[PHP]\_TP5.0\_FastAdmin\_前端；

--王尔贝(我是害虫)

(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>。