**Vtor**

1. **概述**

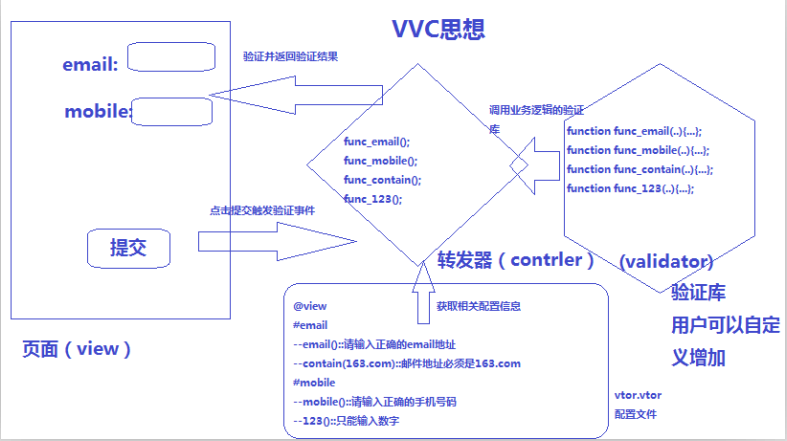
vtor 是一个全新的解耦合页面验证js框架。根据全新的VVC思想(validator-view-controler). 基于配置，使验证与页面分离，更加方便、高效。

Vtor 验证库包含了25种以上的常用验证，email,url,phone,mobile,ajax等等。

用户可以自定义添加或覆盖已有的验证函数库；

用户可以自定义配置文件组织格式。

VVC的工作图：

****

1. **用法**
   1. **引入框架**

<script src='js/jquery.js'> </script>

<script src='../jquery.vtor.js'> </script>

* 1. **给需要做验证的DOM元素添加VID属性**

如：demo.html中，对email验证

电子邮箱地址：<input type='text' vid='iemail' value='' />

**2.3配置验证部件**

如：在验证文件demo.vtor中,配置对电子邮件进行格式验证且必须是163邮箱的

@demo

#iemail

--email()::邮箱格式不正确

--contain(163.com)::请填入163邮箱地址

验证文件格式说明：

注释方式：

多行注释/\* \*/

单行注释://

@view //页面模块名以@开头；可以有多个

#DOM VID //页面上DOM VID以#开头；可以有多个

--functionName([param])::message //验证函数类型，参数，提示

/\*

验证语--开头；

::后表示验证错误提示信息；无参数时()可省略不写；

可以复合验证；既一个DOM可以有多个验证语句；

可以有多个。\*/

\*/

**2.4执行验证**

<script>

$(function(){

$vtor.$({ //初始化；参数1：验证文件路径；参数2：页面模块名

path:'vtor.vtor',//验证文件路径

view:'model1',//使用的页面模块名

auto:true,//是否自动验证

autoFunc:'change'//自动验证的函数名（取自jquery bind里的第一个参数,默认keyup）

});

$("#submitButton").click(function(){

var r=$vtor.$v();//执行验证

if(r){//r是返回的验证结果

$( ‘form’).submit()

}

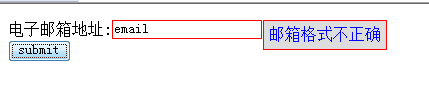
});

})

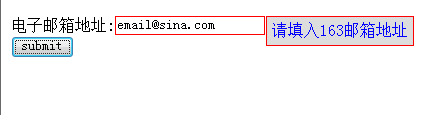
</script>

**2.5验证结果**

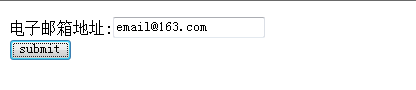
邮箱格式不正确：

****

不是163邮箱：

****

全正确**：**

****

1. **用户自定义配置**

**3.1概述**

用户可以通过在vconf/vtor.config.js文件中自定义修改验证文件组织格式、扩充或覆盖验证函数库。

相关参数配置项如下表：

|  |  |  |
| --- | --- | --- |
| **参数名** | **参数说明** | **默认** |
| func\_pre | 验证函数前缀 | func |
| view\_pre | 页面模块前缀 | @ |
| id\_pre | DOM ID前缀 | # |
| validateTerm\_pre | 验证语句前缀 | -- |
| msg\_pre | 验证提示信息前缀 | :: |
| param\_begin | 验证函数参数开始字符 | ( |
| param\_end | 验证函数参数结束字符 | ) |
| param\_split | 验证函数参数分隔字符 | , |
| msg\_type | 验证结果信息提示类型:single只提示最靠前的一个错误信息；multiple提示全部错误信息 | single |
| errMsg\_class | 提示框错误的样式 | vtor-input-err-msg |
| errInput\_class | 输入框错误的样式 | vtor-input-err-border |
| okMsg\_class | 提示框正确的样式 | vtor-input-ok-msg |
| okInput\_class | 输入框正确的样式 | vtor-input-ok-border |
| vtor\_suf | 验证文件扩展名配置 | Vtor |
| configMethod | 获取vtor配置文件的方式；一般有两种方式：get,post; | post |
| show | 消息框显示的动作:function(vid,msg,result){//这里写你的自定义代码；vid是输入框的jQuery对象，msg是错误提示框的jQuery对象;result表示验证结果} | function(vid,msg,result){msg.show(500);} |
| bind | 初始化验证组件完毕时执行的函数:function(vid,msg){//这里写你的自定义代码；vid是输入框的jQuery对象，msg是错误提示框的jQuery对象;} | function(vid,msg){} |

\*以上参数均区分大小写

需要引入js

<script src='../vconf/vtor.config.js'> </script>

**3.2验证文件组织格式**

自定义验证文件的代码：

custom.vtor={

view\_pre:'#?'//配置文件中验证模块用@?标识

id\_pre:'#?'//配置文件中VID用#?标识

validateTerm\_pre:’-?’//配置文件中验证语句用-?标识

};

配置文件就可以这样配置：

@?view //页面模块名以@?开头；可以有多个

#?domvId //页面上DOM VID以#?开头；

-?functionName([param])::message //验证语句以-?开头

**3.3扩充或覆盖验证函数库**

用户自定义函数的代码：

custom.funcs={

//只能输入数字（无参数）

func\_numeric:function(id){

var val=$vtor.$id(id);//获取dom vid的输入值

return /^\d+$/.test(val);},

//只能输入限定位数的数字（有参数）

func\_numericLimit:function(id,param){

var val=$vtor.$id(id);//获取dom vid的输入值

if(val.length!=param[0]){return false;}

return /^\d+$/.test(val);

}

}

定义好以上两个函数，用户就可以在vtor配置文件中配置使用了

例：

@demo

#inum

-- numeric()::请输入数字//无参也可以：-- numeric::请输入数字

-- numericLimit(4)::请输入4位数字

函数格式= <func\_pre><\_><functionName>:function(<vid>[,param]){[functionBody]};

即：

函数格式=验证函数前缀[默认func]+下划线+函数功能名称:function(DOM VID[,参数]){[方法体]}

**3.3消息样式**

自定义样式有两种方式：

1. 直接修改css/vtor.css文件中的两个样式的内容
2. 在vconf/vtor.config.js中配置相关css类名

第一种就不说了。我们说第二种：

**首先**自己写好样式：demo.css

eb是错误框的样式，em是错误消息的样式

.eb{border:1px solid #555}

.em{border:1px solid #555;background:#ddd;padding:5px;margin-left:5px;

color:blue;max-width:180px;min-width:50px;

}

**然后**在config.js中配置相关信息

custom.vtor={

errMsg\_class:'em',//配置错误消息样式

errInput\_class:'eb'//配置错误框样式

}

custom.vtor={

msg\_class:'em',//配置错误消息样式

errInput\_class:'eb'//配置错误框样式

}

**最后**在html中引入css样式文件（也可以直接写在style块中。总之能加载到两个css类就可以）

<link rel='stylesheet' href='demo.css'/>

**3.3消息事件**

比如：现在我需要消息出现与消失的效果是淡入淡出

可以这样配置：

custom.vtor={

show:function(vid,msg,result){

msg.fadeIn(500);//淡入

} ,

bind:function(vid,msg){

vid.focus(function(){

msg.fadeOut(500);//淡出

})

} }