" title="移除">移除</a>
</td>
</tr>
<?php endforeach; ?>
<tr>
<td colspan="5"><input type="submit" value="删除所选" /></td>
</tr>
</table>
```

最终节点管理的效果：（即有中文名，又有对应 TP 中节点的名字）

<div>商品模块</div> <div>商品分类管理</div> <div>商品类型管理</div> <div>商品类型列表</div> <div>权限管理</div> <div>节点列表</div> <div>角色列表</div> <div>管理员列表</div> <div>订单管理</div> <div>商品类型列表</div>	后台中心 节点列表			
	<input type="checkbox"/>	模块名/控制器名/方法名	节点标题	是否启用
	<input type="checkbox"/>	Goods	商品模块	启用
	<input type="checkbox"/>	Category	商品分类管理	启用
	<input type="checkbox"/>	bdel	批量删除	启用
	<input type="checkbox"/>	lst	列表	启用
	<input type="checkbox"/>	del	删除	启用
	<input type="checkbox"/>	edit	修改	启用
	<input type="checkbox"/>	add	添加	启用
	<input type="checkbox"/>	Type	商品类型管理	启用
	<input type="checkbox"/>	add	添加	启用
	<input type="checkbox"/>	Type	商品类型列表	启用
	<input type="checkbox"/>	add	添加	启用
	<input type="checkbox"/>	edit	修改	启用

## 左侧按钮按权限显示

后面左侧只显示当前管理员有权限访问的按钮。

原理：因为 TP 中 RBAC 的特点，我们左侧只取出前两级的节点（模块名、控制器名），当用户点击一个按钮时，我们固定跳到这个地址：模块名/控制器名/lst。就是第三级的方法名写死就是 lst。

实现：

1. 在管理员模型(Adminuser/Model/UserModel.class.php)中添加一个方法（获取当前管理员所有可以访问的前两级按钮）

```
// 获取当前管理员所拥有的所有的权限
public function getPrivilege()
{
    // 取出当前登录的管理员的ID
    $userId = $_SESSION['user_id'];
    // 如果是超级管理员就从节点表是取出所有启用的前两级权限
    if(isset($_SESSION['iamroot']))
    {
        // 先取出所有顶级的节点
        $sql = 'SELECT id,title,name FROM sh_node WHERE status=1 AND level=1';
        $data = $this->query($sql);
        // 循环每个顶级节点取出二级的并放到children中
        foreach ($data as $k => $v)
        {
            $sql = 'SELECT id,title,name FROM sh_node WHERE status=1 AND level=2 AND pid='.$v['id'];
            $data[$k]['children'] = $this->query($sql);
        }
        return $data;
    }
}
```

```

81     else
82     {
83         // 如果不是超级管理员就执行SQL, 根据当前管理员的ID取出这个管理员所拥有的权限
84         $sql = "select node.id,node.name,node.title from sh_role as role,sh_role_user as user,sh_access as access,sh_node as node where user.user_id='{$userId}' and user.role_id=role.id and ( access.role_id=role.id or (access.role_id=role.pid and role.pid!=0 ) ) and role.status=1 and access.node_id=node.id and node.level=1 and node.status=1";
85         $data = $this->query($sql);
86         foreach ($data as $k => $v)
87         {
88             $sql = "select node.id,node.name,node.title from sh_role as role,sh_role_user as user,sh_access as access,sh_node as node where user.user_id='{$userId}' and user.role_id=role.id and ( access.role_id=role.id or (access.role_id=role.pid and role.pid!=0 ) ) and role.status=1 and access.node_id=node.id and node.level=2 and node.pid={$v['id']} and node.status=1";
89             $data[$k]['children'] = $this->query($sql);
90         }
91         return $data;
92     }

```

2. 在左侧按钮的页面(Home/Index/menu.html)中执行上面的方法取出当前管理员有权限访问的按钮并循环输出:

```

120     <span class="tab-front" id="menu-tab">菜单</span>
121     </p>
122 </div>
123 <?php
124 $userModel = D('Adminuser/User');
125 $allPri = $userModel->getPrivilege();
126 ?>
127 <div id="main-div">
128     <div id="menu-list">
129         <ul id="menu-ul">
130             <?php foreach ($allPri as $k => $v): ?>
131                 <li class="explode">
132                     <?php echo $v['title']; ?>
133                     <ul>
134                         <?php foreach ($v['children'] as $k1 => $v1):
135                             $url = U($v['name'].'/'.$v1['name'].'/lst');
136                             ?>
137                             <li class="menu-item"><a href="<?php echo $url; ?>" target="main-frame"><?php echo $v1['title']; ?></a></li>
138                         <?php endforeach; ?>
139                     </ul>
140                 </li>
141             <?php endforeach; ?>

```

循环顶级节点

循环二级节点

拼出这个按钮点击时跳转的地址

到此完成, 左侧只会显示有权限访问的按钮。

扩展: 现在左侧的按钮是通过查询 node 表中的记录显示出来的, 也就是说, 以后我们每次在项目中添加了新的模块、控制器、方法时, 都要把这些节点插入到节点表中才可以。如果手动添加比较麻烦, 我们可以修改代码生成器, 在生成代码的同时, 把对应节点的数据自动插入到节点表中:

1. 修改代码生成器模板中的 config.php 文件, 在生成配置文件时, 在配置文件中添加一个新的配置项, 用来指定生成的模块的节点的中文名

```

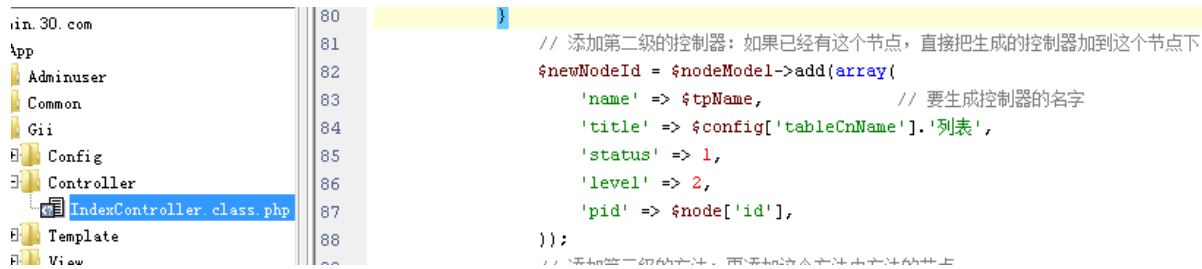
1 return array(
2     'tableName' => '<?php echo $__v; ?>', // 数据库中的表名
3     'moduleName' => '<?php echo $tpName; ?>', // 代码生成到的模块
4     'moduleCnName' => '', // 模块的中文名-这个名字就是插入节点表时的模块节点名字
5     'tableCnName' => '<?php echo $tableInfo['Comment']; ?>',
6
7     <?php
8     $fields_arr = array();
9     foreach ($tableFields as $k => $v)
10     {
11         if($v['Field'] == 'id')
12             continue;
13         $fields_arr[] = "{$v['Field']}";
14     }
15
16     return $fields_arr;
17 }

```

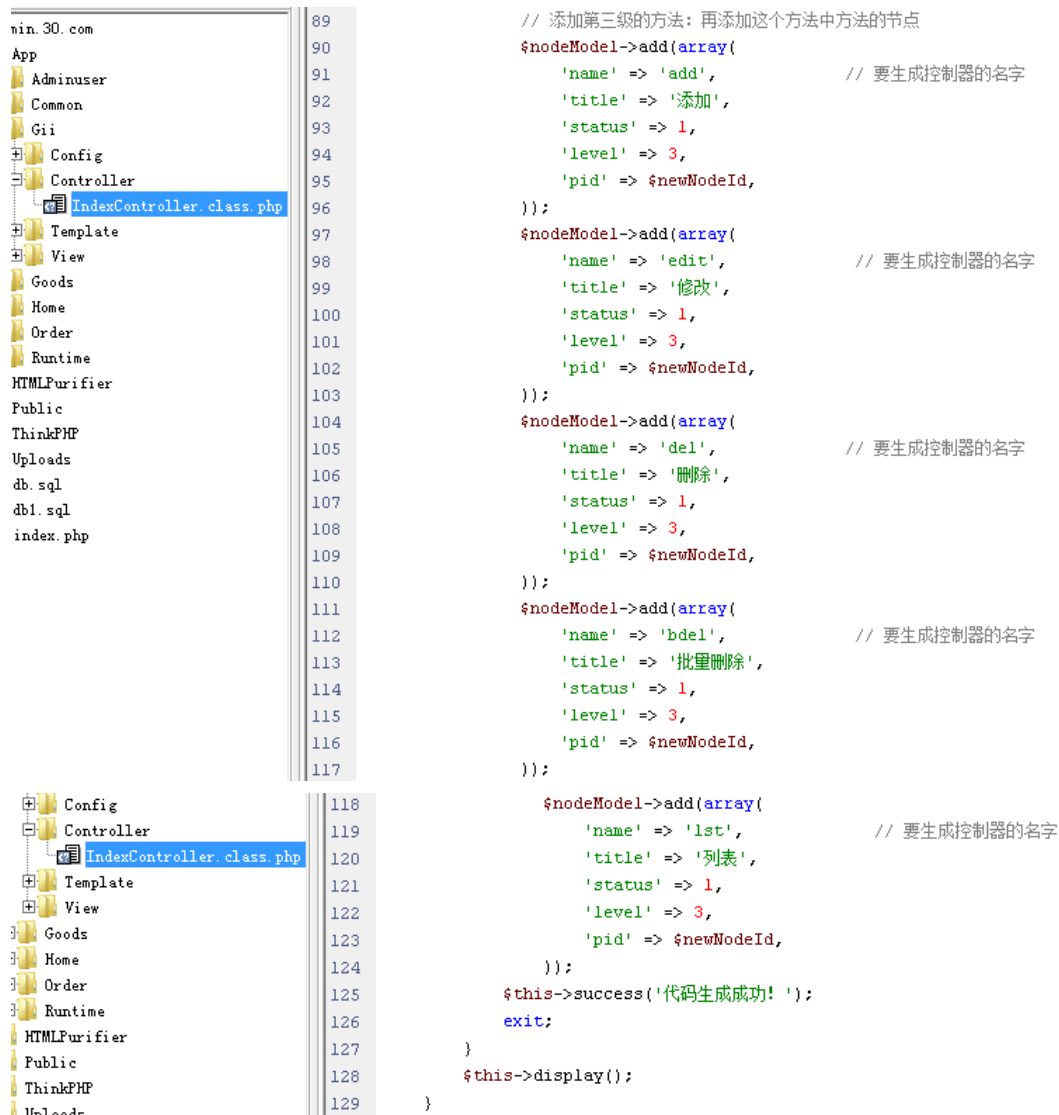
## 2. 修改代码生成器，在生成完代码之后向节点表中插入对应数据：



上面添加了顶级模块的节点，再在这个节点下添加一个二级节点（生成的控制器的节点）

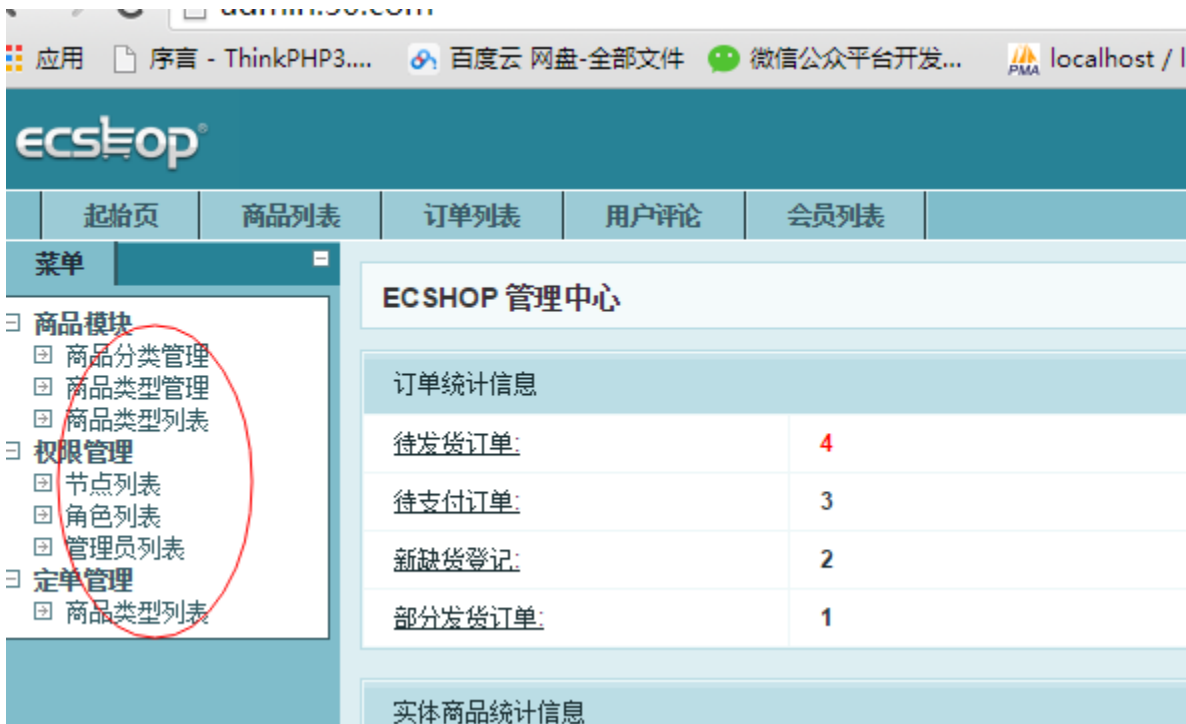


最后在上面添加的控制器节点下再添加五个方法的节点:



到此完成！现在代码生成器生成完代码之后直接刷新页面左侧就会直接显示出节点对应的按钮：





到此 RBAC 和代码生成器基本已经完成！以后可以根据这些工具做网站事务功能了。

现在做完之后，以后每做一个新的功能，每添加一个新的控制器都要把对应的节点添加到节点表这样才能分配，每次手动添加节点太麻烦，所以我们修改代码生成器，在生成代码的同时自动把生成的控制器中的方法添加节点表：

## TP 中的 RBAC 总结

使用 TP 中的 RBAC 的流程：

一、使用 TP 中自带的四张表，然后再加上自己建的一张表（管理员表）

1. 权限表：

```
CREATE TABLE IF NOT EXISTS `think_node` (
  `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(20) NOT NULL,
  `title` varchar(50) DEFAULT NULL,
  `status` tinyint(1) DEFAULT '0',
  `remark` varchar(255) DEFAULT NULL,
  `sort` smallint(6) unsigned DEFAULT NULL,
  `pid` smallint(6) unsigned NOT NULL,
  `level` tinyint(1) unsigned NOT NULL,
  PRIMARY KEY (`id`),
  KEY `level` (`level`),
  KEY `pid` (`pid`),
  KEY `status` (`status`),
  KEY `name` (`name`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

这个跟level值有关，level=1,name存的是模块的名字  
level=2,name控制器的名字  
level=3,name方法的名字

这个节点的中文名字

level只能是三级（1,2,3）

我们要自己写这张表的 CRUD 的代码：

- 商品列表
- 添加新商品
- 商品分类
- 商品品牌
- 商品回收站
- 订单管理
  - 订单列表
  - 订单查询
  - 添加订单
  - 发货单列表
  - 退货单列表
- 权限管理
  - 节点列表
  - 角色列表
  - 管理员列表

<input type="checkbox"/>	模块名/控制器名/方法名	节点标题	是否启用	操作
<input type="checkbox"/>	Goods	商品模块	启用	编辑   移除
<input type="checkbox"/>	Category	-----商品分类管理	启用	编辑   移除
<input type="checkbox"/>	list	-----列表	启用	编辑   移除
<input type="checkbox"/>	edit	-----修改	启用	编辑   移除
<input type="checkbox"/>	add	-----添加	启用	编辑   移除

删除所选

这个表中的数据是由程序员在开发完程序之后，把程序中所有的节点都添加上。

## 2. 角色表:

```

57 CREATE TABLE IF NOT EXISTS `think_role` (
58   `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,
59   `name` varchar(20) NOT NULL,
60   `pid` smallint(6) DEFAULT NULL,
61   `status` tinyint(1) unsigned DEFAULT NULL,
62   `remark` varchar(255) DEFAULT NULL,
63   PRIMARY KEY (`id`),
64   KEY `pid` (`pid`),
65   KEY `status` (`status`)
66 ) ENGINE=MyISAM DEFAULT CHARSET=utf8 ;

```

我们要写程序实现这个表的 CRUD:

- 添加订单
- 发货单列表
- 退货单列表
- 权限管理
  - 节点列表
  - 角色列表
  - 管理员列表

<input type="checkbox"/>	角色名称	是否启用	操作
<input type="checkbox"/>	经理	不启用	编辑   移除
<input type="checkbox"/>	总监	启用	编辑   移除

删除所选

在添加和修改角色时，可以为这 一个角色分配权限（这个工作是管理员做的）

- 商品管理
  - 商品列表
  - 添加新商品
  - 商品分类
  - 商品品牌
  - 商品回收站
- 订单管理
  - 订单列表
  - 订单查询
  - 添加订单
  - 发货单列表
  - 退货单列表
- 权限管理
  - 节点列表
  - 角色列表
  - 管理员列表

角色名称

请输入 \*

是否启用

☒ 启用
☐ 不启用

选择权限

☐ 商品模块
☐ -----商品分类管理
☐ -----列表
☐ -----修改
☐ -----添加

确定

重置

上面功能中的为一个角色分配角色的数据是存在下面这个表中的（一个角色都拥有哪些权限）

```

1 -----
2 CREATE TABLE IF NOT EXISTS `think_access` (
3   `role_id` smallint(6) unsigned NOT NULL,
4   `node_id` smallint(6) unsigned NOT NULL,
5   `level` tinyint(1) NOT NULL,
6   `module` varchar(50) DEFAULT NULL,
7   KEY `groupId` (`role_id`),
8   KEY `nodeId` (`node_id`)
9 ) ENGINE=MyISAM DEFAULT CHARSET=utf8;

```

3. 现在有了权限有了角色，还需要为管理员指定角色  
我们要自己建一个管理员实现管理员的 CRUD

```
107 # 管理员
108 CREATE TABLE IF NOT EXISTS `sh_user` (
109   `id` smallint(6) unsigned NOT NULL AUTO_INCREMENT,
110   `username` varchar(30) NOT NULL comment '用户名',
111   `realname` varchar(30) NOT NULL comment '真实姓名',
112   `password` char(32) not null comment '密码',
113   `status` tinyint(1) unsigned default '1' comment '是否启用: 1: 启用 0: 禁用',
114   `is_supervisor` enum("是","否") not null default '否' comment '是否超级管理员',
115   PRIMARY KEY (`id`),
116   KEY `status` (`status`)
117 ) ENGINE=MyISAM DEFAULT CHARSET=utf8 ;
118 INSERT INTO sh_user values(1,'admin','admin','16c450b7f77e73e74deef16a7f5d91f1',1,'是');
```

上面这个表只存管理员的基本信息（用户名，密码等等），至于一个管理员属于哪些角色的数据是存在 TP 提供的表中的：

```
CREATE TABLE IF NOT EXISTS `think_role_user` (
  `role_id` mediumint(9) unsigned DEFAULT NULL,
  `user_id` char(32) DEFAULT NULL,
  KEY `group_id` (`role_id`),
  KEY `user_id` (`user_id`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

\*/

#### 商品管理

- 商品列表
- 添加新商品
- 商品分类
- 商品品牌
- 商品回收站

#### 订单管理

- 订单列表
- 订单查询
- 添加订单
- 发货单列表
- 退货单列表

#### 权限管理

- 节点列表
- 角色列表
- 管理员列表

一个管理员属于的角色  
是在存储在TP提供的表中的

所在角色 ☐ 经理 ☐ 总监 \*

用户名  请输入 \*

真实姓名  请输入 \*

密码  请输入 \*

是否启用 ☒ 启用 ☐ 不启用 \*

是否超级管理员 ☒ 否 ☐ 是 \*

管理员的基本信息是存在我们建的管理员表中

确定

重置

到此五张表完成，做 RBAC 第一步先要使用上面五张表完成后台对表的操作功能。

二、修改配置文件，配置 TP 中的 RBAC:

```

14  /***** RBAC相关配置 *****/
15  'RBAC_ROLE_TABLE' => 'sh_role',
16  'RBAC_USER_TABLE' => 'sh_role_user',
17  'RBAC_ACCESS_TABLE' => 'sh_access',
18  'RBAC_NODE_TABLE' => 'sh_node',
19  'USER_AUTH_MODEL' => 'User', // 管理员模型的名字
20  'USER_AUTH_KEY' => 'user_id', // 在SESSION中用来保存管理员ID的变量名是什么
21  'USER_AUTH_TYPE' => '2', // 2: 每次检查权限时都现查数据库, <2登录时把权限存到SESSION中, 之后直接读SESSION不读数据库了,
    效果: 2: 改过权限之后, 马上生效 <2. 改过权限之后需要重新登录新的权限才会生效
22  'ADMIN_AUTH_KEY' => 'iamroot', // 超级管理员的变量名标识, TP会检查SESSION中是否有这个变量, 如果有就认为是超级管理员就拥有所有
    有的权限
23  'USER_AUTH_ON' => TRUE, // 是否启用RBAC
24  'REQUIRE_AUTH_MODULE' => '', // 哪些控制器需要验证权限
25  'NOT_AUTH_MODULE' => 'Index', // 不需要验证权限的控制器
26  'REQUIRE_AUTH_ACTION' => '', // 哪些方法需要验证
27  'NOT_AUTH_ACTION' => '', // 哪些方法不需要验证权限
28  'GUEST_AUTH_ON' => FALSE, // 是否允许匿名
29  'USER_AUTH_GATEWAY' => '/Home/Login/login', // 登录的地址是什么

```

用到的TP中的表名是什么

我们自己创建的管理员表的模型的名字

三、这些基本信息都准备好之后，就可以在程序上进行验证了：在操作任何一方法之前先验证权限：

```

3  use Think\Controller;
4  class IndexController extends Controller {
5      public function __construct()
6      {
7          parent::__construct();
8          \Org\Util\Rbac::checkLogin();
9          if(!\Org\Util\Rbac::AccessDecision())
10             $this->error('无权访问!');
11      }
12  public function top()

```

如果没有登录就跳转到配置文件中配置的登录网关，如果配置文件中定义了允许匿名登录，那么这个方法基本没有用

判断有没有权限访问当前这个操作

注意：可以使用上面两行代码验证权限，至于代码写到哪就在项目中找一个公共地方写。我们是在 Index 控制器的构造函数中写的，然后让所有的其他控制器都继承自这个控制器。

扩展：验证权限的原理？AccessDecision()方法都干什么事了？

1. 先查询数据库取出当前管理员所拥有的权限（可以访问的节点）

```

array
  'GOODS' =>
    array
      'CATEGORY' =>
        array
          'LST' => string '22' (length=2)
          'ADD' => string '23' (length=2)

```

2. 判断用户当前要访问的节点是否在这个三维数组中。

问题一、如何知道当前用户访问的是哪个节点

TP 中自带三个常量，代表当前正在访问的节点名称：

MODULE\_NAME

CONTROLLER\_NAME

ADD\_NAME

所以判断有没有权限就是看这三个常量是否在这上面的三维数组中。

```

5
7 //权限认证的过滤器方法
8 static public function AccessDecision($appName=MODULE_NAME) {
9 //检查是否需要认证
10 if(self::checkAccess()) {
11 //存在认证识别号，则进行进一步的访问决策
12 $accessGuid = md5($appName.CONTROLLER_NAME.ACTION_NAME);
13 if(empty($_SESSION[C('ADMIN_AUTH_KEY')])) {
14 if(C('USER_AUTH_TYPE')==2) {
15 //加强验证和即时验证模式 更加安全 后台权限修改可以即时生效
16 //通过数据库进行访问检查
17 $accessList = self::getAccessList($_SESSION[C('USER_AUTH_KEY')]);
18 }else {
19 //如果是管理员或者当前操作已经认证过，无需再次认证
20 if( $_SESSION[$accessGuid]) {
21 return true;
22 }
23 //登录验证模式，比较登录后保存的权限访问列表
24 $accessList = $_SESSION['_ACCESS_LIST'];
25 }
26 //判断是否为组件化模式，如果是，验证其全模块名
27 if(!isset($accessList[strtoupper($appName)][strtoupper(CONTROLLER_NAME)][strtoupper(ACTION_NAME)]) {
28 $_SESSION[$accessGuid] = false;
29 return false;
30 }
31 else {
32 $_SESSION[$accessGuid] = true;

```

四、最后我们还需要自己做一个管理员登录的功能：

登录成功之后一定要在 SESSION 中至少存两个变量，这个变量名和配置文件中相同：

```

,
/***** 登录成功 *****/
// 以下两行要和配置文件中变量名匹配，否则 TP中的RBAC无法验证权限
session('user_id', $user['id']); // 管理员的id
// 是否超级管理员
if($user['is_supervisor'] == '是')
    session('iamroot', 1);
session('username', $user['username']);
session('realname', $user['realname']);
return TRUE;
}
else

```