[PHP]\_TP5.0\_FastAdmin\_文档；

--王尔贝(我是害虫)

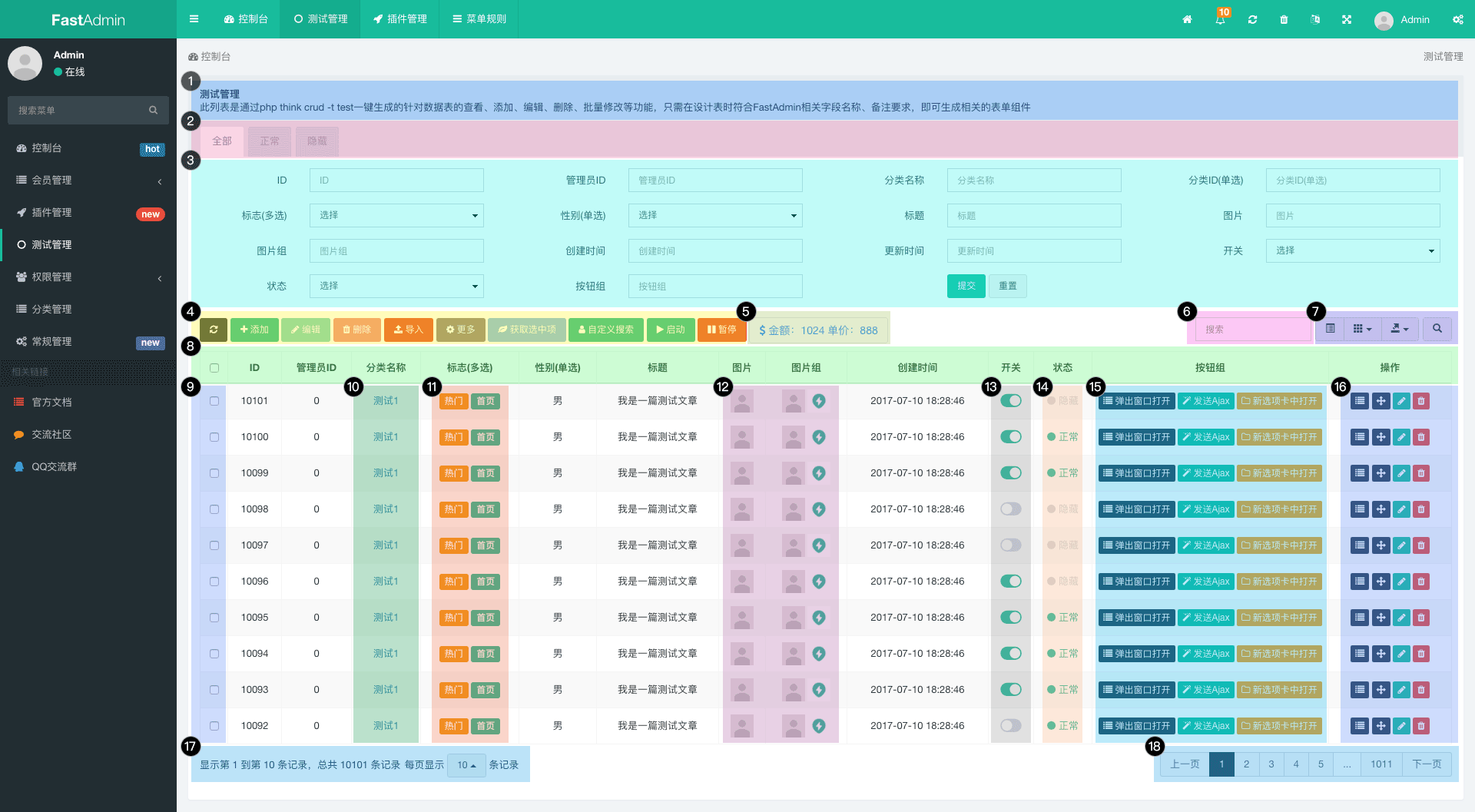
(整理 <https://doc.fastadmin.net/docs/index.html>)

--

--

[PHP]\_TP5.0\_FastAdmin\_0、图解FastAdmin表格列表的功能

(图 https://cdn.forum.fastadmin.net/uploads/201809/15/a6309f467fcade6c47d226f2890ebe29)



--

--

**一、功能描述**

(请根据图片上的数字索引查看对应功能说明)

--

**1.菜单名称和描述**

默认生成的CRUD是没有菜单名称和描述显示的，如果需要显示则可以修改权限管理->菜单规则，给对应菜单的添加上备注信息后即可显示，支持HTML。

--

**2.TAB过滤选项卡**

在一键生成CRUD时，如果表中存在status字段且为ENUM类型，则会生成相应的TAB过滤选项卡，如果需要生成其它字段的过滤选项卡则可以在使用php think crud时使用--headingfilterfield=你的字段名称来指定字段。

--

**3.通用搜索**

通用搜索的内容是根据bootstrap-table配置的字段columns决定的，渲染的内容及格式由FastAdmin自动进行渲染，如果需要禁用或删除某一选项，可以在JS中配置operate:false来删除通用搜索中的选项。例如通常情况下我们的在JS中进行字段的配置如“{field: 'createtime', title: \_\_('Create Time')},”。

**(1)、通用搜索常用配置**

这里默认是启用的通用搜索，针对通用搜索，有以下几个常用的配置。

//// 常用配置

operate: '=' //用于查询时的操作符，默认为=，为false表示禁用此字段的通用搜索，支持!=、LIKE、NOT LIKE、>、<、>=、<=、FIND\_IN\_SET、IN、NOT IN、BETWEEN、NOT BETWEEN、RANGE、NOT RANGE、NULL、NOT NULL、false；

searchList: //用于渲染列表项的数据，支持的格式有JSON Array、JSON Object、$.getJSON、Function；

addclass: //用于给input或select添加额外的class属性；

type: //用于定义input文本框的类型,默认为text；

data: //用于给input或select添加额外的属性；

**(2)、常用配置示例**

// 时间区间搜索

{field: 'createtime', title: \_\_('Createtime'), operate: 'RANGE', addclass: 'datetimerange', formatter: Table.api.formatter.datetime},

// 金额区间搜索

{field: 'money', title: \_\_('Money'), operate: 'RANGE'},

// 下拉列表搜索

{field: 'flag', title: \_\_('Flag'), searchList: {"hot": \_\_('Flag hot'), "index": \_\_('Flag index'), "recommend": \_\_('Flag recommend')}, operate: 'FIND\_IN\_SET', formatter: Table.api.formatter.label},

// 动态下拉列表搜索

{field: 'type', title: \_\_('Type'), searchList: $.getJSON("ajax/dynamicselect")},

// 禁用通用搜索

{field: 'keywords', title: \_\_('Keywords'), operate: false},

//

如果我们需要完全自定义我们的通用搜索栏，我们可以在配置bootstrap-table时定义searchFormTemplate选项来完全重写我们的通用搜索栏，具体请参考开发示例插件中的自定义搜索示例。

--

**4.工具栏按**钮

FastAdmin在一键CRUD时会自动生成添加、编辑、删除、导入、更多按钮的HTML，这些按钮会根据用户所拥有的权限来控制是否显示或隐藏。我们可以在控制器对应的index.html视图文件中任意修改或删除对应的按钮。请特别注意这几个自动生成的按钮都通过拥有的class属性来绑定相关的事件，例如添加按钮拥有btn-add这个class、框架所已经占用的class如下。

**(1)、已被占用class**

btn-add: 添加按钮使用；

btn-edit: 编辑按钮使用；

btn-del: 删除按钮使用；

btn-import: 导入按钮使用；

btn-more: 更多按钮使用；

btn-multi: 指操作使用；

btn-disabled: 添加此class后则只有在列表有选中数据时按钮才会变为可使用；

//

如果我们想点击添加按钮后默认全屏，则可以给添加按钮加上data-area='["100%","100%"]'即可默认全屏。如果我们想自定义按钮并添加事件，我们需要在视图中添加相应的HTML代码，然后在对应的JS文件中添加按钮的执行事件，切记不可在视图中直接编写JS或jQuery代码来绑定事件。

--

**5.动态渲染统计信息**

很多时候我们需要在页面额外显示服务端传回的动态数据，此时我们只需要在index.html视图中添加如下信息。

<a href="javascript:;" class="btn btn-default" style="font-size:14px; color:dodgerblue;">

<i class="fa fa-dollar"></i>

<span class="extend">

金额：<span id="money">0</span>

单价：<span id="price">0</span>

</span>

</a>

//

//// 然后在控制器对应的JS中的index方法中添加以下的JS。

// 当表格数据加载完成时

table.on('load-success.bs.table', function (e, data) {

// 这里可以获取从服务端获取的JSON数据

console.log(data);

// 这里我们手动设置底部的值

$("#money").text(data.extend.money);

$("#price").text(data.extend.price);

});

//

//// 注意务必将这段代码添加在“var table = $("#table");”之后，其中data.extend.money和data.extend.price就是我们在服务端动态返回的数据，如下。

$result = array("total" => $total, "rows" => $list, "extend" => ['money' => 1024, 'price' => 888]);

return json($result);

//

通过以上配置即可动态显示服务端返回的额外数据。

--

**6.快速搜索**

快速搜索在键入关键词时将实时从服务端搜索数据，如果你的数据表数据较大，建议关闭此功能，关闭的方法是使用search:false，其次快速搜索默认只会搜索主键id这个字段，如果你需要搜索其它字段，则需要在服务端你的控制器中定义$searchFields这个值，如“protected $searchFields = 'id,name,title';//快速搜索字段;”。这样在快速搜索时将会搜索“id,name,title”这三个字段。如果需要修改默认文本框的placeholder，可以在表格初始化前定义如下。

$.fn.bootstrapTable.locales[Table.defaults.locale]['formatSearch'] = function(){return "自定义placeholder文本";};

--

**7.浏览模式、显示隐藏列、导出、通用搜索**

**浏览模式**可以切换卡片视图和表格视图两种模式，如果不需要此功能，可以设置“showToggle: false”。

**显示隐藏列**可以快速切换字段列的显示和隐藏，如果不需要此功能，可以设置“showColumns: false”，如果想要表格中的字段列默认隐藏可以设置字段属性“visible: false”即可默认隐藏

**导出按钮**默认将导出整个表的所有行，如果需要仅导出当前分页的数据，需要设置“exportDataType: 'basic'”，如果想导出选中的行，则可以设置为“exportDataType: 'selected'”，如果不需要此功能，可以设置“showExport: false”。

**通用搜索**指表格上方的搜索，通用搜索的表单默认是隐藏的，如果需要默认显示，需要设置"searchFormVisible: true"，如果不需要通用搜索功能，可以设置"commonSearch: false"。如果想要控制字段列不参与搜索则可以设置字段列属性为"operate: false"即可。

--

**8.字段配置**

默认字段的控制是根据控制器对应的JS来配置的，因此字段配置是通过JS，而在我们的视图index.html中是没有任何字段配置的，通常我们的配置如下。

columns: [

[

{checkbox: true},

{field: 'id', title: \_\_('Id')},

{field: 'admin\_id', title: \_\_('Admin\_id')},

{field: 'category.name', title: \_\_('分类名称'), formatter:Table.api.formatter.search},

{field: 'category\_id', title: \_\_('Category\_id'), visible: false},

{field: 'flag', title: \_\_('Flag'), searchList: {"hot": \_\_('Flag hot'), "index": \_\_('Flag index'), "recommend": \_\_('Flag recommend')}, operate: 'FIND\_IN\_SET', formatter: Table.api.formatter.label},

{field: 'genderdata', title: \_\_('Genderdata'), searchList: {"male": \_\_('Genderdata male'), "female": \_\_('Genderdata female')}, formatter: Table.api.formatter.normal},

{field: 'title', title: \_\_('Title')},

{field: 'image', title: \_\_('Image'), formatter: Table.api.formatter.image},

{field: 'images', title: \_\_('Images'), formatter: Table.api.formatter.images},

{field: 'createtime', title: \_\_('Createtime'), operate: 'RANGE', addclass: 'datetimerange', formatter: Table.api.formatter.datetime},

{field: 'updatetime', title: \_\_('Updatetime'), operate: 'RANGE', addclass: 'datetimerange', formatter: Table.api.formatter.datetime, visible: false},

{field: 'weigh', title: \_\_('Weigh'), operate: false, visible: false},

{field: 'switch', title: \_\_('Switch'), searchList: {"1": \_\_('Yes'), "0": \_\_('No')}, formatter: Table.api.formatter.toggle},

{field: 'status', title: \_\_('Status'), searchList: {"normal": \_\_('Normal'), "hidden": \_\_('Hidden')}, formatter: Table.api.formatter.status},

]

]

//

**(1)、字段配置参数**

checkbox:true, //是否为首列复选框;

field:'name' //字段名称,如果启用了关联查询这里可以使用别名,比如:category.name,请注意服务端返回的字段一一对应，如果使用了一个不存在的字段，将不会渲染任何数据;

title:'名称' //字段标题,显示于头部的标题;

operate:'=' //通用搜索的操作符,详见上方通用搜索介绍;

visible:false //字段是否可见，为false时将默认不可见;

formatter:Table.api.formatter.search //格式化显示的内容,FastAdmin内部定义了许多通用的格式化方法;

events: //定义元素响应的事件;

searchList: //定义通用搜索下拉列表的数据;

addclass: //通用搜索文本框或下拉列表的额外class;

type: //通用搜索文本框的类型;

data: //通用搜索文本框或下拉列表的额外属性;

//

//// FastAdmin封装了许多常用的formatter方法，我们可以快速的调用即可。

> `Table.api.formatter.icon` 快速将字段渲染成一个按钮，仅支持Fontawesome按钮;

> `Table.api.formatter.image` 快速将字段渲染成图片展示的形式;

> `Table.api.formatter.images` 快速将字段渲染成多图片展示的形式,字段数据请以`,`进行分隔;

> `Table.api.formatter.status` 快速将字段渲染成状态，默认`normal/hidden/deleted/locked`这四个状态;

> `Table.api.formatter.url` 快速将字段渲染成URL框;

> `Table.api.formatter.search` 快速将字段渲染成可搜索的链接，点击后将执行搜索;

> `Table.api.formatter.addtabs` 快速将字段渲染成可添加到选项卡的链接，点击后将把链接添加到选项卡;

> `Table.api.formatter.flag` 快速将字段渲染成标志，仅支持`index/hot/recommend/new`这四种标志;

> `Table.api.formatter.label` 快速将字段渲染Label标签;

> `Table.api.formatter.datetime` 快速时间戳数据渲染成日期时间数据;

> `Table.api.formatter.operate` 操作栏固定按钮;

> `Table.api.formatter.buttons` 快速生成多个按钮;

> `Table.api.formatter.toggle` 快速生成切换按钮;

//

--

**9.复选框**

如果我们需要不需要复选框则移除{checkbox: true}即可。

--

**10.分类名称(关联搜索出分类表的名称)**

这里显示的分类名称是根据分类表关联查询出来的结果，如果我们启用关联查询，我们必须在当前控制器中设置属性"protected $relationSearch = true;"，同时我们的index方法也需要重写，请参考下方的完整代码中PHP部分。如果我们启用了关联查询，当两个表中的字段有冲突时，我们必须在字段中加上别名。请参考下方的完整代码中JS部分。

--

**11.标志**

我们可以使用"formatter:Table.api.formatter.flag"来渲染标志字段，默认会将数据库的值渲染以下几种颜色，{index: 'success', hot: 'warning', recommend: 'danger', 'new': 'info'}。如果我们需要扩展额外的颜色，则可以使用“custom:{aaa: 'info', bbb:'danger'}”，这样当值为aaa时会显示为蓝色，bbb时显示为红色。

--

**12.图片和图片组**

我们可以使用“formatter:Table.api.formatter.image”、“formatter:Table.api.formatter.images”。以上两种方式来渲染图片或图片组，请注意如果是图片组的情况下数据值应该是以“,”进行分隔的。

--

**13.开关**

我们可以使用“formatter:Table.api.formatter.toggle”来生成开关组件，默认情况下是根据数据库值1和0来表示开和关，我们可以通过额外的配置和定义开和关，比如“yes: 'open', no: 'close'”，则此时会根据数据库值是open还是close来展示开关，开关在点击的时候默认是只允许修改数据库的status字段的，如果我们开关不是status字段，我们需要在服务端对应的控制器中定义‘protected $multiFields="id,name,swith";’，多个字段以“,”进行分隔。

--

**14.状态渲染**

我们可以使用“formatter:Table.api.formatter.status”来渲染状态，默认根据以下值进行状态的颜色渲染，{normal: 'success', hidden: 'gray', deleted: 'danger', locked: 'info'}。如果我们状态有额外的值，我们可以使用custom来进行扩展显示的颜色，如下“custom: {rejected:'danger', agreed:'success'}”，状态渲染显示的文本是根据searchList配置的值进行渲染的。

--

**15.自定义按钮**

按钮组的功能是根据第8项中的Table.api.formatter.buttons进行生成的，代码如下。

{

field: 'buttons',

width: "120px",

title: \_\_('按钮组'),

table: table,

events: Table.api.events.operate,

buttons: [

{

name: 'detail',

text: \_\_('弹出窗口打开'),

title: \_\_('弹出窗口打开'),

classname: 'btn btn-xs btn-primary btn-dialog',

icon: 'fa fa-list',

url: 'example/bootstraptable/detail',

callback: function (data) {

Layer.alert("接收到回传数据：" + JSON.stringify(data), {title: "回传数据"});

},

visible: function (row) {

//返回true时按钮显示,返回false隐藏

return true;

}

},

{

name: 'ajax',

text: \_\_('发送Ajax'),

title: \_\_('发送Ajax'),

classname: 'btn btn-xs btn-success btn-magic btn-ajax',

icon: 'fa fa-magic',

url: 'example/bootstraptable/detail',

confirm: '确认发送',

success: function (data, ret) {

Layer.alert(ret.msg + ",返回数据：" + JSON.stringify(data));

//如果需要阻止成功提示，则必须使用return false;

//return false;

},

error: function (data, ret) {

console.log(data, ret);

Layer.alert(ret.msg);

return false;

}

},

{

name: 'addtabs',

text: \_\_('新选项卡中打开'),

title: \_\_('新选项卡中打开'),

classname: 'btn btn-xs btn-warning btn-addtabs',

icon: 'fa fa-folder-o',

url: 'example/bootstraptable/detail'

}

],

formatter: Table.api.formatter.buttons

}

//

//// 按钮配置支持的参数

name 按钮唯一标识,其中add/edit/del/dragsort已经被占用,请勿使用。我们可以在HTML视图文件的table使用data-buttons-标识来控制显示。

text 按钮的文本内容,如果不需要显示文本可忽略。

title 鼠标移上去的标题或弹窗/选项显示的标题。

icon 按钮的图标,请使用font-awesome图标库。

classname 按钮的class, 其中classname中的btn-dialog、btn-ajax、btn-addtabs、btn-click，FastAdmin已经为这几个固定的Class注册了事件，所以可以直接使用，如果想要实现其它功能，需要自己手动编写代码绑定事件才可使用。

url 按钮的链接/Ajax事件请求的URL/弹窗链接/选项卡链接，直接function和string类型,此链接会自动在链接后添加ids/{ids},{ids}为当行主键ID，如果需要传递其它字段值，请在URL中使用{字段名}占位即可。

refresh 自动刷新,只针对btn-ajax事件。

confirm 确认框提示文字,配置后会在确认操作再执行对应的事件，只针对btn-ajax/btn-dialog/btn-addtabs事件。

success 事件成功的回调,只针对btn-ajax事件。

error 事件失败的回调,只针对btn-ajax事件。

callback 弹窗回传的回调,只针对btn-dialog事件,需要在对应打开的页面中使用“Fast.api.close(data);”进行回传数据。

hidden 是否隐藏按钮,按钮默认显示,支持function和bool类型。

visible 是否显示按钮,按钮默认显示,支持function和bool类型。

click 当classname包含btn-click时的点击回调事件。

extend 按钮扩展信息,可以任意定制按钮的参数,比如我们想在新窗口中打开链接，则配置extend:' target="\_blank"'即可。

--

**16.操作**

操作区域默认是排序、编辑、删除这三个按钮，此功能也是根据第8项中Table.api.formatter.operate来实现的。排序按钮只在表中存在weigh字段时才会出现，编辑按钮和删除按钮会根据管理员所拥有的权限进行按需显示。如果我们需要重写编辑(排序、删除)按钮的相关属性，则可以使用buttons来定义编辑(排序、删除)的相关属性。其次Table.api.formatter.operate也支持buttons属性来配置多个其它按钮，如示例图中的详情按钮。请参考下方完整代码中JS部分。

--

**17.分页信息**

分页信息显示的文字可以通过在表格初始化前定义。

$.fn.bootstrapTable.locales[Table.defaults.locale]['formatSearch'] = function (pageFrom, pageTo, totalRows) {

return '显示第 ' + pageFrom + ' 到第 ' + pageTo + ' 条记录，总共 ' + totalRows + ' 条记录';

};

分页大小或分页大小选项可以在表格初始化时传入以下参数进行配置，“pageSize: 10, pageList: [10, 25, 50, 'All'],”。

--

**18.翻页信息**

翻页信息会根据服务端返回的数据行数自动进行渲染，如果返回的行数不满足分页条件时，此项是不会显示的。

--

--

**二、完整代码**

--

**1、PHP代码**

<?php

namespace app\admin\controller;

use app\common\controller\Backend;

/\*\*

\* 测试管理

\*

\* @icon fa fa-circle-o

\* @remark 此列表是通过php think crud -t test一键生成的针对数据表的查看、添加、编辑、删除、批量修改等功能，只需在设计表时符合FastAdmin相关字段名称、备注要求，即可生成相关的表单组件。

\*/

class Test extends Backend

{

// 模型

protected $model = null;

// 是否关联查询

protected $relationSearch = true;

// 初始化

public function \_initialize()

{

parent::\_initialize();

$this->model = model('Test');

}

/\*\*

\* 查看

\*/

public function index()

{

if ($this->request->isAjax())

{

list($where, $sort, $order, $offset, $limit) = $this->buildparams();

$total = $this->model

->with("category")

->where($where)

->order($sort, $order)

->count();

$list = $this->model

->with("category")

->where($where)

->order($sort, $order)

->limit($offset, $limit)

->select();

$result = array("total" => $total, "rows" => $list, "extend" => ['money' => 1024, 'price' => 888]);

return json($result);

}

return $this->view->fetch();

}

}1、PHP代码

<?php

namespace app\admin\controller;

use app\common\controller\Backend;

/\*\*

\* 测试管理

\*

\* @icon fa fa-circle-o

\* @remark 此列表是通过php think crud -t test一键生成的针对数据表的查看、添加、编辑、删除、批量修改等功能，只需在设计表时符合FastAdmin相关字段名称、备注要求，即可生成相关的表单组件。

\*/

class Test extends Backend

{

// 模型

protected $model = null;

// 是否关联查询

protected $relationSearch = true;

// 初始化

public function \_initialize()

{

parent::\_initialize();

$this->model = model('Test');

}

/\*\*

\* 查看

\*/

public function index()

{

if ($this->request->isAjax())

{

list($where, $sort, $order, $offset, $limit) = $this->buildparams();

$total = $this->model

->with("category")

->where($where)

->order($sort, $order)

->count();

$list = $this->model

->with("category")

->where($where)

->order($sort, $order)

->limit($offset, $limit)

->select();

$result = array("total" => $total, "rows" => $list, "extend" => ['money' => 1024, 'price' => 888]);

return json($result);

}

return $this->view->fetch();

}

}

--

**2、JS代码**

define(['jquery', 'bootstrap', 'backend', 'table', 'form'], function ($, undefined, Backend, Table, Form) {

var Controller = {

index: function () {

// 初始化表格参数配置

Table.api.init({

extend: {

index\_url: 'test/index',

add\_url: 'test/add',

edit\_url: 'test/edit',

del\_url: 'test/del',

multi\_url: 'test/multi',

table: 'test',

}

});

var table = $("#table");

//当表格数据加载完成时

table.on('load-success.bs.table', function (e, data) {

//这里可以获取从服务端获取的JSON数据

console.log(data);

//这里我们手动设置底部的值

$("#money").text(data.extend.money);

$("#price").text(data.extend.price);

});

// 初始化表格

table.bootstrapTable({

url: $.fn.bootstrapTable.defaults.extend.index\_url,

pk: 'id',

sortName: 'weigh',

columns: [

[

{checkbox: true},

{field: 'id', title: \_\_('Id')},

{field: 'admin\_id', title: \_\_('Admin\_id')},

{field: 'category.name', title: \_\_('分类名称'), formatter:Table.api.formatter.search},

{field: 'category\_id', title: \_\_('Category\_id'), visible: false},

{field: 'flag', title: \_\_('Flag'), searchList: {"hot": \_\_('Flag hot'), "index": \_\_('Flag index'), "recommend": \_\_('Flag recommend')}, operate: 'FIND\_IN\_SET', formatter: Table.api.formatter.label},

{field: 'genderdata', title: \_\_('Genderdata'), searchList: {"male": \_\_('Genderdata male'), "female": \_\_('Genderdata female')}, formatter: Table.api.formatter.normal},

{field: 'title', title: \_\_('Title')},

{field: 'image', title: \_\_('Image'), formatter: Table.api.formatter.image},

{field: 'images', title: \_\_('Images'), formatter: Table.api.formatter.images},

{field: 'createtime', title: \_\_('Createtime'), operate: 'RANGE', addclass: 'datetimerange', formatter: Table.api.formatter.datetime},

{field: 'updatetime', title: \_\_('Updatetime'), operate: 'RANGE', addclass: 'datetimerange', formatter: Table.api.formatter.datetime, visible: false},

{field: 'weigh', title: \_\_('Weigh'), operate: false, visible: false},

{field: 'switch', title: \_\_('Switch'), searchList: {"1": \_\_('Yes'), "0": \_\_('No')}, formatter: Table.api.formatter.toggle},

{field: 'status', title: \_\_('Status'), searchList: {"normal": \_\_('Normal'), "hidden": \_\_('Hidden')}, formatter: Table.api.formatter.status},

{

field: 'buttons',

width: "120px",

title: \_\_('按钮组'),

table: table,

events: Table.api.events.operate,

buttons: [

{

name: 'detail',

text: \_\_('弹出窗口打开'),

title: \_\_('弹出窗口打开'),

classname: 'btn btn-xs btn-primary btn-dialog',

icon: 'fa fa-list',

url: 'example/bootstraptable/detail',

callback: function (data) {

Layer.alert("接收到回传数据：" + JSON.stringify(data), {title: "回传数据"});

},

visible: function (row) {

//返回true时按钮显示,返回false隐藏

return true;

}

},

{

name: 'ajax',

text: \_\_('发送Ajax'),

title: \_\_('发送Ajax'),

classname: 'btn btn-xs btn-success btn-magic btn-ajax',

icon: 'fa fa-magic',

url: 'example/bootstraptable/detail',

confirm: '确认发送',

success: function (data, ret) {

Layer.alert(ret.msg + ",返回数据：" + JSON.stringify(data));

//如果需要阻止成功提示，则必须使用return false;

//return false;

},

error: function (data, ret) {

console.log(data, ret);

Layer.alert(ret.msg);

return false;

}

},

{

name: 'addtabs',

text: \_\_('新选项卡中打开'),

title: \_\_('新选项卡中打开'),

classname: 'btn btn-xs btn-warning btn-addtabs',

icon: 'fa fa-folder-o',

url: 'example/bootstraptable/detail'

}

],

formatter: Table.api.formatter.buttons

},

{

field: 'operate', title: \_\_('Operate'), table: table, events: Table.api.events.operate,

buttons: [

{

name: 'detail',

title: \_\_('详情'),

classname: 'btn btn-xs btn-primary btn-dialog',

icon: 'fa fa-list',

url: 'test/detail',

callback: function (data) {

Layer.alert("接收到回传数据：" + JSON.stringify(data), {title: "回传数据"});

}

}],

formatter: Table.api.formatter.operate

}

]

]

});

// 绑定TAB事件(刷新表格数据)

$('.panel-heading a[data-toggle="tab"]').on('shown.bs.tab', function (e) {

var field = $(this).closest("ul").data("field");

var value = $(this).data("value");

var options = table.bootstrapTable('getOptions');

options.pageNumber = 1;

options.queryParams = function (params) {

var filter = {};

if (value !== '') {

filter[field] = value;

}

params.filter = JSON.stringify(filter);

return params;

};

table.bootstrapTable('refresh', {});

return false;

});

// 为表格绑定事件

Table.api.bindevent(table);

},

add: function () {

Controller.api.bindevent();

},

edit: function () {

Controller.api.bindevent();

},

api: {

bindevent: function () {

Form.api.bindevent($("form[role=form]"));

}

}

};

return Controller;

});

--

3、HTML代码

<div class="panel panel-default panel-intro">

<!-- tab页 -->

<div class="panel-heading">

{:build\_heading(null,FALSE)}

<ul class="nav nav-tabs" data-field="status">

<li class="active"><a href="#t-all" data-value="" data-toggle="tab">{:\_\_('All')}</a></li>

{foreach name="statusList" item="vo"}

<li><a href="#t-{$key}" data-value="{$key}" data-toggle="tab">{$vo}</a></li>

{/foreach}

</ul>

</div>

<div class="panel-body">

<div id="myTabContent" class="tab-content">

<div class="tab-pane fade active in" id="one">

<div class="widget-body no-padding">

<div id="toolbar" class="toolbar">

<a href="javascript:;" class="btn btn-primary btn-refresh" title="{:\_\_('Refresh')}" ><i class="fa fa-refresh"></i> </a>

<a href="javascript:;" class="btn btn-success btn-add {:$auth->check('test/add')?'':'hide'}" title="{:\_\_('Add')}" ><i class="fa fa-plus"></i> {:\_\_('Add')}</a>

<a href="javascript:;" class="btn btn-success btn-edit btn-disabled disabled {:$auth->check('test/edit')?'':'hide'}" title="{:\_\_('Edit')}" ><i class="fa fa-pencil"></i> {:\_\_('Edit')}</a>

<a href="javascript:;" class="btn btn-danger btn-del btn-disabled disabled {:$auth->check('test/del')?'':'hide'}" title="{:\_\_('Delete')}" ><i class="fa fa-trash"></i> {:\_\_('Delete')}</a>

<a href="javascript:;" class="btn btn-danger btn-import {:$auth->check('test/import')?'':'hide'}" title="{:\_\_('Import')}" id="btn-import-file" data-url="ajax/upload" data-mimetype="csv,xls,xlsx" data-multiple="false"><i class="fa fa-upload"></i> {:\_\_('Import')}</a>

<div class="dropdown btn-group {:$auth->check('test/multi')?'':'hide'}">

<a class="btn btn-primary btn-more dropdown-toggle btn-disabled disabled" data-toggle="dropdown"><i class="fa fa-cog"></i> {:\_\_('More')}</a>

<ul class="dropdown-menu text-left" role="menu">

<li><a class="btn btn-link btn-multi btn-disabled disabled" href="javascript:;" data-params="status=normal"><i class="fa fa-eye"></i> {:\_\_('Set to normal')}</a></li>

<li><a class="btn btn-link btn-multi btn-disabled disabled" href="javascript:;" data-params="status=hidden"><i class="fa fa-eye-slash"></i> {:\_\_('Set to hidden')}</a></li>

</ul>

</div>

<a class="btn btn-info btn-disabled disabled btn-selected" href="javascript:;"><i class="fa fa-leaf"></i> 获取选中项</a>

<a class="btn btn-success btn-singlesearch" href="javascript:;"><i class="fa fa-user"></i> 自定义搜索</a>

<a class="btn btn-success btn-change btn-start" data-params="action=start" data-url="example/bootstraptable/start" href="javascript:;"><i class="fa fa-play"></i> 启动</a>

<a class="btn btn-danger btn-change btn-pause" data-params="action=pause" data-url="example/bootstraptable/pause" href="javascript:;"><i class="fa fa-pause"></i> 暂停</a>

<a href="javascript:;" class="btn btn-default" style="font-size:14px;color:dodgerblue;">

<i class="fa fa-dollar"></i>

<span class="extend">

金额：<span id="money">0</span>

单价：<span id="price">0</span>

</span>

</a>

</div>

<table id="table" class="table table-striped table-bordered table-hover table-nowrap"

data-operate-edit="{:$auth->check('test/edit')}"

data-operate-del="{:$auth->check('test/del')}"

width="100%">

</table>

</div>

</div>

</div>

</div>

</div>

整理( <https://ask.fastadmin.net/article/323.html> )

--

--

[PHP]\_TP5.0\_FastAdmin\_1、介绍

(整理 <https://doc.fastadmin.net/docs/index.html>)

--

FastAdmin是一款基于ThinkPHP5+Bootstrap的极速后台开发框架。

--

**一、主要特性**

1、基于Auth验证的权限管理系统。

(1)、支持无限级父子级权限继承，父级的管理员可任意增删改子级管理员及权限设置。

(2)、支持单管理员多角色。

(3)、支持管理子级数据或个人数据。

2、强大的一键生成功能。

(1)、一键生成CRUD,包括控制器、模型、视图、JS、语言包、菜单、回收站等。

(2)、一键压缩打包JS和CSS文件，一键CDN静态资源部署。

(3)、一键生成控制器菜单和规则。

(4)、一键生成API接口文档。

3、完善的前端功能组件开发。

(1)、基于AdminLTE二次开发。

(2)、基于Bootstrap开发，自适应手机、平板、PC。

(3)、基于RequireJS进行JS模块管理，按需加载。

(4)、基于Less进行样式开发。

(5)、基于Bower进行前端组件包管理。

4、强大的插件扩展功能，在线安装卸载升级插件。

5、通用的会员模块和API模块。

6、共用同一账号体系的Web端会员中心权限验证和API接口会员权限验证。

7、二级域名部署支持，同时域名支持绑定到插件。

8、多语言支持，服务端及客户端支持。

9、强大的第三方模块支持(CMS、博客、知识付费问答、投票)。

10、整合第三方短信接口(阿里云、腾讯云短信)。

11、无缝整合第三方云存储(七牛、阿里云OSS、又拍云)功能。

12、第三方富文本编辑器支持(Summernote、Kindeditor、百度编辑器)。

13、第三方登录(QQ、微信、微博)整合。

14、第三方支付(微信、支付宝)无缝整合，微信支持PC端扫码支付。

15、丰富的插件应用市场。

--

**二、安装使用**

https://doc.fastadmin.net

--

**三、在线演示**

https://demo.fastadmin.net

用户名：admin；

密　码：123456；

提　示：演示站数据无法进行修改，请下载源码安装体验全部功能；

--

**四、界面截图**

--

**五、问题反馈**

在使用中有任何问题，请使用以下联系方式联系我们。

交流社区: https://forum.fastadmin.net

QQ群: 636393962(满) 708784003(满) 964776039(3群)

Email: (karsonzhang#163.com, 把#换成@)

Github: https://github.com/karsonzhang/fastadmin

Gitee: https://gitee.com/karson/fastadmin

--

**六、特别鸣谢**

(感谢以下的项目,排名不分先后)

ThinkPHP：http://www.thinkphp.cn

AdminLTE：https://adminlte.io

Bootstrap：http://getbootstrap.com

jQuery：http://jquery.com

Bootstrap-table：https://github.com/wenzhixin/bootstrap-table

Nice-validator: https://validator.niceue.com

SelectPage: https://github.com/TerryZ/SelectPage

--

**七、版权信息**

FastAdmin遵循Apache2开源协议发布，并提供免费使用。本项目包含的第三方源码和二进制文件之版权信息另行标注。版权所有Copyright © 2017-2019 by FastAdmin (<https://www.fastadmin.net>) All rights reserved。

--

--

[PHP]\_TP5.0\_FastAdmin\_2、安装

(整理 <https://doc.fastadmin.net/docs/install.html>)

--

**一、环境要求**

FastAdmin推荐你使用阿里云(<https://www.fastadmin.net/go/aliyun>)和腾讯云(<https://www.fastadmin.net/go/tencent>)服务器。

PHP >= 5.5.0 (推荐PHP7.1版本)

Mysql >= 5.5.0 (需支持innodb引擎)

Apache 或 Nginx

PDO PHP Extension

MBstring PHP Extension

CURL PHP Extension

Node.js (可选,用于安装Bower和LESS,同时打包压缩也需要使用到)

Composer (可选,用于管理第三方扩展包)

Bower (可选,用于管理前端资源)

Less (可选,用于编辑less文件,如果你需要增改css样式,最好安装上)

--

**二、完整包安装**

前往官网下载页面(<https://www.fastadmin.net/download.html>)下载完整包解压到你的项目目录。

添加虚拟主机并绑定到项目中的public目录。

访问 <http://www.yoursite.com/install.php> 进行安装。

--

**三、命令行安装**

强烈建议使用命令行安装，因为采用命令行安装的方式可以和FastAdmin随时保持更新同步。使用命令行安装请提前准备好Git、Node.js、Composer、Bower环境，我们为Windows下开发者准备了一个简单的视频安装教程( <https://www.fastadmin.net/video/install.html> )，可跟着教程一步一步安装。Linux下FastAdmin的安装请使用以下命令进行安装。

// 克隆FastAdmin到你本地

git clone https://gitee.com/karson/fastadmin.git

// 进入目录

cd fastadmin

// 下载前端插件依赖包

bower install

// 下载PHP依赖包

composer install

// 一键创建数据库并导入数据

php think install -u 数据库用户名 -p 数据库密码

// 添加虚拟主机并绑定到fastadmin/public目录

--

**四、常见问题**

1、如果使用命令行安装则后台管理默认账号是admin，默认密码是123456。

2、提示请先下载完整包覆盖后再安装，说明你是直接从仓库下载的代码，请从官网下载完整包覆盖后再进行安装。

3、执行php think install时出现Access denied for user ...，请确保数据库服务器、用户名、密码配置正确。

4、执行php think install时报不是内部或外部命令? 请将php.exe所在的目录路径加入到环境变量PATH中。

5、如果提示当前权限不足，无法写入配置文件application/database.php，请检查database.php是否可读，还有可能是当前安装程序无法访问父目录，请检查PHP的open\_basedir配置。

6、如果提示找不到fastadmin.fa\_admin表或表不存在，请检查你的MySQL是否开启了支持innodb。

6、如果在Linux环境中使用的是root账户，bower install执行出错，请尝试添加上--allow-root参数。

7、如果访问后台右侧空白，请检查资源是否下载完整，可使用bower install多试两次或下载资源包覆盖。

8、如果composer install失败，请尝试在命令行进行切换配置到国内源，命令如下composer config -g repo.packagist composer <https://packagist.laravel-china.org>。

遇到问题到社区或QQ群(636393962)反馈。

--

--

[PHP]\_TP5.0\_FastAdmin\_3、命令行

(整理 <https://doc.fastadmin.net/docs/command.html>)

--

FastAdmin基于ThinkPHP5强大的命令行功能扩展了一系列命令行功能，可以很方便的一键生成CRUD、生成权限菜单、压缩打包CSS和JS、安装配置插件等功能。

--

**一、一键生成CRUD**

在FastAdmin中可以快速的一键生成CRUD，其中包括控制器、模型、视图、验证器、语言包、JS。

//

**1、准备工作**

在数据库中创建一个fa\_test数据表，编辑好表字段结构，并且一定写上字段注释和表注释，相关数据表字段的设计要求可以参考数据库章节。FastAdmin在生成CRUD时会根据字段属性、字段注释、表注释自动生成语言包、组件和排版。请确保php所在的目录已经加入到系统环境变量，否则会提示找不到该命令。打开命令行控制台进入到FastAdmin根目录，也就是think文件所在的目录。

//

**2、常用命令**

// 生成fa\_test表的CRUD

php think crud -t test

// 生成fa\_test表的CRUD且一键生成菜单

php think crud -t test -u 1

// 删除fa\_test表生成的CRUD

php think crud -t test -d 1

// 生成fa\_test表的CRUD且控制器生成在二级目录下

php think crud -t test -c mydir/test

// 生成fa\_test\_log表的CRUD且生成对应的控制器为testlog

php think crud -t test\_log -c testlog

// 生成fa\_test表的CRUD且对应的模型名为testmodel

php think crud -t test -m testmodel

// 生成fa\_test表的CRUD且生成关联模型category，外链为category\_id，关联表主键为id

php think crud -t test -r category -k category\_id -p id

// 生成fa\_test表的CRUD且所有以list或data结尾的字段都生成复选框

php think crud -t test --setcheckboxsuffix=list --setcheckboxsuffix=data

// 生成fa\_test表的CRUD且所有以image和img结尾的字段都生成图片上传组件

php think crud -t test --imagefield=image --imagefield=img

// 关联多个表,参数传递时请按顺序依次传递，支持以下几个参数relation/relationmodel/relationforeignkey/relationprimarykey/relationfields/relationmode

php think crud -t test --relation=category --relation=admin --relationforeignkey=category\_id --relationforeignkey=admin\_id

// 生成v\_phealth\_db2数据库下的fa\_test表的CRUD

php think crud -t test --db=v\_phealth\_db2

//

**3、参数介绍**

-t, --table=TABLE 表名，带不带表前缀均可

-c, --controller[=CONTROLLER] 生成的控制器名,可选,默认根据表名进行自动解析

-m, --model[=MODEL] 生成的模型名,可选,默认根据表名进行自动解析

-i, --fields[=FIELDS] 生成的数据列表中可见的字段，默认是全部

-f, --force[=FORCE] 是否覆盖模式,如果目标位置已经有对应的控制器或模型会提示

-l, --local[=LOCAL] 是否本地模型,默认1,置为0时,模型将生成在common模块下

-r, --relation[=RELATION] 关联模型表名，带不带表前缀均可

-e, --relationmodel[=RELATIONMODEL] 生成的关联模型名,可选,默认根据表名进行自动解析

-k, --relationforeignkey[=RELATIONFOREIGNKEY] 表外键,可选,默认会识别为使用 模型\_id 名称

-p, --relationprimarykey[=RELATIONPRIMARYKEY] 关联模型表主键,可选,默认会自动识别

-s, --relationfields[=RELATIONFIELDS] 关联模型表显示的字段，默认是全部

-o, --relationmode[=RELATIONMODE] 关联模型,hasone或belongsto [default: "belongsto"]

-d, --delete[=DELETE] 删除模式,将删除之前使用CRUD命令生成的相关文件

-u, --menu[=MENU] 菜单模式,生成CRUD后将继续一键生成菜单

--db[=key] 多数据库支持(参数为tp5中配置的数据库key 在application\config.php添加数据库配置信息)

--setcheckboxsuffix[=SETCHECKBOXSUFFIX] 自动生成复选框的字段后缀

--enumradiosuffix[=ENUMRADIOSUFFIX] 自动生成单选框的字段后缀

--imagefield[=IMAGEFIELD] 自动生成图片上传组件的字段后缀

--filefield[=FILEFIELD] 自动生成文件上传组件的字段后缀

--intdatesuffix[=INTDATESUFFIX] 自动生成日期组件的字段后缀

--switchsuffix[=SWITCHSUFFIX] 自动生成可选组件的字段后缀

--citysuffix[=CITYSUFFIX] 自动生成城市选择组件的字段后缀

--selectpagesuffix[=SELECTPAGESUFFIX] 自动生成Selectpage组件的字段后缀

--ignorefields[=IGNOREFIELDS] 排除的字段

--editorclass[=EDITORCLASS] 自动生成富文本组件的字段后缀

--headingfilterfield[=HEADINGFILTERFIELD] 自动生成筛选过滤选项卡的字段，默认是status字段

--sortfield[=SORTFIELD] 排序字段

//

**4、常见问题**

(1)、如果你的表带有下划级会自动生成带层级的控制器和视图，如果你不希望生成带层级的控制器和视图，请使用-c 参数，例如：php think crud -t test\_log -c testlog将会生成testlog这个控制器，同理如果你的普通表想生成带层级的控制器则可以使用php think crud -t test -c mydir/test。

(2)、FastAdmin自带一个fa\_test表用于测试CRUD能支持的字段名称和类型，请直接使用php think crud -t test生成查看。

(3)、生成CRUD后，关联表外键在列表未显示对应的关联表数据信息，此时建议你使用在线命令行插件进行可视化生成。

(4)、生成CRUD后，在添加或编辑时外键字段未能正确显示关联表数据列表，请查看数据库章节常见问题中的说明。

(5)、如果需要生成回收站，请务必保证你的表中存在deletetime字段，且默认值为null，这样在一键CRUD时将自动生成回收站的功能，如果deletetime默认值为0，数据将默认进入回收站。

//

**5、使用范例**

更多CRUD一键生成可使用的参数请使用php think crud --help查看。

--

**二、一键生成菜单**

FastAdmin可通过命令控制台快速的一键生成后台的权限节点，同时后台的管理菜单也会同步改变，操作非常简单。

//

**1、准备工作**

首先确保已经将FastAdmin配置好，数据库连接正确，同时确保已经通过上一步的一键生成CRUD已经生成了test的CRUD。请确保php所在的目录已经加入到系统环境变量，否则会提示找不到该命令。打开控制台进入到FastAdmin根目录，也就是think文件所在的目录。

//

**2、常用命令**

// 一键生成test控制器的权限菜单

php think menu -c test

// 一键生成mydir/test控制器的权限菜单

php think menu -c mydir/test

// 删除test控制器生成的菜单

php think menu -c test -d 1

// 一键全部重新所有控制器的权限菜单

php think menu -c all-controller

//

**3、常见问题**

(1)、在使用php think menu前确保你的控制器已经添加或通过php think crud生成好。

(2)、如果之前已经生成了菜单,需要再次生成,请登录后台手动删除之前生成的菜单或使用php think menu -c 控制器名 -d 1来删除。

(3)、如果生成层级目录的菜单，在后台展示时父级菜单会以目录名称显示，如果需要修改可以在application/admin/lang/zh-cn.php中追加相应的语言包即可。

//

**4、使用范例**

更多CRUD一键生成可使用的参数请使用php think menu --help查看。

--

**三、一键压缩打包**

在FastAdmin中如果修改了核心的JS或CSS文件，是需要重新压缩打包后在生产环境下才会生效。FastAdmin采用的是基于RequireJS的r.js进行JS和CSS文件的压缩打包。

//

**1、准备工作**

请先确保你的环境已经安装好Node环境。首先确认你application/config.php中app\_debug的值，当为true的时候是采用的无压缩的JS和CSS，当为false时采用的是压缩版的JS和CSS。请确保php所在的目录已经加入到系统环境变量，否则会提示找不到该命令。打开命令行控制台进入到FastAdmin根目录，也就是think文件所在的目录。

//

**2、常用命令**

// 一键压缩打包前后台的JS和CSS

php think min -m all -r all

// 一键压缩打包后台的JS和CSS

php think min -m backend -r all

// 一键压缩打包前后台的JS

php think min -m all -r js

// 一键压缩打包后台的CSS

php think min -m backend -r css

//

**3、常见问题**

Windows系统需要手动配置node的路径,请参考在Windows下如何压缩打包JS和CSS。如果无法进行打包，可以使用php think min -m all -r all -vvv尝试下，看下错误信息。如果压缩打包后访问不生效，请检查是否是你的浏览器缓存的原因。请不要直接修改以.min.js和.min.css结尾的文件，因为一键压缩打包后会进行覆盖。安装或卸载插件后无需进行压缩打包JS和CSS。

//

**4、影响文件**

在调试模式和生产环境下所加载的JS和CSS是不一样的，压缩打包会重新生成相应环境下的JS和CSS文件，特别注意下所述。

// 调试模式

public/assets/js/require-frontend.js

public/assets/js/require-backend.js

public/assets/css/frontend.css

public/assets/css/backend.css

// 生产环境(打包压缩后会重新生成以下文件)

public/assets/js/require-frontend.min.js

public/assets/js/require-backend.min.js

public/assets/css/frontend.min.css

public/assets/css/backend.min.css

//

**5、使用范例**

JS和CSS文件压缩前和压缩后浏览器请求对比(请右键查看大图)。更多一键生成JS和CSS的参数请使用php think min --help查看。

--

**四、一键生成API文档**

FastAdmin中的一键生成API文档可以在命令行或后台一键生成我们API接口的接口测试文档，可以直接在线模拟接口请求，查看参数示例和返回示例。

//

**1、准备工作**

请确保你的API模块下的控制器代码没有语法错误，控制器类注释、方法名注释完整，注释规则请参考下方注释规则。请确保你的FastAdmin已经安装成功且能正常登录后台。请确保php所在的目录已经加入到系统环境变量，否则会提示找不到该命令。打开命令行控制台进入到FastAdmin根目录，也就是think文件所在的目录。

//

**2、常用命令**

// 一键生成API文档

php think api --force=true

// 指定<https://www.example.com>为API接口请求域名,默认为空

php think api -u <https://www.example.com> --force=true

// 输出自定义文件为myapi.html,默认为api.html

php think api -o myapi.html --force=true

// 修改API模板为mytemplate.html，默认为index.html

php think api -e mytemplate.html --force=true

// 修改标题为FastAdmin,作者为作者

php think api -t FastAdmin -a Karson --force=true

// 查看API接口命令行帮助

php think api -h

//

**3、参数介绍**

-u, --url[=URL] 默认API请求URL地址 [default: ""]

-m, --module[=MODULE] 模块名(admin/index/api) [default: "api"]

-o, --output[=OUTPUT] 输出文件 [default: "api.html"]

-e, --template[=TEMPLATE] 模板文件 [default: "index.html"]

-f, --force[=FORCE] 覆盖模式 [default: false]

-t, --title[=TITLE] 文档标题 [default: "FastAdmin"]

-a, --author[=AUTHOR] 文档作者 [default: "FastAdmin"]

-c, --class[=CLASS] 扩展类 (multiple values allowed)

-l, --language[=LANGUAGE] 语言 [default: "zh-cn"]

//

**4、注释规则**

在我们的控制器中通常分为两部分注释，一是控制器头部的注释，二是控制器方法的注释。

(1)、控制器注释(名称 描述 示例)

@ApiSector API分组名称 (测试分组)

@ApiRoute API接口URL，此@ApiRoute只是基础URL (/api/test)

@ApiInternal 忽略的控制器,表示此控制将不加入API文档 无

@ApiWeigh API方法的排序,值越大越靠前 (99)

(2)、控制器方法注释(名称 描述 示例)

@ApiTitle API接口的标题,为空时将自动匹配注释的文本信息 (测试标题)

@ApiSummary API接口描述 (测试描述)

@ApiRoute API接口地址,为空时将自动计算请求地址 (/api/test/index)

@ApiMethod API接口请求方法,默认为GET (POST)

@ApiSector API分组,默认按钮控制器或控制器的@ApiSector进行分组 (测试分组)

@ApiParams API请求参数,如果在@ApiRoute中有对应的{@参数名}，将进行替换 (name="id", type="integer", required=true, description="会员ID")

@ApiHeaders API请求传递的Headers信息 (name=token, type=string, required=true, description="请求的Token")

@ApiReturn API返回的结果示例 ({"code":1,"msg":"返回成功"})

@ApiReturnParams API返回的结果参数介绍 (name="code", type="integer", required=true, sample="0")

@ApiReturnHeaders API返回的Headers信息 (name="token", type="integer", required=true, sample="123456")

@ApiInternal 忽略的方法,表示此方法将不加入文档 无

@ApiWeigh API方法的排序,值越大越靠前 (99)

//

**5、标准范例**

<?php

namespace app\api\controller;

/\*\*

\* 测试API控制器

\*/

class Test extends \app\common\controller\Api

{

// 无需验证登录的方法

protected $noNeedLogin = ['test'];

// 无需要判断权限规则的方法

protected $noNeedRight = ['\*'];

/\*\*

\* 首页

\*

\* 可以通过@ApiInternal忽略请求的方法

\* @ApiInternal

\*/

public function index()

{

return 'index';

}

/\*\*

\* 私有方法

\* 私有的方法将不会出现在文档列表

\*/

private function privatetest()

{

return 'private';

}

/\*\*

\* 测试方法

\*

\* @ApiTitle (测试名称)

\* @ApiSummary (测试描述信息)

\* @ApiSector (测试分组)

\* @ApiMethod (POST)

\* @ApiRoute (/api/test/test/id/{id}/name/{name})

\* @ApiHeaders (name=token, type=string, required=true, description="请求的Token")

\* @ApiParams (name="id", type="integer", required=true, description="会员ID")

\* @ApiParams (name="name", type="string", required=true, description="用户名")

\* @ApiParams (name="data", type="object", sample="{'user\_id':'int','user\_name':'string','profile':{'email':'string','age':'integer'}}", description="扩展数据")

\* @ApiReturnParams (name="code", type="integer", required=true, sample="0")

\* @ApiReturnParams (name="msg", type="string", required=true, sample="返回成功")

\* @ApiReturnParams (name="data", type="object", sample="{'user\_id':'int','user\_name':'string','profile':{'email':'string','age':'integer'}}", description="扩展数据返回")

\* @ApiReturn ({

'code':'1',

'mesg':'返回成功'

\* })

\*/

public function test($id = '', $name = '')

{

$this->success("返回成功", $this->request->request());

}

}

//

**6、常见问题**

如果控制器的方法是private或protected的，则将不会生成相应的API文档。如果注释不生效，请检查注释文本是否正确。

--

**五、一键管理插件**

FastAdmin中的插件可以通过命令行快速的进行安装、卸载、禁用和启用。

//

**1、准备工作**

请确保你的FastAdmin已经能正常登录后台。请确保php所在的目录已经加入到系统环境变量，否则会提示找不到该命令。打开命令行控制台进入到FastAdmin根目录，也就是think文件所在的目录。

//

**2、常用命令**

// 创建一个myaddon本地插件，常用于开发自己的插件时使用

php think addon -a myaddon -c create

// 刷新插件缓存，如果禁用启用了插件，部分文件需要刷新才会生效

php think addon -a example -c refresh

// 远程安装example插件

php think addon -a example -c install

// 卸载本地的example插件

php think addon -a example -c uninstall

// 启用本地的example插件

php think addon -a example -c enable

// 禁用本地的example插件

php think addon -a example -c disable

// 升级本地的example插件

php think addon -a example -c upgrade

// 将本地的example插件打包成zip文件

php think addon -a example -c package

//

**3、常见问题**

付费插件请直接在后台插件管理或FastAdmin官方插件市场下载，然后进行离线安装。更多一键管理插件的参数请使用php think addon --help查看。

--

**六、一键安装FastAdmin**

FastAdmin可以在命令行使用命令快速的一键安装或重新安装。

//

**1、准备工作**

请确保你的数据库存储引擎支持innodb引擎，如果不支持将无法正常安装FastAdmin。请确保php所在的目录已经加入到系统环境变量，否则会提示找不到该命令。打开命令行控制台进入到FastAdmin根目录，也就是think文件所在的目录。

//

**2、常用命令**

// 一键安装FastAdmin

php think install

// 配置数据库连接地址为127.0.0.1

php think install -a 127.0.0.1

// 配置数据库用户名密码

php think install -u root -p 123456

// 配置数据库表名为dbname

php think install -d dbname

// 配置数据库表前缀为ff\_

php think install -r ff\_

// 强制重新安装FastAdmin

php think install -f 1

//

**3、常见问题**

更多一键安装FastAdmin的参数请使用php think install --help查看。

--

--

[PHP]\_TP5.0\_FastAdmin\_4、插件

(整理 <https://doc.fastadmin.net/docs/addons.html>)

--

在FastAdmin中可自由扩展和开发插件，这里简单整理一个插件文档。在此以官方博客插件为示例来介绍，因为官方博客插件是前后台都有的一套完整方案。

--

**一、目录结构**

blog

├── application

│ └── admin

│ ├── controller

│ │ └── blog

│ │ ├── Category.php

│ │ ├── Comment.php

│ │ └── Post.php

│ ├── lang

│ │ └── zh-cn

│ │ └── blog

│ ├── model

│ │ ├── BlogCategory.php

│ │ ├── BlogComment.php

│ │ └── BlogPost.php

│ └── view

│ └── blog

│ ├── category

│ │ ├── add.html

│ │ ├── edit.html

│ │ └── index.html

│ ├── comment

│ │ ├── add.html

│ │ ├── edit.html

│ │ └── index.html

│ └── post

│ ├── add.html

│ ├── edit.html

│ └── index.html

├── assets

│ ├── css

│ │ ├── main.css

│ │ └── main.css.map

│ ├── fonts

│ ├── img

│ ├── js

│ ├── less

│ └── lib

│ ├── jquery.js

│ └── jquery.pjax.js

├── controller

│ ├── Ajax.php

│ └── Index.php

├── lang

│ └── zh-cn.php

├── model

│ ├── Category.php

│ ├── Comment.php

│ └── Post.php

├── public

│ └── assets

│ └── js

│ └── backend

│ └── blog

│ ├── category.js

│ ├── comment.js

│ └── post.js

├── view

│ ├── common

│ │ ├── commentlist.html

│ │ ├── postlist.html

│ │ └── sidebar.html

│ └── index

│ ├── archieve.html

│ ├── category.html

│ ├── index.html

│ └── post.html

├── Blog.php

├── bootstrap.js

├── LICENSE

├── config.php

├── info.ini

└── install.sql

--

**二、目录介绍**

blog

├── application //此文件夹中所有文件会覆盖到根目录的/application文件夹

├── assets //此文件夹中所有文件会复制到/public/assets/addons/blog文件夹

├── controller //此文件夹为插件控制器目录

├── lang //此文件夹为插件语言包目录

├── model //此文件夹为插件模型目录

├── public //此文件夹中所有文件会覆盖到根目录的/public文件夹

├── view //此文件夹为插件视图目录

├── Blog.php //此文件为插件核心安装卸载控制器,必需存在

├── bootstrap.js //此文件为插件JS启动文件

├── LICENSE //版权文件

├── config.php //插件配置文件,我们在后台插件管理中点配置按钮时配置的文件,必需存在

├── info.ini //插件信息文件,用于保存插件基本信息，插件开启状态等,必需存在

└── install.sql //插件数据库安装文件,此文件仅在插件安装时会进行导入

//

其中的application和public文件夹会覆盖到根目录，这两个文件夹主要用于我们后台管理功能的开发，我们可以先在后台开发好对应的管理功能后，再将对应的功能打包进插件即可，FastAdmin在插件安装和卸载时会自动进行文件冲突检测，如果遇到冲突的文件会提醒用户是否进行覆盖或删除。

//

assets这个文件夹很关键，FastAdmin会将assets中的所有文件夹和文件复制到/public/assets/addons/blog/文件夹中去，这个blog即是我们的插件目录名称，assets文件夹中的所有文件不会进行文件冲突检测，/public/assets/addons/blog/这个目录下的文件，我们在视图文件中可以直接通过"\_\_ADDON\_\_"指向这个路径。因此在开发视图时我们可以先使用相对路径设计，完成后我们再统一加上这个\_\_ADDON\_\_的前缀。

//

controller、lang、model和view这四个文件夹是我们插件前台功能的MVC部分，这部分文件夹不会复制或移动到其它位置。

//

Blog.php这个文件是插件的核心文件，我们可以在这个文件中编写插件安装或卸载时执行的脚本，或者在此插件中编写菜单的生成或删除，同时插件的行为方法也是编写在此文件中的，插件所支持的行为事件会在后面讲到。此文件命令规则为插件目录名称首字母大写。

//

bootstrap.js这个文件是插件的启动文件，插件在安装完启用后，FastAdmin会将此文件中的内容合并到/public/assets/js/addons.js中去，你可以在此编写插件核心JS或注册事件，在此JS中可以使用require依赖其它模块。同时在此插件中可以使用Fast、Backend、Lang等全局对象，因为在此之前此类对象已经加载且注册。

//

config.php这个文件是插件的配置文件，我们在后台插件管理中点配置按钮时会保存在此文件，此文件的内容格式为如下所述。

<?php

return [

[

//配置名称,该值在当前数组配置中确保唯一

'name' => 'yourname',

//配置标题

'title' => '配置标题',

//配置类型,支持string/text/number/datetime/array/select/selects/image/images/file/files/checkbox/radio/bool

'type' => 'string',

//配置select/selects/checkbox/radio/bool时显示的列表项

'content' => [

'1' => '显示',

'0' => '不显示'

],

//配置值

'value' => '1',

//配置验证规则,更多规则可参考nice-validator文件

'rule' => 'required',

'msg' => '验证失败提示文字',

'tip' => '字段填写帮助',

'ok' => '验证成功提示文字'

],

[

'name' => 'yourname2',

'title' => '配置标题2',

'type' => 'radio',

'options' => [

'1' => '显示',

'0' => '不显示'

],

'value' => '1',

'rule' => 'required',

'msg' => '验证失败提示文字',

'tip' => '字段填写帮助',

'ok' => '验证成功提示文字'

],

[

'name' => '\_\_tips\_\_',

'title' => '温馨提示',

'type' => 'string',

'content' =>

array(),

'value' => '该提示将出现的插件配置头部，通常用于提示和说明',

'rule' => '',

'msg' => '',

'tip' => '',

'ok' => '',

'extend' => '',

],

];

config.php中的值在FastAdmin任何地方均可使用get\_addon\_config('blog')来获取配置值。

//

info.ini这个文件仅用于保存插件基础信息和开启状态，此文件的内容格式为如下所述。

name = blog

title = 博客插件

intro = 响应式博客插件，包含日志、评论、分类、归档等

author = Karson

website = https://www.fastadmin.net

version = 1.0.0

state = 1

注意这个name是插件的唯一标识，不能和现在的插件冲突，其次注意下这个name值，如果我们有涉及到数据库，那个表的前缀也必须是\_\_PREFIX\_\_标识名开头。比如我们下面的\_\_PREFIX\_\_blog\_category。

//

install.sql 这个文件中只能是SQL语句，同时在此文件中可以使用\_\_PREFIX\_\_表示数据库表前缀，FastAdmin在安装导入SQL时自动进行替换。此文件的内容格式为如下所述。

;

; 我们在创建数据库时最好加上IF NOT EXISTS

CREATE TABLE IF NOT EXISTS `\_\_PREFIX\_\_blog\_category` (

`id` int(10) unsigned NOT NULL AUTO\_INCREMENT,

`pid` int(10) unsigned NOT NULL DEFAULT '0' COMMENT '父分类ID',

`name` varchar(30) NOT NULL DEFAULT '' COMMENT '分类名称',

`nickname` varchar(50) NOT NULL DEFAULT '' COMMENT '分类昵称',

`flag` set('hot','index','recommend') NOT NULL DEFAULT '' COMMENT '分类标志',

`image` varchar(100) NOT NULL DEFAULT '' COMMENT '图片',

`keywords` varchar(255) NOT NULL DEFAULT '' COMMENT '关键字',

`description` varchar(255) NOT NULL DEFAULT '' COMMENT '描述',

`diyname` varchar(30) NOT NULL DEFAULT '' COMMENT '自定义名称',

`createtime` int(10) unsigned NOT NULL DEFAULT '0' COMMENT '创建时间',

`updatetime` int(10) unsigned NOT NULL DEFAULT '0' COMMENT '更新时间',

`weigh` int(10) NOT NULL DEFAULT '0' COMMENT '权重',

`status` varchar(30) NOT NULL DEFAULT '' COMMENT '状态',

PRIMARY KEY (`id`),

KEY `weigh` (`weigh`,`id`),

KEY `pid` (`pid`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='博客分类表';

;

; 在执行插入时最好加上BEGIN;和COMMIT;

BEGIN;

INSERT INTO `\_\_PREFIX\_\_blog\_category` VALUES ('1', '0', '默认分类', 'default', '', '/assets/img/qrcode.png', '', '', '', '1502112587', '1502112587', '0', 'normal');

COMMIT;

--

**三、行为事件**

FastAdmin中的行为支持ThinkPHP5的所有行为，同时FastAdmin自定义部分专属的行为事件，以下是所有所支持的行为事件。

// 行为事件(标签位 描述 类型说明)

app\_init 应用初始化标签位 系统

app\_begin 应用开始标签位 系统

module\_init 模块初始化标签位 系统

action\_begin 控制器开始标签位 系统

view\_filter 视图输出过滤标签位 系统

app\_end 应用结束标签位 系统

log\_write 日志write方法标签位 系统

log\_write\_done 日志写入完成标签位 系统

response\_end 输出结束标签位 系统

response\_send 响应发送标签位 系统

upload\_after 上传成功标签位 FastAdmin

login\_init 登录标签位 FastAdmin

wipecache\_after 清除缓存后标签位 FastAdmin

admin\_nologin 管理员未登录标签位 FastAdmin

admin\_nopermission 管理员无权限标签位 FastAdmin

upload\_config\_init 上传配置标签位 FastAdmin

config\_init 系统配置标签位 FastAdmin

//

使用行为时在Blog.php中添加上对应的方法，FastAdmin在安装时、禁用、启用即可自动注册行为。但一定请注意在Blog.php中编写行为方法是使用的是驼峰式规则，例如upload\_after，方法名则为uploadAfter，如果方法名使用upload\_after则不会注册成功。至此，FastAdmin插件开发所涉及到的文件和文件夹已经介绍完了，如果有疑问，请及时在群或论坛反馈。

--

**四、插件示例**

我们准备了一篇插件开发简明教程，请点击这里查看(点击这里查看)。

--

**五、插件管理**

FastAdmin插件分为在线安装和命令行安装，在线安装可以直接在后台插件管理进行安装和卸载。命令行安装适合禁用或移除了后台插件管理功能的开发者使用，具体请参考命令行章节：一键管理插件(一键管理插件)。

--

**六、插件市场**

目前FastAdmin官方已经上线插件市场，开发者可以在插件市场下载插件进行离线安装，地址：<https://www.fastadmin.net/store.html>。如果你开发了一款插件需要上架到FastAdmin的插件市场，可以通过在线发布插件的形式分享或出售你的插件或应用，请点击 <https://www.fastadmin.net/postaddon.html> 查看。

--

**七、常见问题**

你在本地离线安装自己开发的插件时提示配置文件未找到时，请检查你的压缩包是否嵌套了目录，请注意不要嵌套目录，直接打包插件文件。

离线上传插件时如果提示上传的插件已经存在，请检查addons目录是否有对应的插件目录，务必移除才可以进行离线安装。

离线上传插件时如果提示Duplicate entry '菜单规则名称' for key 'name'时，请检查后台权限管理->规则管理，是否有存在相同的规则名称，你可以删除或重命名后再进行上传插件。

--

--

[PHP]\_TP5.0\_FastAdmin\_5、架构

(整理 <https://doc.fastadmin.net/docs/construct.html>)

--

**一、架构总览**

FastAdmin基于MVC的设计模式，将我们的应用分为三层（模型M、视图V、控制器C）。

--

**二、目录结构**

FastAdmin目录结构遵循ThinkPHP5官方建议的模块设计。

//// FastAdmin项目目录

├── addons //插件存放目录

├── application //应用目录

│ ├── admin //后台管理应用模块

│ ├── api //API应用模块

│ ├── common //通用应用模块

│ ├── extra //扩展配置目录

│ ├── index //前台应用模块

│ ├── build.php

│ ├── command.php //命令行配置

│ ├── common.php //通用辅助函数

│ ├── config.php //基础配置

│ ├── database.php //数据库配置

│ ├── route.php //路由配置

│ ├── tags.php //行为配置

├── extend

│ └── fast //FastAdmin扩展辅助类目录

├── public

│ ├── assets

│ │ ├── build //打包JS、CSS的资源目录

│ │ ├── css //CSS样式目录

│ │ ├── fonts //字体目录

│ │ ├── img

│ │ ├── js

│ │ │ ├── backend

│ │ │ └── frontend //后台/前台功能模块JS文件存放目录

│ │ ├── libs //Bower资源包位置

│ │ └── less //Less资源目录

│ └── uploads //上传文件目录

│ ├── index.php //应用入口主文件

│ ├── install.php //FastAdmin安装引导

│ ├── admin.php //后台入口文件,强烈建议修改

│ ├── robots.txt

│ └── router.php

├── runtime //缓存目录

├── thinkphp //ThinkPHP5框架核心目录

├── vendor //Composer资源包位置

├── .bowerrc //Bower目录配置文件

├── LICENSE

├── README.md

├── bower.json //Bower前端包配置

├── build.php

├── composer.json //Composer包配置

└── think

--

**三、应用模块**

在FastAdmin中默认有四个应用模块：admin、api、common、index，你也可以扩展开发自己的应用模块。后台模块(admin)是FastAdmin中的核心模块，后台模块又分为系统配置、附件管理、分类管理、插件管理等多个功能模块，更多的功能模块可以在插件管理中自由的安装和卸载。后台的前端是基于AdminLTE和Bootstrap进行了大量二次开发，采用RequireJS进行JS模块化管理和加载。前台模块(index)的结构和后台功能类似，具体请参考后台模块的章节。公共模块(common)是一个特殊的模块，默认是禁止直接访问的，一般用于放置一些公共的类或其它模块的继承基类等。Api模块(api)通常用于对接APP，用于向APP提供接口，目前FastAdmin暂未提供API相关的插件和文档，你可以直接参考ThinkPHP5官方的文档。

--

**四、功能模块**

功能模块指后台管理中的功能模块，比如我们的系统配置、附件管理、分类管理。后台开发的每一个功能模块都是基于MVC的设计模式进行开发 。在FastAdmin中，我们提供了一键生成CRUD的功能，这个一键生成CRUD生成的文件也就是我们标准的MVC文件。

//

//// 标准功能模块文件

├── application

│ └── admin

│ ├── controller

│ │ └── Test.php //控制器类

│ ├── lang

│ │ ├── zh-cn

│ │ │ └── test.php //功能语言包,按需加载

│ │ └── zh-cn.php //后台语言包,默认加载

│ ├── model

│ │ └── Test.php //模型类

│ ├── validate

│ │ └── Test.php //验证器类

│ └── view

│ └── test

│ ├── index.html //列表视图

│ ├── add.html //添加视图

│ └── edit.html //编辑视图

└── public

└── assets

└── js

└── backend

└── test.js //功能模块JS文件

在FastAdmin中每一个功能模块至少对应一个功能模块JS文件，也就是说每一个控制器都对应一个同名的JS文件，其次每一个控制器的方法对应JS文件中同名的方法。具体控制器详细介绍可参考控制器章节，JS的部分可以参考前端章节。

--

--

[PHP]\_TP5.0\_FastAdmin\_6、多语言

(整理 <https://doc.fastadmin.net/docs/language.html>)

--

在FastAdmin中可以在任何位置(控制器、视图、JS)使用"\_\_('语言标识');"调用语言包，如果语言标识不存在，则直接输出该语言标识。

**一、使用方法**

FastAdmin中的“\_\_函数”和ThinkPHP中的lang函数在传参上有些许区别，比如"\_\_('My name is %s', "FastAdmin");"将会返回“My name is FastAdmin”。而如果采用ThinkPHP中的lang中的写法则是“lang('My name is %s', ["FastAdmin"]);”，可以看到ThinkPHP中的第二个参数必须传入数组，而FastAdmin中的“\_\_”则没有这个要求，其实在多个参数时FastAdmin会忽略掉第三个参数$lang

比如“\_\_('This is %s,base on %s', 'FastAdmin', 'ThinkPHP5');”，则会返回“This is FastAdmin,base on ThinkPHP5”。而采用lang的写法则是“lang('This is %s,base on %s', ['FastAdmin', 'ThinkPHP5']);”，因此如果要使第三个参数$lang生效，则只能将第二个参数传为数组或采用ThinkPHP中的lang函数。如果需要在HTML视图文件中使用多语言，则需要使用"{:\_\_('Home')}"的方式调用，而在PHP和JS中均可以使用"\_\_('Home')"的方式发起调用。如果我们调用的多语言不存在，则会原样输出字符。例如我们调用"\_\_('Undefined')"，则会原样输出Undefined字符。

--

**二、加载方式**

在FastAdmin当中，框架会自动按照当前请求的控制器进行加载对应的语言包。例如当前我们是中文环境，如果我们请求的是<https://demo.fastadmin.net/admin/dashboard/index>，则FastAdmin会自动加载application/admin/lang/zh-cn.php、application/admin/lang/zh-cn/Dashboard.php这两个语言包。如果我们请求的路径是<https://demo.fastadmin.net/admin/general/config/index>，则FastAdmin会自动加载application/admin/lang/zh-cn.php、application/admin/lang/zh-cn/general/Config.php这两个语言包。可以看到FastAdmin会默认加载zh-cn.php这个全局语言包。如果我们需要跨模块引入其它模块的语言包，则可以在控制器中使用loadlang方法来引入，如“$this->loadlang('模块名');”。如果需要在JS中跨模块引入语言包，则需要修改Ajax.php中的lang这个方法。

--

三、常见问题

语言包定义是不区分大小写的。

如果JS中对应语言包不生效，请检查服务端语言包是否有错误。

如果你不希望启用多语言，则可以在application/config.php中关闭和设置默认语言。

--

--

[PHP]\_TP5.0\_FastAdmin\_7、数据库

(整理 <https://doc.fastadmin.net/docs/database.html>)

--

这里提供的是数据库表字段规则在你创建表时使用，当按如下的规则进行字段命名、类型设置和备注时使用"php think crud -t 表名"生成CRUD时会自动生成对应的HTML元素和组件。

--

**一、根据字段类型**

int 整型 自动生成type为number的文本框，步长为1。

enum 枚举型 自动生成单选下拉列表框。

set set型 自动生成多选下拉列表框。

float 浮点型 自动生成type为number的文本框，步长根据小数点位数生成。

text 文本型 自动生成textarea文本框。

datetime 日期时间 自动生成日期时间的组件。

date 日期型 自动生成日期型的组件。

timestamp 时间戳 自动生成日期时间的组件。

--

**二、特殊字段**

**category\_id** 分类ID int 将生成选择分类的下拉框,分类类型根据去掉前缀的表名，单选。

**category\_ids** 多选分类ID varchar 将生成选择分类的下拉框,分类类型根据去掉前缀的表名，多选。

**weigh** 权重 int 后台的排序字段，如果存在该字段将出现排序按钮，可上下拖动进行排序。

**createtime** 创建时间 int 记录添加时间字段,不需要手动维护。

**updatetime** 更新时间 int 记录更新时间的字段,不需要手动维护。

**deletetime** 删除时间 int 记录删除时间的字段,不需要手动维护,如果存在此字段将会生成回收站功能,字段默认值务必为null。

**status** 状态字段 enum 列表筛选字段,如果存在此字段将启用TAB选项卡展示列表。

--

**三、以特殊字符结尾的规则**

结尾字符 示例 类型要求 字段说明

**time** refreshtime int 识别为日期时间型数据，自动创建选择时间的组件。

**image** smallimage varchar 识别为图片文件，自动生成可上传图片的组件,单图。

**images** smallimages varchar 识别为图片文件，自动生成可上传图片的组件,多图。

**file** attachfile varchar 识别为普通文件，自动生成可上传文件的组件,单文件。

**files** attachfiles varchar 识别为普通文件，自动生成可上传文件的组件,多文件。

**avatar** miniavatar varchar 识别为头像，自动生成可上传图片的组件,单图。

**avatars** miniavatars varchar 识别为头像，自动生成可上传图片的组件,多图。

**content** maincontent text 识别为内容，自动生成富文本编辑器(需安装富文本插件)。

**\_id** user\_id int/varchar 识别为关联字段，自动生成可自动完成的文本框，单选。

**\_ids** user\_ids varchar 识别为关联字段，自动生成可自动完成的文本框，多选。

**list** timelist enum 识别为列表字段，自动生成单选下拉列表。

**list** timelist set 识别为列表字段，自动生成多选下拉列表。

**data** hobbydata enum 识别为选项字段，自动生成单选框。

**data** hobbydata set 识别为选项字段，自动生成复选框。

**json** configjson varchar 识别为键值组件，自动生成键值录入组件。

**switch** siteswitch tinyint 识别为开关字段，自动生成开关组件。

**温馨提示：**以list或data结尾的字段必须搭配enum或set类型才起作用

--

**四、注释说明**

字段 注释内容 字段类型 字段说明

**status 状态 int** 将生成普通语言包和普通文本框。

**status 状态 enum(‘0’,’1’,’2’)** 将生成普通语言包和单选下拉列表,同时生成TAB选项卡。

**status 状态:0=隐藏,1=正常,2=推荐 enum(‘0’,’1’,’2’)** 将生成多个语言包和单选下拉列表,同时生成TAB选项卡,且列表中的值显示为对应的文字##。

--

**五、常见问题**

1、如果使用php think crud生成过表的CRUD，当修改了表结构类型或新增了字段，此时只有重新生成CRUD或自己手动修改视图文件和JS文件。

2、如果你的表需要生成关联模型，建议你在设计表时遵循以下规范，category\_id字段关联对应fa\_category表主键，company\_info\_id关联对应fa\_company\_info表主键。

3、如果字段名为user\_id，生成的动态下拉列表会自动匹配user/index这个控制器方法，如果你发现CRUD后无法显示关联数据列表，请检查视图中元素data-source的值是否正确。

--

--

[PHP]\_TP5.0\_FastAdmin\_8、控制器

(整理 <https://doc.fastadmin.net/docs/controller.html>)

--

**一、基类控制器**

FastAdmin中定义了三个基类控制器，分别位于如下目录。

application/common/controller/Api.php //API接口基类控制器；

application/common/controller/Backend.php //后台基类控制器；

application/common/controller/Frontend.php //前台基类控制器；

这里我们主要对Backend.php这个控制器基类做详细解读，因为我们后台管理的所有控制器都继承于它，而Frontend.php功能和Backend.php功能类似，而Api.php控制器主要用于API接口开发，这里就不再进行过多介绍。

--

**二、基础结构**

在后台中我们的控制器都必须继承自\app\common\controller\Backend这个基类，其中控制器的index/add/edit/del/multi/recyclebin/destroy/restore/import/selectpage全都是可选的方法，基类的这些方法使用traits进行引入，具体位置在application/admin/library/traits/Backend.php中，CRUD生成的标准控制器如下描述。

<?php

namespace app\admin\controller;

/\*\*

\* 文章管理

\*

\* @icon fa fa-list

\* @remark 用于统一管理网站的所有文章

\*/

class Article extends Backend

{

protected $model = null;

protected $noNeedLogin = [];

protected $noNeedRight = ['selectpage'];

public function \_initialize()

{

parent::\_initialize();

}

/\*\*

\* 默认生成的控制器所继承的父类中有index/add/edit/del/multi/destroy/restore/recyclebin八个方法，

\* 因此在当前控制器中可不用编写增删改查的代码,如果需要自己控制这部分逻辑，

\* 需要将application/admin/library/traits/Backend.php中对应的方法复制到当前控制器,然后进行修改。

\*/

}

基类中所定义的方法如下，以下方法都是通过application/admin/library/traits/Backend.php引入的。

class Backend extends Controller{

/\*\*

\* 查看

\*/

public function index(){}

/\*\*

\* 添加

\*/

public function add($ids = ""){}

/\*\*

\* 编辑

\*/

public function edit($ids = ""){}

/\*\*

\* 删除

\*/

public function del($ids = ""){}

/\*\*

\* 批量更新

\*/

public function multi($ids = ""){}

/\*\*

\* 回收站

\*/

public function recyclebin(){}

/\*\*

\* 真实删除

\*/

public function destroy($ids = ""){}

/\*\*

\* 还原

\*/

public function restore($ids = ""){}

/\*\*

\* 导入

\*/

protected function import(){}

/\*\*

\* 下拉筛选

\*/

public function selectpage()

{

return parent::selectpage();

}

}

我们在开发过程中最好注释好每一个控制器和控制器的方法，因为我们后期可以使用php think menu -c all-controller一键生成后台管理的菜单，注释支持@icon/@remark/@internal这三个属性，分别表示图标/备注/忽略，如果是protected或private的方法在后期一键生成时会自动忽略。控制器支持目录层级，如果在使用目录层级的时候，请注意当前控制器的命名空间，比如当前控制器文件位置是application/admin/controller/member/Teacher.php，当Teacher.php的命名空间请务必是"namespace app\admin\controller\member;"，如果命名空间不对会报找不到控制器的错误。

--

**三、属性和方法**

**//// 属性**

当我们的控制器继承自app\common\controller\Backend以后，我们就可以使用以下属性。

/\*\*

\* 无需登录的方法,同时也就不需要鉴权了

\* @var array

\*/

protected $noNeedLogin = [];

/\*\*

\* 无需鉴权的方法,但需要登录

\* @var array

\*/

protected $noNeedRight = [];

/\*\*

\* 布局模板

\* @var string

\*/

protected $layout = 'default';

/\*\*

\* 权限控制类

\* @var Auth

\*/

protected $auth = null;

/\*\*

\* 快速搜索时执行查找的字段

\*/

protected $searchFields = 'id';

/\*\*

\* 是否是关联查询

\*/

protected $relationSearch = false;

/\*\*

\* 是否开启数据限制

\* 支持auth/personal

\* 表示按权限判断/仅限个人

\* 默认为禁用,若启用请务必保证表中存在admin\_id字段

\*/

protected $dataLimit = false;

/\*\*

\* 数据限制字段

\*/

protected $dataLimitField = 'admin\_id';

/\*\*

\* 是否开启Validate验证

\*/

protected $modelValidate = false;

/\*\*

\* 是否开启模型场景验证

\*/

protected $modelSceneValidate = false;

/\*\*

\* Multi方法可批量修改的字段

\*/

protected $multiFields = 'status';

//

我们可以直接在当前控制器使用$this->属性名来调用所支持的属性，也支持直接在当前控制器定义相关属性来覆盖父类的属性，同时TP5中\think\Controller所支持的属性也全部支持。

//

**//// 方法**

基类app\common\controller\Backend中所支持的方法如下所述。

/\*\*

\* 加载语言文件

\* @param string $name

\*/

protected function loadlang($name)

{

}

/\*\*

\* 渲染配置信息

\* @param mixed $name 键名或数组

\* @param mixed $value 值

\*/

protected function assignconfig($name, $value = '')

{

}

/\*\*

\* 生成查询所需要的条件,排序方式

\* @param mixed $searchfields 快速查询的字段

\* @param boolean $relationSearch 是否关联查询

\* @return array

\*/

protected function buildparams($searchfields = null, $relationSearch = null)

{

}

/\*\*

\* 获取数据限制的管理员ID

\* 禁用数据限制时返回的是null

\* @return mixed

\*/

protected function getDataLimitAdminIds()

{

}

/\*\*

\* Selectpage的实现方法

\*

\* 当前方法只是一个比较通用的搜索匹配,请按需重载此方法来编写自己的搜索逻辑,$where按自己的需求写即可

\* 这里示例了所有的参数，所以比较复杂，实现上自己实现只需简单的几行即可

\*

\*/

protected function selectpage()

{

}

我们可以直接在当前控制器使用$this->方法名()来调用所支持的方法，同时TP5中\think\Controller所支持的方法也全部支持。

--

**四、数据限制**

在后台开发的过程中经常会有这样的一个需求，每个管理员单独管理自己添加的数据或单独管理自己下级管理员添加的数据，管理员之间的数据是不相通的，每个管理员看到的数据是不同的。在FastAdmin中可以很方便的实现此功能。

//

// 首先我们需要在当前控制器添加以下两个属性

protected $dataLimit = 'auth'; //默认基类中为false，表示不启用，可额外使用auth和personal两个值；

protected $dataLimitField = 'admin\_id'; //数据关联字段,当前控制器对应的模型表中必须存在该字段；

//

$dataLimit = false; //表示不启用，显示所有数据;

$dataLimit = 'auth'; //表示显示当前自己和所有子级管理员的所有数据;

$dataLimit = 'personal'; //表示仅显示当前自己的数据;

//

$dataLimitField字段默认为admin\_id，请注意添加该字段类型为int(10)。通过以上配置后，在列表数据的时候将默认添加条件过滤不属自己权限的数据，在添加时会自动维护admin\_id的数据，在编辑、删除的时候会自动控制权限避免越权操作。如果需要将原有的数据加入到FastAdmin后台管理权限控制当中，比如已有的数据已经有标识归属，但这个归属体系并非是FastAdmin的后台管理员体系。在这个时候我们就需要重写基类的getDataLimitAdminIds方法，将此方法返回数据标识的归属ID数组集合，这样即可使用FastAdmin的后台管理权限进行管理。

--

**五、关联查询**

目前FastAdmin后台index方法支持一对一关联查询，比如我们一篇文章有归属分类，我们在列出数据时需要同时列表文章分类名称。

//

// 首先我们需要在当前控制器中添加以下属性

protected $relationSearch = true;

//

// 然后我们修改控制器的index方法

public function index()

{

if ($this->request->isAjax())

{

list($where, $sort, $order, $offset, $limit) = $this->buildparams();

$total = $this->model

->with(["category"])

->where($where)

->order($sort, $order)

->count();

$list = $this->model

->with(["category"])

->where($where)

->order($sort, $order)

->limit($offset, $limit)

->select();

$result = array("total" => $total, "rows" => $list);

return json($result);

}

return $this->view->fetch();

}

//

// 然后在控制器对应的model(非关联model)中添加以下代码

public function category()

{

// 设置预载入查询方式为JOIN方式

return $this->belongsTo('Category', 'category\_id')->setEagerlyType(0);

}

更多的关联用户可以参考TP5关联模型的章节：关联模型( <https://www.kancloud.cn/manual/thinkphp5/142357> )。

//

// 我们在控制器对应的JS中可以直接使用category.id、category.name等关联表的字段

// 初始化表格

table.bootstrapTable({

url: $.fn.bootstrapTable.defaults.extend.index\_url,

columns: [

[

{field: 'state', checkbox: true, },

{field: 'id', title: 'ID', operate: '='},

{field: 'title', title: \_\_('Title'), operate: 'LIKE %...%'},

{field: 'category.image', title: \_\_('Image'), operate: false, formatter: Table.api.formatter.image},

{field: 'category.name', title: \_\_('Name'), operate: '='},

{field: 'ip', title: \_\_('IP'), operate: '='},

{field: 'createtime', title: \_\_('Create time'), formatter: Table.api.formatter.datetime, operate: 'RANGE', addclass: 'datetimerange'},

{field: 'operate', title: \_\_('Operate'), table: table, events: Table.api.events.operate, formatter: Table.api.formatter.operate}

]

],

});

--

**六、数据校验**

在FastAdmin中默认的add/edit方法可以使用模型验证，验证器位于application/admin/validate/模型名.php中，模型验证默认是关闭的状态，如果需要启用，我们需要在当前控制器定义以下属性。

// 启用数据验证

protected $modelValidate = true; //是否开启Validate验证，默认是false关闭状态;

protected $modelSceneValidate = true; //是否开启模型场景验证，默认是false关闭状态;

当开启模型验证后，我们的添加和修改操作都会首先进行模型验证，验证不通过将会抛出错误信息，具体的模型验证规则可以参考TP5官方文档的模型验证规则：<https://www.kancloud.cn/manual/thinkphp5/129355>。场景验证可以参考TP5场景验证章节：<https://www.kancloud.cn/manual/thinkphp5/129322>。

--

**七、权限控制**

// 在基类中我们有定义以下两个属性;

protected $noNeedLogin = []; //无需登录的方法,同时也就不需要鉴权了;

protected $noNeedRight = []; //无需鉴权的方法,但需要登录;

比如我们有定义一个方法mywork，而这个方法是不需要登录即可访问的，则我们需要在当前的控制器定义“protected $noNeedLogin = ['mywork'];”。比如我们有定义一个方法mytest，而这个方法是需要登录后任何管理员都可以访问，则我们需要在当前的控制器定义“protected $noNeedRight = ['mytest'];”。如果我们需要动态定义，请务必放在调用父类的\_initialize方法之前，否则是不会生效的。

--

**八、视图渲染**

基类app\common\controller\Backend会默认渲染以下几个对象到视图中。

// 渲染站点配置

$this->assign('site', $site);

// 渲染配置信息

$this->assign('config', $config);

// 渲染权限对象

$this->assign('auth', $this->auth);

// 渲染管理员对象

$this->assign('admin', Session::get('admin'));

我们可以在视图中使用{$site.name}、{$config.modulename}、{$auth.id}、{$admin.username}来获取我们所需要的数据。

//

// $site所支持的数据对应为application/extra/site.php(后台配置保存入数据库后写入配置文件)

// $config所支持的数据

'site' => $site中的'name', 'cdnurl', 'version', 'timezone', 'languages'字段,

'upload' => application/extra/upload.php中数据,

'modulename' => 'admin',

'controllername' => 控制器名,

'actionname' => 方法名,

'jsname' => 控制器JS所加载的路径,

'moduleurl' => 后台module的链接,

'language' => 当前语言,

'fastadmin' => application/config.php中fastadmin的配置

//

$auth是一个对象，所对应的类是application/admin/library/Auth.php。

$admin是一个当前管理员登录的session数据，存储有管理员的用户名、昵称、ID、头像等信息。

//

// 如果我们需要在JS中使用以上数据，则使用如下方式来获取相关的配置信息。

Config.site.name

Config.modulename

//

如果我们需要自己在控制器中透传数据到JS中去，则可以使用控制器的assignconfig方法来透传，使用“$this->assignconfig('demo', ['name'=>'名称']);”，然后我们就可以在JS中使用“Config.demo.name”来获取对应的数据。

--

**九、模板布局**

控制器默认全部采用模板布局，因此我们的页面都会自动加上头部和尾部，如果我们有特殊的页面不需要采用模板布局，我们可以使用“$this->view->engine->layout(false);”来关闭当前方法的模板布局。如果我们需要使用自己的模板布局，在当前控制器定义“protected $layout = '布局模板';”即可。请注意如果采用了自己的模板布局或禁用了模板布局，将无法使用FastAdmin的JS按需加载和Config变量访问。

--

--

[PHP]\_TP5.0\_FastAdmin\_9、模块；

--王尔贝(我是害虫)

(整理 <https://doc.fastadmin.net/docs/module.html>)

--

**一、前台**

这里的前台指整个前台index模块，这里仅做部分前台功能使用介绍，如果需要查看前端开发文档，请查看前端章节的文档。FastAdmin的前台首页比较简单，只有一个单页面。同时在FastAdmin中我们编写了一个简单的会员中心，只有简单的注册、登录、找回密码、个人中心等。其它功能都需要自己二次开发。当然FastAdmin中提供有CMS和博客插件，这两个插件都有完整的前后台功能。

**1、流程介绍**

FastAdmin的前台模块完全遵循ThinkPHP5的开发规范，在此基础上我们在前台提供了一个类似后台的权限验证模块，我们可以方便快捷地控制我们的可访问权限。

**2、基类解析**

//// 基类控制器

前台的所有功能模块的控制器都是继承于application/common/controller/Frontend.php这个基类控制器。在基类控制器中我们有定义一些基础属性和通用方法。

//

//// 基础属性

/\*\*

\* 布局模板

\* @var string

\*/

protected $layout = '';

/\*\*

\* 无需登录的方法,同时也就不需要鉴权了

\* @var array

\*/

protected $noNeedLogin = [];

/\*\*

\* 无需鉴权的方法,但需要登录

\* @var array

\*/

protected $noNeedRight = [];

/\*\*

\* 权限Auth

\* @var Auth

\*/

protected $auth = null;

//

//// 基类方法

/\*\*

\* 加载语言文件

\* @param string $name

\*/

protected function loadlang($name)

{

}

/\*\*

\* 渲染配置信息

\* @param mixed $name 键名或数组

\* @param mixed $value 值

\*/

protected function assignconfig($name, $value = '')

{

}

//

以上的属性和方法我们都可以通过在当前控制器定义来达到覆盖的目的。

**3、功能模块**

**(1)、首页模块**

首页模块比较简单，只是一个单页。完全遵循ThinkPHP5的开发结构。你可以按需修改或移除此功能模块。

**(2)、会员模块**

FastAdmin的前台自带一个简单的会员功能模块，可以进行会员的注册、登录、找回密码、会员中心、修改个人资料、修改密码等操作。会员模块可用于进行前台会员功能开发时使用。此处的会员模块和API中的会员模块账号是相通的，但他们登录时是不会互相影响的，可以同时登录。FastAdmin的会员模块有注册几个事件，我们可以在事件中自定义我们的操作。你可以按照以下的方式监听相应的事件行为。

// 登录成功后的事件

Hook::add('user\_login\_successed', function ($user) {

});

// 注册成功后的事件

Hook::add('user\_register\_successed', function ($user) {

});

// 会员删除后的事件

Hook::add('user\_delete\_successed', function ($user) {

});

// 会员注销后的事件

Hook::add('user\_logout\_successed', function ($user) {

});

--

**二、API**

这里的API指整个API接口模块，这里仅做部分API功能模块的使用介绍，如果需要查看前端开发和后端开发文档，请查看相对应的文档。

--

**1、流程介绍**

FastAdmin的API模块完全遵循ThinkPHP5的开发规范，在此基础上我们在API模块中提供了一个和前台相同的权限验证模块，我们可以方便快捷的控制我们的可访问权限。FastAdmin的API模块采用默认的路由方式进行匹配，当然我们完全可以自定义我们的路由规则，达到个性化URL接口的目的。

--

**2、基类解析**

API的所有功能模块的控制器都是继承于application/common/controller/Api.php这个基类控制器。在基类控制器中我们有定义一些基础属性和通用方法。

//

//// 基础属性

/\*\*

\* @var array 前置操作方法列表

\*/

protected $beforeActionList = [];

/\*\*

\* 无需登录的方法,同时也就不需要鉴权了

\* @var array

\*/

protected $noNeedLogin = [];

/\*\*

\* 无需鉴权的方法,但需要登录

\* @var array

\*/

protected $noNeedRight = [];

/\*\*

\* 权限Auth

\* @var Auth

\*/

protected $auth = null;

//

//// 基类方法

/\*\*

\* 加载语言文件

\* @param string $name

\*/

protected function loadlang($name)

{

}

/\*\*

\* 操作成功返回的数据

\* @param string $msg 提示信息

\* @param mixed $data 要返回的数据

\* @param int $code 错误码，默认为1

\* @param string $type 输出类型

\* @param array $header 发送的 Header 信息

\*/

protected function success($msg = '', $data = null, $code = 1, $type = null, array $header = [])

{

}

/\*\*

\* 操作失败返回的数据

\* @param string $msg 提示信息

\* @param mixed $data 要返回的数据

\* @param int $code 错误码，默认为0

\* @param string $type 输出类型

\* @param array $header 发送的 Header 信息

\*/

protected function error($msg = '', $data = null, $code = 0, $type = null, array $header = [])

{

}

/\*\*

\* 返回封装后的 API 数据到客户端

\* @access protected

\* @param mixed $msg 提示信息

\* @param mixed $data 要返回的数据

\* @param int $code 错误码，默认为0

\* @param string $type 输出类型，支持json/xml/jsonp

\* @param array $header 发送的 Header 信息

\* @return void

\* @throws HttpResponseException

\*/

protected function result($msg, $data = null, $code = 0, $type = null, array $header = [])

{

}

/\*\*

\* 前置操作

\* @access protected

\* @param string $method 前置操作方法名

\* @param array $options 调用参数 ['only'=>[...]] 或者 ['except'=>[...]]

\* @return void

\*/

protected function beforeAction($method, $options = [])

{

}

/\*\*

\* 验证数据

\* @access protected

\* @param array $data 数据

\* @param string|array $validate 验证器名或者验证规则数组

\* @param array $message 提示信息

\* @param bool $batch 是否批量验证

\* @param mixed $callback 回调方法（闭包）

\* @return array|string|true

\* @throws ValidateException

\*/

protected function validate($data, $validate, $message = [], $batch = false, $callback = null)

{

}

//

以上的属性和方法我们都可以通过在当前控制器定义来达到覆盖的目的。

--

**3、功能模块**

**(1)、会员模块**

我们在FastAdmin的API中集成了简单的会员接口，可以进行会员的注册、登录、找回密码、会员中心、修改个人资料、修改密码等操作。会员模块可用于进行API会员功能开发时使用。此处的会员模块和前台中的会员模块账号是相通的，但他们登录时是不会互相影响的，可以同时登录。FastAdmin的会员模块有注册几个事件，我们可以在事件中自定义我们的操作。你可以按照以下的方式监听相应的事件行为。

// 登录成功后的事件

Hook::add('user\_login\_successed', function ($user) {

});

// 注册成功后的事件

Hook::add('user\_register\_successed', function ($user) {

});

// 会员删除后的事件

Hook::add('user\_delete\_successed', function ($user) {

});

// 会员注销后的事件

Hook::add('user\_logout\_successed', function ($user) {

});

**(2)、短信模块**

我们在FastAdmin的API中集成了简单的短信模块，可以根据相对应的事件进行短信的发送和检测是否正确等功能。在使用短信模块时请确保已经正确在后台正确安装并启用了第三方短信插件。

(3)、公共模块

公共模块一般用于客户端应用初始化时调用，例如APP的版本检测、APP的首屏轮换图等功能。

(4)、检测模块

检测模块一般用于检测客户端提交数据的有效性验证，也常用于在前台进行数据录入时的实时有效性校验。

--

**四、公共**

**1、Token验证**

**(1)、功能介绍**

Token验证主要用于会员登录状态信息的维护和验证，通常情况下不需要我们调用此类的方法，在一些特殊情况下我们可以手动调用。

**(2)、使用示例**

//// 获取Token信息

// 获取Token的详情、关联的会员ID、过期时间、有效期等信息

Token::get('c2259a37-5fee-4c4b-93b0-1d7313e1d1ac');

//

//// 设置会员的Token信息

// 新增会员Token并更新，且有效期为3600秒

Token::set('c2259a37-5fee-4c4b-93b0-1d7313e1d1ac', 1, 3600);

//

//// 判断会员Token是否可用

// 通过Token和会员ID来判断Token是否可以使用

Token::check('c2259a37-5fee-4c4b-93b0-1d7313e1d1ac', 1);

//

//// 删除单个会员Token

// 删除指定的Token

Token::delete('c2259a37-5fee-4c4b-93b0-1d7313e1d1ac');

//

//// 删除指定会员的所有Token

// 删除会员ID为1的所有Token

Token::clear(1);

--

**2、邮件发送**

**(1)、功能介绍**

FastAdmin中的邮件发送采用phpmailer进行邮件发送，在使用邮件发送功能前请先在后台常规管理->系统配置中配置好邮件的相关信息。

**(2)、使用示例**

首先我们需要采用单例或实例化一个Email对象。“$email = new \app\common\library\Email;”，其次我们可以设置邮件主题正文、接收者、标题等信息，比如“$email->subject('这里是邮件标题')->to('youremail@163.com')->message('这里是邮件正文')->send();”。如果我们邮件发送失败，想获取错误的详情，可使用“$email->getError();”来获取到错误详情。

--

**3、短信发送**

**(1)、功能介绍**

在我们开发过程中经常会用到短信发送和短信推广的功能，FastAdmin提供了一个简单实用的短信接口供开发者调用。在使用短信发送之前，务必在后台安装好我们短信服务商的插件，如果我们要使用的服务商未提供有FastAdmin的插件，我们则需要自己开发一个，或注册一个sms\_send和sms\_check的事件用于我们的发送和检测操作。

**(2)、使用示例**

// 首先最常用的是发送短信，比如我们发送一个注册验证码;

Sms::send('13888888888', '1234', 'register');

//

// 发送以后我们有时需要检测验证码是否正确，则可以使用;

Sms::check('13888888888', '1234', 'register');

//

// 当然有些时候我们还需要发送营销短信或通知，则可以使用;

Sms::notice('13888888888', '消息内容', 'SMS\_10001');

//

// 如果我们需要清空指定手机号的验证码，则可以使用;

Sms::flush('13888888888', 'register');

--

**4、辅助函数**

在FastAdmin中我们有提供几个常用的辅助函数。

/\*\*

\* 获取语言变量值

\* @param string $name 语言变量名

\* @param array $vars 动态变量值

\* @param string $lang 语言

\* @return mixed

\*/

function \_\_($name, $vars = [], $lang = '')

{

}

/\*\*

\* 将字节转换为可读文本

\* @param int $size 大小

\* @param string $delimiter 分隔符

\* @return string

\*/

function format\_bytes($size, $delimiter = '')

{

}

/\*\*

\* 将时间戳转换为日期时间

\* @param int $time 时间戳

\* @param string $format 日期时间格式

\* @return string

\*/

function datetime($time, $format = 'Y-m-d H:i:s')

{

}

/\*\*

\* 获取语义化时间

\* @param int $time 时间

\* @param int $local 本地时间

\* @return string

\*/

function human\_date($time, $local = null)

{

}

/\*\*

\* 获取上传资源的CDN的地址

\* @param string $url 资源相对地址

\* @return string

\*/

function cdnurl($url)

{

}

/\*\*

\* 判断文件或文件夹是否可写

\* @param string $file 文件或目录

\* @return bool

\*/

function is\_really\_writable($file)

{

}

/\*\*

\* 删除文件夹

\* @param string $dirname 目录

\* @param bool $withself 是否删除自身

\* @return boolean

\*/

function rmdirs($dirname, $withself = true)

{

}

/\*\*

\* 复制文件夹

\* @param string $source 源文件夹

\* @param string $dest 目标文件夹

\*/

function copydirs($source, $dest)

{

}

function mb\_ucfirst($string)

{

}

/\*\*

\* 附加关联字段数据

\* @param array $items 数据列表

\* @param mixed $fields 渲染的来源字段

\* @return array

\*/

function addtion($items, $fields)

{

}

/\*\*

\* 返回打印数组结构

\* @param string $var 数组

\* @param string $indent 缩进字符

\* @return string

\*/

function var\_export\_short($var, $indent = "")

{

}

--

--

[PHP]\_TP5.0\_FastAdmin\_10、前端

(整理 https://doc.fastadmin.net/docs/frontend.html)

--

**一、基础介绍**

FastAdmin的前端部分使用或涉及到主要是RequireJS,jQuery,AdminLTE,Bower,Less,CSS。

>RequireJS主要是用于JS的模块化加载；

>Bower主要用于管理第三方插件；

>Less主要是用于我们编写LESS和编译成CSS代码；

在阅读接下来的文档之前最好先简单的了解下RequireJS和Bower，而jQuery是我们必须要掌握的工具库。FastAdmin中前端的最常用的第三方插件有Layer(<http://layer.layui.com/>)，Toastr(<https://codeseven.github.io/toastr/>)。Layer用于弹窗，Toastr用于提示。

--

**二、目录结构**

public

├── assets

│ ├── addons

│ ├── css

│ ├── fonts

│ ├── img

│ ├── js

│ │ ├── backend

│ │ ├── frontend

│ ├── less

│ └── libs

//

assets主要存在我们框架所需要使用到的静态资源;

assets/js/backend主要存在后台控制器所对应的JS模块;

assets/libs主要存在第三方的插件;

assets/less主要存在Less文件;

assets/img主要存在图片相关文件;

assets/css主要存在CSS样式相关文件;

assets/addons主要存放插件的相关静态资源;

--

**三、标准模块**

在控制器章节我们有提到每个控制器都对应一个JS模块，控制器名称和JS中模块名称是一一对应的。比如我们的控制器是application/admin/controller/Article.php，则我们JS中对应的JS模块是public/assets/js/backend/article.js。如果我们的控制器是application/admin/controller/Member/Teacher.php，则我们JS中对应的JS模块是public/assets/js/backend/member/teacher.js。每一个控制器请求的方法对应JS模块中一个方法。比如控制器Article.php中的index方法，对应的是JS的article模块中的Controller.index方法，如果我们添加了自己的方法detail方法，则对应Controller.detail方法。

**1、JS标准控制器模块写法**

define(['jquery', 'bootstrap', 'backend', 'table', 'form'], function ($, undefined, Backend, Table, Form) {

var Controller = {

index: function () {

// 初始化表格参数配置

Table.api.init({

extend: {

index\_url: 'category/index',

add\_url: 'category/add',

edit\_url: 'category/edit',

del\_url: 'category/del',

multi\_url: 'category/multi',

dragsort\_url: '',

table: 'category',

}

});

var table = $("#table");

// 初始化表格

table.bootstrapTable({

url: $.fn.bootstrapTable.defaults.extend.index\_url,

escape: false,

pk: 'id',

sortName: 'weigh',

pagination: false,

commonSearch: false,

columns: [

[

{checkbox: true},

{field: 'id', title: \_\_('Id')},

{field: 'type', title: \_\_('Type')},

{field: 'name', title: \_\_('Name'), align: 'left'},

{field: 'nickname', title: \_\_('Nickname')},

{field: 'flag', title: \_\_('Flag'), operate: false, formatter: Table.api.formatter.flag},

{field: 'image', title: \_\_('Image'), operate: false, formatter: Table.api.formatter.image},

{field: 'weigh', title: \_\_('Weigh')},

{field: 'status', title: \_\_('Status'), operate: false, formatter: Table.api.formatter.status},

{field: 'operate', title: \_\_('Operate'), table: table, events: Table.api.events.operate, formatter: Table.api.formatter.operate}

]

]

});

// 为表格绑定事件

Table.api.bindevent(table);

},

add: function () {

Controller.api.bindevent();

},

edit: function () {

Controller.api.bindevent();

},

api: {

bindevent: function () {

Form.api.bindevent($("form[role=form]"));

}

}

};

return Controller;

});

//

我们可以看到该模块第一行为RequireJS标准语法，意思是此模块依赖'jquery', 'bootstrap', 'backend', 'table', 'form'几个模块，同时将它们加载为$, undefined, Backend, Table, Form这几个对应，我们就可以放心大胆在其中使用这几个对象了。其中有定义一个Controller 对象，它有index/add/edit/api四个方法，这四个方法分别与我们控制器中的方法一一对应。

--

**四、工具模块**

在FastAdmin中我们封装了非常多实用的模块类便于我们使用。以下做详细介绍。

--

**1、表单(Form)**

表单模块主要用于框架表单组件元素的渲染和事件绑定，当我们自定了一个表单后，必须使用表单模块中Form.api.bindevent进行绑定表单，否则不会有任何作用。

**(1)、引入方式**

require(['form'], function(Form){});

**(2)、内部方法**

// 为表单绑定事件，将自动绑定上传/富文本/下拉框/selectpage/表单验证等功能，FastAdmin中最常用的方法；

Form.api.bindevent(form, success, error, submit);

// 表单自定义事件存储对象

Form.api.custom

// 提交表单的方法，在表单完成验证后进行提交

Form.api.submit(form, success, error, submit);

// 以下事件为具体第三方插件实现的方法，可以在调用`Form.api.bindevent`之前修改对应的方法来修改相应功能；

Form.events.bindevent(form)

Form.events.citypicker(form)

Form.events.cxselect(form)

Form.events.datetimepicker(form)

Form.events.faselect(form)

Form.events.fieldlist(form)

Form.events.plupload(form)

Form.events.selectpage(form)

Form.events.selectpicker(form)

Form.events.validator(form, success, error, submit)

**(3)、使用示例**

Form.api.bindevent($("form[role=form]"), function(data, ret){

// 如果我们需要在提交表单成功后做跳转，可以在此使用'location.href="链接";'进行跳转;

Toastr.success("成功");

}, function(data, ret){

Toastr.success("失败");

}, function(success, error){

//bindevent的第三个参数为提交前的回调

//如果我们需要在表单提交前做一些数据处理，则可以在此方法处理

//注意如果我们需要阻止表单，可以在此使用"return false;"即可

//如果我们处理完成需要再次提交表单则可以使用submit提交,如下

//Form.api.submit(this, success, error);

return false;

});

以上代码表示当表单提交处理成功后提示成功，处理失败提示失败。

--

**2、上传(Upload)**

上传模块主要用于框架JS端的上传逻辑，默认采用plupload进行上传，也可以采用Ajax的方法进行上传，plupload的方法上传支持上传进度展示。

**(1)、引入方式**

require(['upload'], function(Upload){});

**(2)、内部方法**

// 为表单中的上传按钮绑定plupload上传事件

Upload.api.plupload(element, onUploadSuccess, onUploadError);

// 上传自定义事件存储对象

Upload.api.custom

// 采用Ajax的方法进行上传文件

Upload.api.send(file, onUploadSuccess, onUploadError);

// plupload上传组件配置

Upload.config.classname

Upload.config.container

Upload.config.previewtpl

// upload上传事件配置

Upload.event.onBeforeUpload(up, file);

Upload.event.onFileAdded(up, files);

Upload.event.onPostInit(up);

Upload.event.onUploadError(ret, onUploadError, button);

Upload.event.onUploadResponse(response);

Upload.event.onUploadSuccess(ret, onUploadSuccess, button);

**(3)、使用示例**

// 使用Plupload上传

Upload.api.plupload($(".plupload"), function(data, ret){

Toastr.success("成功");

}, function(data, ret){

Toastr.success("失败");

});

// 使用Ajax上传

Upload.api.send(file, function(data, ret){

Toastr.success("成功");

}, function(data, ret){

Toastr.success("失败");

});

以上代码表示当表单提交处理成功后提示成功，处理失败提示失败。

--

**3、表格(Table)**

表格模块主要用于渲染数据列表，对数据列表进行排序、过滤、筛选、绑定事件、增删改查等等操作。我们在CRUD一键生成后看到的列表就是使用表格模块进行完成的。

**(1)、引入方式**

require(['table'], function(Table){});

**(2)、内部方法**

// 为渲染的表格绑定上增删改查等事件

Table.api.bindevent(table);

// 生成按钮链接

Table.api.buttonlink(column, buttons, value, row, index, type);

// 表格格式化对象集，包含多个格式化数据的方法

Table.api.formatter

// 根据行获取行数据

Table.api.getrowdata(table, index)

// 初始化表格配置

Table.api.init(defaults, columnDefaults, locales)

// 表格批量操作的方法

Table.api.multi

// 替换URL链接中的{变量名}

Table.api.replaceurl

// 获取选中的ID

Table.api.selectedids

// Bootstrap-table表格默认列的配置

Table.columnDefaults

// 表格相关DOM元素类配置

Table.config

// Bootstrap-table默认config配置

Table.defaults

// 已经渲染的表格对象

Table.list

--

**(3)、使用示例**

// 初始化表格参数配置

Table.api.init({

extend: {

index\_url: 'category/index',

add\_url: 'category/add',

edit\_url: 'category/edit',

del\_url: 'category/del',

multi\_url: 'category/multi',

dragsort\_url: '',

table: 'category',

}

});

var table = $("#table");

// 初始化表格

table.bootstrapTable({

url: $.fn.bootstrapTable.defaults.extend.index\_url,

escape: false,

pk: 'id',

sortName: 'weigh',

pagination: false,

commonSearch: false,

columns: [

[

{checkbox: true},

{field: 'id', title: \_\_('Id')},

{field: 'type', title: \_\_('Type')},

{field: 'name', title: \_\_('Name'), align: 'left'},

{field: 'nickname', title: \_\_('Nickname')},

{field: 'flag', title: \_\_('Flag'), operate: false, formatter: Table.api.formatter.flag},

{field: 'image', title: \_\_('Image'), operate: false, formatter: Table.api.formatter.image},

{field: 'weigh', title: \_\_('Weigh')},

{field: 'status', title: \_\_('Status'), operate: false, formatter: Table.api.formatter.status},

{field: 'operate', title: \_\_('Operate'), table: table, events: Table.api.events.operate, formatter: Table.api.formatter.operate}

]

]

});

//

// 为表格绑定事件

Table.api.bindevent(table);

表格的详细介绍可以查看一张图解表格：<https://forum.fastadmin.net/d/323>。

--

--

[PHP]\_TP5.0\_FastAdmin\_11、组件

(整理 <https://doc.fastadmin.net/docs/component.html>)

--

FastAdmin默认集成了多个第三方组合，如表单验证、文件上传、下拉列表、时间选择、城市选择、Selectpage，所有的组件都必须使用Form.api.bindevent("form[role=form]")来进行初始化，如果不进行初始化是无法对相应组件进行渲染和事件绑定。

//

// 如果我们想单独为元素绑定事件和渲染，我们可以使用如下的代码

组件 代码 介绍

文件上传 Form.events.plupload($("form")); //渲染并绑定form中的上传组件;

动态下拉列表 Form.events.selectpage($("form")); //渲染并绑定form中的Selectpage组件;

表单验证 Form.events.validator($("form")); //渲染并绑定form中的表单验证;

城市选择 Form.events.citypicker($("form")); //渲染并绑定form中的城市选择组件;

日期时间 Form.events.datetimepicker($("form")); //渲染并绑定form中的日期时间组件;

下拉列表 Form.events.selectpicker($("form")); //渲染并绑定form中的Selectpicker组件;

附件选择 Form.events.faselect($("form")); //渲染并绑定form中的选择附件组件;

键值组件 Form.events.fieldlist($("form")); //渲染并绑定form中的键值组件;

开关组件 Form.events.switcher($("form")); //渲染并绑定form中的开关组件;

滑块组件 Form.events.slider($("form")); //渲染并绑定form中的滑块组件;

--

**一、文件上传**

FastAdmin支持将文件或图片直传到第三方云存储服务器而不需要通过服务器进行中转，你可以直接在后台插件管理安装第三方云存储的插件后使用，目前支持以下云储存平台。

平台 特点 插件下载；

又拍云 申请加入联盟可享每月免费15G流量、图片处理 下载(<https://www.fastadmin.net/store/upyun.html>)；

七牛云 实名认证后免费10G流量、稳定、图片处理 下载(<https://www.fastadmin.net/store/qiniu.html>)；

阿里OSS 阿里系、稳定、图片处理、支持挂载为分区 下载(<https://www.fastadmin.net/store/alioss.html>)；

Ucloud 每月20G免费流量、图片处理 下载(<https://www.fastadmin.net/store/ucloud.html>)；

腾讯COS 每月50G免费流量、图片处理 下载(<https://www.fastadmin.net/store/cos.html>)；

//

在使用第三方云存储功能之前请先注册一个账号并新增一个云储存服务,你可以通过FastAdmin的邀请链接注册，FastAdmin会获得相应平台的CDN流量。

又拍云：https://www.fastadmin.net/go/upyun

七牛云：https://www.fastadmin.net/go/qiniu

阿里云：https://www.fastadmin.net/go/aliyun

腾讯云：https://www.fastadmin.net/go/tencent

//

FastAdmin的上传功能非常强大，我们只需要简单的配置即可支持单图或多图上传。

<div class="form-group">

<label for="c-avatar" class="control-label col-xs-12 col-sm-2">头像:</label>

<div class="col-xs-12 col-sm-8">

<!--文件上传-->

<div class="input-group">

<input id="c-avatar" data-rule="" class="form-control" size="50" name="row[avatar]" type="text" value="{$row.avatar}">

<div class="input-group-addon no-border no-padding">

<span><button type="button" id="plupload-avatar" class="btn btn-danger plupload" data-input-id="c-avatar" data-mimetype="image/gif,image/jpeg,image/png,image/jpg,image/bmp" data-multiple="false" data-preview-id="p-avatar"><i class="fa fa-upload"></i> 上传</button></span>

<span><button type="button" id="fachoose-avatar" class="btn btn-primary fachoose" data-input-id="c-avatar" data-mimetype="image/\*" data-multiple="false"><i class="fa fa-list"></i> 选择</button></span>

</div>

<span class="msg-box n-right" for="c-avatar"></span>

</div>

<ul class="row list-inline plupload-preview" id="p-avatar"></ul>

</div>

</div>

//

//// 组件元素：配置一个文本框、一个上传按钮、一个选择按钮、一个预览的DIV。

文本框 主要用于接收上传后返回的图片链接，文本框id属性是必选的，这里的id是c-avatar。

上传按钮 主要用于点击后上传文件，有几个属性非常重要，请参考下方的上传按钮属性介绍。

选择按钮 主要用于点击后选择已经上传的文件，有几个属性非常重要，请参考下方的选择按钮属性介绍。

预览区域 主要用于预览上传或选择文件后的预览，id属性是改造的，这里的id是p-avatar。

//

//// 上传按钮支持属性

属性 示例值 说明。

data-url ajax/upload 用于配置上传文件接收的地址。

data-multipart {"key1":"value1"} 用于上传时附加额外的参数信息。

data-input-id c-avatar 用于填充返回URL地址的设文本框。

data-mimetype image/gif,image/jpeg,image/png,image/jpg,image/bmp 用于过滤允许上传的文件类型，支持mimetype或文件后缀名。

data-multiple false 是否支持多图或多文件模式。

data-preview-id p-avatar 用于预览返回URL地址的DIV。

data-maxsize 10M 用于限制最大可上传的文件大小。

//

//// 选择按钮支持属性(属性 示例值 说明)

data-input-id c-avatar 用于填充返回URL地址的设文本框。

data-mimetype image/gif,image/jpeg,image/png,image/jpg,image/bmp 用于过滤允许上传的文件类型，支持mimetype或文件后缀名。

data-multiple false 是否支持多图或多文件模式。

data-preview-id p-avatar 用于预览返回URL地址的DIV。

//

如果我们想直接通过JS上传一个文件到我们的服务器，我们可以使用Upload.api.send(file, success, failure, complete)来上传文件。

--

**二、动态下拉列表(SelectPage)**

FastAdmin中的动态下拉列表使用的是优秀强大的Selectpage插件来支持，FastAdmin对其进行了二次开发。下面介绍一个最基础的动态下拉列表示例，如下所述。

<input id="c-name" data-rule="required" data-source="category/selectpage" class="form-control selectpage" name="row[name]" type="text" value="">

其中需要给元素class添加一个selectpage，其次需要增加一个data-source="category/selectpage"这个属性，category/selectpage为我们控制器提交列表的方法。FastAdmin的Selectpage列表中显示字段默认读取的是name字段，如果我们返回的列表中不包含name字段，将无法展现下拉列表数据。此时我们需要添加使用data-field="你要显示的字段"即可。FastAdmin的Selectpage列表中主键字段默认读取的是id字段，如果我们的主键不是id字段，则我们可以添加并使用data-primary-key="你的主键ID字段"来修改。

//

//// Selectpage扩展属性(属性 功能 示例)

data-source 提供数据源的URL地址 data-source="category/index"。

data-field 列表显示读取的字段 data-field="username"。

data-primary-key 列表选中后渲染的字段 data-primary-key="uid"。

data-pagination 是否开启分页 data-pagination="true"。

data-page-size 分页大小 data-page-size="10"。

data-multiple 是否支持多选 data-multiple="true"。

data-max-select-limit 最多可选择数量 data-max-select-limit="3"。

data-order-by 排序字段 data-order-by="id"。

data-params 自定义扩展参数 data-params='{"custom[type]":"test"}'。

data-select-only 是否为只读模式 data-select-only="true"。

//

Selectpage的data-params支持function类型,如果需要动态传参(例如联动查询),则可以在JS中将data-params添加一个function处理即可。FastAdmin在生成CRUD时会对包含下划线的字段默认生成动态下拉列表，比如user\_id将自动生成data-source="user/index"，默认读取的是id和name字段，如果需要修改，请参考上方的参数修改方法。

//

data-params自定义扩展参数支持使用function动态返回数据，请在表单初始化之前使用，例如传递动态选择的类型。

$("#c-name").data("params", function (obj) {

return {custom: {type: $("#c-type").val()}};

});

更多的使用方法请参考Selectpage官方教程(<https://terryz.github.io/>)。

--

**三、富文本编辑器**

FastAdmin的富文本编辑器只需要给对应的textarea增加一个class为editor即可，FastAdmin在渲染时即会将textarea渲染为富文本编辑器，目前支持summernote、nkeditor和ueditor等富文本编辑器，需安装对应的插件即可正常使用。

//

//// 插件 特点 插件下载

Summernote 操作简单、易用、图片直传第三方存储 下载(<https://www.fastadmin.net/store/summernote.html>)。

Nkeditor 基于Kindeditor、多主题、简洁强大、图片直传第三方存储 下载(<https://www.fastadmin.net/store/nkeditor.html>)。

Wangeditor 轻量、简洁、易用、图片直传第三方存储 下载(<https://www.fastadmin.net/store/wangeditor.html>)。

Simeditor 漂亮、简单、功能简单、图片直传第三方存储 下载(<https://www.fastadmin.net/store/simeditor.html>)。

Umeditor 百度出品、简单、易用、支持公式、图片直传第三方存储 下载(<https://www.fastadmin.net/store/umeditor.html>)。

Ueditor 百度出品、功能全、图片直传第三方存储 下载(<https://www.fastadmin.net/store/ueditor.html>)。

//

如果是自定义表单,例如在index页面,切记使用Form.api.bindevent绑定事件才会生效,否则单独添加个editor是不会生效的。在安装完对应的富文本插件后切记启用、刷新插件缓存并清除浏览器缓存后才生效。请务必同时只启用一个富文本编辑器，如果需要不同的页面展示不同的编辑器，需要修改编辑器插件代码，修改对应的选择器，将editor改成自定义的class。

--

**四、表单验证**

FastAdmin的表单验证采用的是Nice-validator验证插件，Nice-validator是一款非常强大的表单验证插件，通过简单在元素上配置规则，即可达到验证的效果。在FastAdmin当中我们只需要给元素添加data-rule="规则"即可开启Nice-validator的验证，常用的规则如下所述。

//

//// 规则 描述 示例

required 字段必填 required。

checked 必选，只适用于checkbox和radio checked。

match(name) 当前字段值必须和 name 字段的值匹配 match('row[username]')。

remote(URL) 请求服务端验证 remote('validate/check\_username\_unique')。

integer 整数 integer。

range(n~) 数值范围, 请填写不小于 n 的数 range(3~)。

length(n) 请填写 n 个字符 length(3)。

filter 过滤 <>`"' 和字符实体编码的字符 filter。

digits 必须为数字 digits。

letters 必须为字母 letters。

date 必须为日期,yyyy-mm-dd格式 date。

time 必须为时间,hh:ii格式 time。

email 必须为email格式 email。

url 必须为URL链接 url。

qq 必须为QQ号 qq。

IDcard 必须为身份证号码 IDcard。

tel 必须为电话号码 tel。

mobile 必须为手机号码 mobile。

zipcode 必须为邮箱 zipcode。

chinese 必须为中文字符 chinese。

username 必须为3-12位数字、字母、下划线 username。

password 必须为6-16位字符，不能有空格 password。

//

如果你的表单不希望使用nice-validator组件进行验证，可以给form添加一个novalidate属性即可。更多的使用方法请参考Nice-validator官方教程(<https://validator.niceue.com/docs/>)。

--

**五、城市选择**

FastAdmin中集成了强大的city-picker城市选择插件，可以很方便的选择省份和城市。我们只需要简单的为input元素添加一个data-toggle="city-picker"属性即可自动渲染相应的城市选择组件

//

//// 城市选择属性(属性 描述 示例)

data-level 选择城市的级别，支持province/city/district，默认为district data-level="city"。

data-simple 使用简单的地址，比如使用内蒙古替代内蒙古自治区，默认为false data-simple="true"。

data-responsive 是否为自适应，默认为false data-responsive="true"。

placeholder 默认提示的文字 placeholder="请选择省/市"。

//

city-picker组件默认选择后渲染的是中文城市信息，我们可以通过在JS中监听city-picker更新后的事件来获取省份城市地区对应的code值。代码如下所述。

$("#city-picker").on("cp:updated", function() {

var citypicker = $(this).data("citypicker");

var code = citypicker.getCode("district") || citypicker.getCode("city") || citypicker.getCode("province");

$("#code").val(code);

});

如果我们数据库中存放的是地区的code值，在显示时我们则需要把对应的code通过读取数据库转换成我们的地区中文，然后再设定input的value值即可。更多的使用方法请参考city-picker官方教程(<https://github.com/tshi0912/city-picker>)。

--

**六、日期时间**

在FastAdmin中的日期时间组件采用的是Bootstrap-datetimepicker插件。我们在使用只可以为文本框添加一个class为datetimepicker的值即可自动添加日期时间选择框。同时我们还可以通过配置以下属性来自定义我们日期选择器的功能。

//

//// 属性 描述 示例

data-date-format 日期时间的格式，支持Moment.js的格式 data-date-format=" YYYY-MM-DD"。

data-date-min-date 最小可选择的日期 data-date-min-date="2011-10-01"。

data-date-max-date 最大可选择的日期 data-date-max-date="2046-10-01"。

data-date-use-current 使用当前的日期时间 data-date-use-current="true"。

data-date-default-date 默认日期 data-date-default-date="2011-10-01"。

data-date-disabled-dates 禁用的日期组 data-date-disabled-dates='["2011-10-02"]'。

data-date-enabled-dates 可用的日期组 data-date-enabled-dates='["2011-10-02"]'。

data-date-use-strict 使用严格的日期时间,错误日期将被忽略 data-date-use-strict="true"。

data-date-side-by-side 日期时间并排显示 data-date-side-by-side="true"。

//

更多的使用方法请参考Bootstrap-datetimepicker官方教程(<https://eonasdan.github.io/bootstrap-datetimepicker/>)。

--

**七、下拉列表**

在FastAdmin中集成了Bootstrap-select插件，可以对原有的select元素重新渲染，并增加相应的功能。我们可以直接给select元素添加一个class为selectpicker的值即可，FastAdmin在检测到以后会自动进行渲染，我们同时可以给select添加以下属性用于配置selectpicker。

//

//// 属性 介绍 添加位置 示例

data-live-search 是否启用动态搜索 select data-live-search="true"。

data-tokens 添加搜索的关键字 option data-tokens="keyword keyword2"。

data-max-options 最大可选择option的数量 select\optgroup data-max-options="2"。

title 自定义默认提示语 select title="请选择相应的分类"。

title 自定义选中以后显示的文字 option title="分类1"。

data-style 定义样式 select data-style="btn-primary"。

//

更多的使用方法请参考：Selectpicker官方教程(<https://silviomoreto.github.io/bootstrap-select/>)。

--

**八、键值组件**

键值组件是FastAdmin开发的一项简洁实用的基础组件，在FastAdmin中很多模块都有使用到该组件，例如常规管理->系统配置->字典配置均使用此组件开发，我们在插件管理配置中也经常可以看到键值组件的身影。

//

// 键值组件最常用使用方法

<dl class="fieldlist" data-name="row[configgroup]">

<dd>

<ins>键名</ins>

<ins>键值</ins>

</dd>

<dd>

<a href="javascript:;" class="btn btn-sm btn-success btn-append"><i class="fa fa-plus"></i> 追加</a>

</dd>

<textarea name="row[configgroup]" class="form-control hide" cols="30" rows="5">{"basic":"基础配置","email":"邮件配置","dictionary":"字典配置","user":"会员配置","example":"示例分组"}</textarea>

</dl>

通过将以上代码放置在我们的表单中，然后使用Form.api.bindevent("form")或Form.events.fieldlist("form")进行初始化即可。

//

以上是最简洁的使用方法，fieldlist还有更强大的自定义功能，如下所述。

<dl class="fieldlist" data-name="row[test]" data-template="testtpl">

<dd>

<ins>姓名</ins>

<ins>性别</ins>

<ins>年龄</ins>

<ins>成绩</ins>

</dd>

<dd>

<a href="javascript:;" class="btn btn-sm btn-success btn-append"><i class="fa fa-plus"></i> 追加</a>

</dd>

<textarea name="row[test]" class="form-control hide" cols="30" rows="5">[{"name":"张三"},{"gender":"男"},{"age":"23"},{"score":"80"}]</textarea>

</dl>

<!--定义模板-->

<script type="text/javascript" id="testtpl">

<dd class="form-inline">

<input type="text" name="row[<%=name%>][<%=index%>][name]" class="form-control" value="<%=row['name']%>" size="10">

<input type="text" name="row[<%=name%>][<%=index%>][gender]" class="form-control" value="<%=row['gender']%>" size="30">

<input type="text" name="row[<%=name%>][<%=index%>][age]" class="form-control" value="<%=row['age']%>" size="30">

<input type="text" name="row[<%=name%>][<%=index%>][score]" class="form-control" value="<%=row['score']%>" size="30">

<span class="btn btn-sm btn-danger btn-remove"><i class="fa fa-times"></i></span>

<span class="btn btn-sm btn-primary btn-dragsort"><i class="fa fa-arrows"></i></span>

</dd>

</script>

通过以上定义，可以任意自定义我们展示项的数据。

//

如果我们需要在点击追加按钮以后再对新增的展示项绑定事件，我们可以在JS中通过监听事件来给新增的元素绑定事件。

$(document).on("fa.event.appendfieldlist", 'data-name="row[test]"', function(){

Form.api.bindevent(this);

});

--

**九、开关组件**

开关组件常用于状态值的变更或只有两个值的切换。使用开关组件只需要给我们的操作按钮添加data-toggle="switcher"即可，如下所述。

<input id="c-switch" name="row[switch]" type="hidden" value="0">

<a href="javascript:;" data-toggle="switcher" class="btn-switcher" data-input-id="c-switch" data-yes="1" data-no="0" >

<i class="fa fa-toggle-on text-success {eq name="$row.switch" value="0"}fa-flip-horizontal text-gray{/eq} fa-2x"></i>

</a>

--

--

[PHP]\_TP5.0\_FastAdmin\_12、常见问题

(整理 <https://doc.fastadmin.net/docs/faq.html>)

--

如果你在使用FastAdmin的过程中发现任何问题,请到问答社区发贴: <https://ask.fastadmin.net>。

--

**一、在FastAdmin中如何开启调试模式？**

开启调试模式的方法有两种：

1、打开application/config.php，找到app\_debug,将它的值置为true即可。

2、如果你有启用.env环境配置，修改其中app\_debug的值为true即可。

--

**二、为什么在调试模式下功能正常，但在生产环境下功能失效？**

如果你有修改了框架核心的JS文件或者修改了核心样式文件，需要使用命令重新压缩打包JS和CSS后才会在生产环境下生效，具体请参考：<https://doc.fastadmin.net/docs/command.html#>一键压缩打包-3。

--

**三、为什么在外网访问后台速度非常慢？**

如果你在外网开启了调试模式，因为在调试模式下加载的文件非常多，访问速度会非常慢，请在外网或生产环境下关闭调试模式。如果有更高的响应要求，建议采用CDN部署静态资源。部署方式请参考：#如何将静态资源采用CDN方式部署到第三方云存储-13( <https://doc.fastadmin.net/docs/faq.html#> )。

--

**四、如何启用后台管理多级菜单功能？**

FastAdmin从1.0.0.20180513\_beta版本开始新增了多级菜单功能，开发者可以很方便的在配置文件中修改是否开启多级菜单功能，找到application/config.php文件最下方有个multiplenav配置，默认是false，如果需要启用，请将multiplenav置为true即可。如果启用了多级菜单后，菜单规则中的第一级将作为一级菜单显示在顶部。我们可以在权限管理->菜单规则中额外添加一级菜单，然后再重新规划我们的菜单。

--

**五、为什么我下载的FastAdmin没有头像首字母功能？**

头像首字母功能从1.0.0.20190410\_beta才新加的功能，如果你使用旧版本是没有此功能的。

--

**六、插件管理配置了错误的伪静态导致后台无法登录**

如果在后台配置某一插件的伪静态错误时，导致后台完全无法打开时，你可以尝试使用<http://www.yoursite.com/admin.php> 进行登录后台，登录后再重新配置伪静态即可。

--

**七、如何启用后台登录验证码功能**

FastAdmin中的后台登录验证功能默认是关闭的，如果需要启用请修改application/config.php文件最下方的login\_captcha，修改为true即可。

--

**八、如何在生成CRUD时生成回收站功能**

FastAdmin从1.0.0.20190301\_beta版本开始已经支持自动生成回收站功能，如果需要生成回收站功能，请确保你的FastAdmin版本，其次确保你的数据表中存在deletetime这个字段，只需保证存在此字段，CRUD在生成时会自动生成回收站相关的前后端代码和文件。

--

**九、如何在控制器或模型中获取当前登录的管理员或登录用户信息**

在FastAdmin所提供的基类Frontend、Backend、Api中都有提供获取当前管理员或登录用户的信息方法。在后台管理的控制器中可以通过$this->auth->id来获取管理员ID，$this->auth->字段名获取管理员的其它信息，如果需要在后台的Model中获取当前登录的管理员ID，可以通过两种方式获取。

// 第一种，使用Session来获取管理员信息

\think\Session::get('admin');

//

// 第二种，使用单例方法来获取

$auth = \app\admin\library\Auth::instance();

//

在前台或API的控制器中可以通过$this->auth->id来获取当前登录会员的ID，$this->auth->字段员获取登录会员的其它信息，如果需要在前台的Model中获取当前登录的用户信息，可以能和以下的方式获取。

// 获取Auth对象

$auth = \app\common\library\Auth::instance();

// 获取会员模型

$user = $auth->getUser();

--

**十、安装后提示控制器不存在:E或控制器不存在:N**

出现这种情况一般是由于Web服务器的PATH\_INFO未配置正确，导致服务器接收到了错误的PATH\_INFO值，请检查你的PATH\_INFO并修复后再重试。

--

**十一、FastAdmin的数据库SQL文件在哪里**

FastAdmin在安装时会自动创建数据库和数据表,免除了你手动创建数据库和导入数据库的烦恼。但很多时候我们需要构造自己的安装SQL，这就需要修改安装SQL文件。FastAdmin的数据库安装文件保存在 application/admin/command/Install/fastadmin.sql( <https://gitee.com/karson/fastadmin/raw/master/application/admin/command/Install/fastadmin.sql> )。

--

**十二、如何修改后台默认皮肤**

为了进一步提升加载速度，后台默认启用了绿色主题的皮肤，如何修改其它皮肤呢？

找到/public/assets/css/backend.css 这个文件，默认是如下所述。

@import url("../css/bootstrap.css");

@import url("../css/fastadmin.css");

@import url("../css/skins/skin-green.css");

@import url("../css/iconfont.css");

@import url("../libs/font-awesome/css/font-awesome.min.css");

@import url("../libs/toastr/toastr.min.css");

@import url("../libs/fastadmin-layer/dist/theme/default/layer.css");

@import url("../libs/bootstrap-table/dist/bootstrap-table.min.css");

@import url("../libs/eonasdan-bootstrap-datetimepicker/build/css/bootstrap-datetimepicker.min.css");

@import url("../libs/bootstrap-daterangepicker/daterangepicker.css");

@import url("../libs/nice-validator/dist/jquery.validator.css");

@import url("../libs/bootstrap-select/dist/css/bootstrap-select.min.css");

@import url("../libs/fastadmin-selectpage/selectpage.css");

//

其中可以看到只加载了skin-green.css这个皮肤，如果需要启用其它皮肤可以在文件末尾追加“@import url("../css/skins/skin-颜色标识.css");”。颜色标识总共有 black/black-light/blue/blue-light/green/green-light/purple/purple-light/red/red-light/yellow/yellow-light总12个颜色标识。如果需要一次性加载全部的皮肤样式，则把skin-green改为\_all-skins即可。最后修改application/admin/views/index/index.html中第6行，将skin-green换成你需要的颜色标识。注意生产环境还需要执行php think min -m all -r all才会生效。

--

**十三、为什么前台自己创建的控制器提示你没有权限访问**

这是由于控制器的基类你肯定继承的是Frontend，Frontend基类有鉴权判断，有以下两种办法可以尝试。添加控制器的属性$noNeedRight=["\*"]，表示无需再做鉴权。后台会员规则中添加相应的规则，再给对应的会员组赋予相应的权限即可。

--

**十四、为什么后台给管理员所在组分配了权限，管理员仍然提示没有权限访问**

如果是你自己添加的控制器，可以使用一键生成菜单命令(php think menu -c 控制器名)来生成菜单，如果你是手动添加的规则菜单，权限规则必须细化到控制器的方法才可以。

--

**十五、php think install报不是内部或外部命令**

这是由于php.exe文件所在目录未加入到PATH环境变量导致的。找到php.exe文件所在的目录，将该目录加入到系统PATH环境变量中后，重启CMD即可解决。

--

**十六、php think install报command not found**

这是由于在Linux环境下未找到php的脚本程序，有两种解决办法，首先尝试使用which php找到php所在的位置。找到php脚本程序所在的目录，加入到PATH环境变量中去，使用export PATH=$PATH:php脚本程序所在目录。找到php脚本程序文件，使用ln -s php脚本程序文件 /usr/bin/php。

--

**十七、安装后访问后台和会员中心都只显示首页**

这是由于伪静态没有生效或错误导致的。请参考下方的Apache和Nginx伪静态配置。

--

**十八、安装后只能访问首页，其它页均报no input file specified**

这是由于伪静态没有生效或错误导致的。这种情况一般在Apache下伪静态不工作的情况下出现。首先确保已经启用Apache的伪静态，确保目录已经配置好权限，如下面的Directory配置。

// Apache

<VirtualHost \*:80>

DocumentRoot "/Users/Karson/Project/fastadmin/public"

ServerName fa.com

ServerAlias fa.com \*.fa.com

<Directory "/Users/Karson/Project/fastadmin">

AllowOverride All

Options Indexes FollowSymLinks

Require all granted

</Directory>

</VirtualHost>

其次伪静态规则在Apache fastcgi模式下会导致No input file specified。请修改public目录下的.htaccess文件。默认的“RewriteRule ^(.\*)$ index.php/$1 [QSA,PT,L]”

修改成“RewriteRule ^(.\*)$ index.php [L,E=PATH\_INFO:$1]”。

--

**十九、安装后只能访问首页，其它页均报404 page not found**

这是由于伪静态未配置或没有生效导致的。这种情况一般在Nginx下未配置伪静态的情况下出现，建议将虚拟主机的root绑定至public目录。

// Nginx

server {

listen 80;

server\_name [www.fa.com](http://www.fa.com) \*.fa.com;

root "C:/phpstudy/WWW/fastadmin/public";

location / {

index index.html index.htm index.php;

#主要是这一段一定要确保存在

if (!-e $request\_filename) {

rewrite ^(.\*)$ /index.php?s=/$1 last;

break;

}

#结束

#autoindex on;

}

location ~ \.php(.\*)$ {

fastcgi\_pass 127.0.0.1:9000;

fastcgi\_index index.php;

fastcgi\_split\_path\_info ^((?U).+\.php)(/?.+)$;

fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;

fastcgi\_param PATH\_INFO $fastcgi\_path\_info;

fastcgi\_param PATH\_TRANSLATED $document\_root$fastcgi\_path\_info;

include fastcgi\_params;

}

}

请将C:/phpstudy/WWW改成你服务器对应所在的目录。如果你使用的是 lnmp.org 的一键安装LNMP环境，请查阅 <https://lnmp.org/faq/lnmp-vhost-add-howto.html#rewrite> 的伪静态配置。

--

**二十、composer install时无法下载package**

这是由于composer默认配置是国外的源，如遇网络故障则会导致无法下载。执行命令前先开启代理，使用国内的镜像源，有以下两种启用方法。

//// 方法一： 修改 composer 的全局配置文件（推荐方式）

// 打开命令行窗口并执行如下命令

composer config -g repo.packagist composer https://packagist.laravel-china.org

// 方法二： 修改当前项目的 composer.json 配置文件

打开命令行窗口，进入你的项目的根目录（也就是 composer.json 文件所在目录），执行如下命令。

composer config repo.packagist composer https://packagist.laravel-china.org

感谢：<https://packagist.laravel-china.org>。

--

**二十一、如何修改或禁用左侧菜单栏的角标**

FastAdmin后台左侧菜单栏有彩色的小角标，这一般用于通知和提醒操作，在后台开发时是非常方便的一个小功能，如何修改和禁用它呢？

找到/application/admin/controller/Index.php中的index方法，其中有一段如下代码。

$menulist = $this->auth->getSidebar([

'dashboard' => 'hot',

'auth' => ['new', 'red', 'badge'],

'auth/admin' => 12,

'auth/rule' => 4,

'general' => ['18', 'purple'],

]);

数组的键名是对应的左侧菜单栏的相对链接，数组的键值是需要显示的文字或数字，可以传字符串或数组。如果是字符串，则角标的颜色是按照'red', 'green', 'yellow', 'blue', 'teal', 'orange', 'purple'的方式进行循环的。如果是数组，这三个值分别表示：[显示的文字, 颜色，展现方式(badge或label)]。如果需要删除这个小角标，则可以直接到数组置为空即可。在这里仅仅是PHP端操作小角标的方式，在JS端同样可以进行相应的操作。

// 在你的模块中可以调用

top.window.Backend.api.sidebar({

'auth/admin': 44

});

// 具体使用方法同PHP端相同。如何动态的在JS中移除一个小角标呢，采用以下的方法即可。

top.window.Backend.api.sidebar({

'auth/admin': 0

});

--

**二十二、后台上传文件时提示HTTP Error.(code:-200)错误**

在后台上传文件时如果遇到-200错误，首先请开启调试模式，然后请使用谷歌浏览器的开发者工具，按F12，切换到Network，看具体的错误信息。

--

**二十三、在Windows下如何压缩打包JS和CSS**

在FastAdmin中压缩打包JS和CSS文件需要NodeJS的支持。在Windows下需要手动配置Node的可执行文件,请修改application/admin/command/Min.php中$nodeExec的值。

如你的Node可执行文件是C:/Program Files/nodejs/node.exe，则请配置$nodeExec = '"C:/Program Files/nodejs/node.exe"'。

--

**二十四、提示你所浏览的页面暂时无法访问**

如果我们在FastAdmin开发过程中遇到此错误，说明我们application/config.php中的app\_debug是关闭的，必须开启app\_debug为true才可以显示出详细的错误信息。如果开启app\_debug仍然显示不出详细错误，请确保php.ini中的display\_error为开启状态。

--

**二十五、提示未知的数据格式或网络错误**

很多时候都有可能遇到提示未知的数据格式或网络错误这个提示，产生这个错误的原因一般来说都是服务端报错，导致返回的数据不是JSON格式或直接未返回。

准备工作：首先确保你的FastAdmin开启了调试模式application/config.php中的app\_debug置为true。两种定位错误的方法：使用Chrome浏览器，打开开发者工具，选中Network(网络)选项卡,刷新一下页面或重新请求一次，定位到我们请求的URL，点击然后在Preview即可看到错误信息；直接查看runtime/log目录下的错误日志。修复错误后再重试即可。FastAdmin建议运行在PHP5.5及以上版本，因此如果提示网络错误请检查你的PHP是否低于该版本。

--

**二十六、安装时提示无法写入application/database.php**

造成此问题的原因通常有几下两种情况。

1、application/database.php文件没有写入权限，请先查看你的apache或nginx的运行用户，然后修改文件的所有者，再添加上写入的权限，通常情况下需要执行以下命令。

cd /yourpath/fastadmin/

chown www-data:www-data ./ -R

chmod a+w ./application/database.php

chmod a+w ./runtime/ -R

2、其次就是可能你的目录绑定到public后，open\_basedir限制了访问父目录，可以配置下open\_basedir，添加上级目录即可，通常情况下Linux需要在配置文件中追加的代码如下所述。

fastcgi\_param PHP\_ADMIN\_VALUE "open\_basedir=/yourpath/fastadmin/:/tmp/:/proc/";

如果使用的是lnmp一键安装包，可能需要修改.user.ini文件，在.user.ini文件中添加“open\_basedir=/yourpath/fastadmin/:/tmp/:/proc/”。

3、如果使用的是宝塔面板，请直接在网站->配置->网站目录中修改运行目录为public目录。

--

**二十七、如何将静态资源采用CDN方式部署到第三方云存储**

FastAdmin可以将静态的资源部署到又拍云、七牛云或阿里OSS，可大大的加快网站的访问。默认FastAdmin的静态资源是不采用CDN部署的，如果需要启用，需要修改以下两个文件的配置。

修改application/extra/site.php中cdnurl的值为你CDN的地址，修改application/config.php中“\_\_CDN\_\_”的值为你CDN的地址，请将你的静态资源public/assets文件夹上传至你的CDN空间中去。

如果采用了静态资源部署CDN，在后台插件管理中对插件执行安装、禁用、启用、卸载后都需要将public/assets/addons/目录和public/assets/js/addons.js文件再次同步更新到CDN。如果你需要将上传的文件直传至又拍云、七牛云或阿里OSS，请在插件管理中下载对应第三方云存储的插件并配置好即可。

--

--

[PHP]\_TP5.0\_FastAdmin\_13、贡献代码

(整理 <https://doc.fastadmin.net/docs/contributing.html>)

--

**一、开发**

我们欢迎广大开发者贡献大家的智慧，让FastAdmin变得更完美。

**1、开始之前**

请严格遵循以下代码标准：PSR-2( <https://github.com/php-fig/fig-standards/blob/master/accepted/PSR-2-coding-style-guide.md> )；使用4个空格作为缩进；

**2、操作流程**

首先Fork karsonzhang/fastadmin(Github) 或 karson/fastadmin(码云)，然后Clone 到你本地：git clone <https://github.com/><yourname>/fastadmin.git

或 git clone <https://gitee.com/><yourname>/fastadmin.git。编写代码并提交，Push到你的分支。创建 Pull Request 并描述你完成的功能或者做出的修改。

--

**二、更新文档**

我们的文档也是开源的，源代码托管在 karsonzhang/fastadmin-doc。

**1、操作流程**

首页Fork karsonzhang/fastadmin-doc。然后Clone到你的本地：git clone <https://github.com/><yourname>/fastadmin-doc.git。编写代码并提交。Push 到你的仓库。创建 Pull Request 并描述你完成的功能或者做出的修改。

--

**三、报告BUG**

当你在使用过程中遇到问题，请查阅 常见问题( <https://doc.fastadmin.net/docs/faq.html> ) 或者在 FastAdmin交流社区( <https://forum.fastadmin.net/> ) 提交你的问题。

--

--