BootStrap常用插件使用

# datetimepicker的使用

|  |
| --- |
| #1引入依赖  <!—css依赖 -->  <link rel="stylesheet" href="$rc.contextPath/bootstrap/css/bootstrap. min.css"><link rel="stylesheet" href="$rc.contextPath/assets/css/bootstrap-datetimepicker.min.css">  <--javascript-->  <script src="$rc.contextPath/plugins/jQuery/jquery-2.2.3.min.js"> </script>  <script src="$rc.contextPath/bootstrap/js/bootstrap.min.js"> </script>  <script src="$rc.contextPath/assets/js/bootstrap-datetimepicker.min .js"></script>  <script src="$rc.contextPath/assets/js/bootstrap-datetimepicker.zh-CN.js"></script> |
| #2使用方式  $(**'#birthdayInput'**).datetimepicker({  **format**: **'yyyy-mm-dd'**,//日期的格式  **startDate**:**'1900-01-01'**,//选择器的开始日期  **autoclose**:**true**,//日期选择完成后是否关闭选择框  **bootcssVer**:3,//显示向左向右的箭头  **language**:**'zh\_CN'**,//语言  **minView**: **"month"**,//表示日期选择的最小范围，默认是hour  }); |
| #3常用配置说明  1\_format：String. 默认值: 'mm/dd/yyyy'（日期格式， p, P, h, hh, i, ii, s, ss, d, dd, m, mm, M, MM, yy, yyyy 的任意组合。）  2\_startDate： Date. 默认值：开始时间  3\_endDate：Date. 默认值：结束时间  4\_autoclose：Boolean. 默认值：false（当选择一个日期之后是否立即关闭此日期时间选择器。）  5\_startView：Number, String. 默认值：2, 'month'（日期时间选择器打开之后首先显示的视图。 可接受的值：0-4,hour,day,month,year,decade）  6\_minView：Number, String. 默认值：0, 'hour'（日期时间选择器所能够提供的最精确的时间选择视图。）  7\_maxView：Number, String. 默认值：4, 'decade'日期时间选择器最高能展示的选择范围视图。  8\_todayBtn：Boolean, "linked". 默认值: false（）如果此值为true 或 "linked"，则在日期时间选择器组件的底部显示一个 "Today" 按钮用以选择当前日期。如果是true的话，"Today" 按钮仅仅将视图转到当天的日期，如果是"linked"，当天日期将会被选中。）  9\_language：String. 默认值: 'en'。中文zh\_CN  10\_minuteStep：Number. 默认值: 5(此数值被当做步进值用于构建小时视图。对于每个 minuteStep 都会生成一组预设时间（分钟）用于选择。)  11\_showClear: true : 启用删除按钮 |
| /事件change,chageDate,chageYear,…  $('#date-end').datetimepicker().on('changeDate', function(ev){  if (ev.date.valueOf() < date-start-display.valueOf()){  ....  }  }); |

# datatable的使用

|  |
| --- |
| #1引入依赖(jQuery,bootstrap以及以下)  <link rel="stylesheet" href="$rc.contextPath/assets/css/bootstrap-table.min.css">  <script src="$rc.contextPath/assets/js/bootstrap-table.min.js"> </script>  <script src="$rc.contextPath/assets/js/bootstrap-table-zh-CN.min.js" ></script> |
| #2基本使用  方式一：是通过table 的data-url属性指定数据源。  <table data-toggle="table" **data-url="getDatatable"**  data-row-style="rowStyle" data-click-to-select="true"  data-search="true" data-show-refresh="true"  data-show-toggle="true" data-show-columns="true">  <thead>  <tr>  <th data-field="state" data-checkbox="true">></th>  <th class="col-xs-3" **data-align="center"** data-field="name">Name</th>  <th class="col-xs-1" data-align="center" data-field="stars">Stars</th>  <th class="col-xs-1" data-align="center" data-field="forks">Forks</th>  <th class="col-xs-7" data-align="left" data-field="description">Description</th>  </tr>  </thead>  </table>  方式二：通过JavaScript初始化表格时指定url来获取数据  $('#tableList').bootstrapTable({  url : 'list', // 服务器数据的加载地址  method : 'post',  columns : [ {  title : '序号',  align : 'center',  valign:'middle',  formatter: function (value, row, index) {  return index+1;  },    }, {  field : 'username',  title : '用户名',  align : 'center',  valign:'middle',  sortable : true,  cellStyle :function(value, row, index){  var specialCss ={"color": "red"};  if(row.username==="18868801131"){  return {  css : specialCss  };  }  return "";    },    }, {  field : 'gmtCreate',  title : '开始时间',  align : 'center',  valign:'middle',  sortable: true,  }, {  title : '操作',  align : 'left',  valign:'middle',  width : '20%',  formatter: function (value, row, index) {  var viewOporator = "<a href='#' class='btn btn-primary btn-sm view'><span class='fa fa-eye'></span>查看</a> ";  var updateOperator = "<a href='#' class='btn btn-info btn-sm edit'><span class='fa fa-edit'></span>修改</a> ";  var removeOperator = "<a href='#' class='btn btn-warning btn-sm cancel'><span class='fa fa-trash'></span>删除</a> "  var htmlStr = viewOporator + updateOperator +removeOperator;  return htmlStr;  },  events : {  'click .edit' : function(e,value,row,index){    },  'click .cancel' : function(e,value,row,index){  }  },  } ],// 页面需要展示的列，后端交互对象的属性  cache : false, // 是否使用缓存  sortable: true,  pagination : true, // 开启分页  sidePagination : 'server',// 服务器端分页  pageNumber : 1,// 默认加载页  pageSize : 10,// 每页数据  pageList : [ 10, 20, 50, 100 ],// 可选的每页数据  queryParams : function(params) {  return {  limit : params.limit,  offset : params.offset,  sortName : params.sort || "id",  sortOrder: params.order || "desc",  }  }  });  第二种方式交第一种而言在处理复杂数据时更为灵活，一般使用第二种方式来进行table数据填充。数据来源（url,type)，分页，排序，单元格的属性（width,height对其方式，样式cellStyle，数据格式化formatter）,绑定事件，复选框（check : true）等 |

# dialog的使用

|  |
| --- |
| **#引入依赖**  <link rel="stylesheet" href="$rc.contextPath/assets/css/bootstrap-dialog.min.css">  <script src="$rc.contextPath/assets/js/bootstrap-dialog.min.js"> </script> |
| **#使用例子**  BootstrapDialog.show({  title: 'dialog标题',  closable:false ,  size : BootstrapDialog.SIZE\_NORMAL,  type : BootstrapDialog.TYPE\_WARNING,  message: 'dialog content',  buttons: [{  id: 'btn-ok', //  icon: 'glyphicon glyphicon-check',  label: 'OK',  cssClass: 'btn-primary',  autospin: false,  action: function(dialogRef){  dialogRef.close(); //关闭dialog  }  }]  });  **#常用选项**  **title：**dialog标题  **closable：false**表示点击其他地方不关闭dialog  **size**：**BootstrapDialog.** **SIZE\_NORMAL** **SIZE\_WIDE** **SIZE\_LARGE**  **type**：弹窗类型**BootstrapDialog.TYPE\_PRIMARY** **TYPE\_DEFAULT TYPE\_DEFAULT** **TYPE\_SUCCESS** **TYPE\_WARNING** **TYPE\_DANGER**  **message**：主体内容  **buttons**  id：可选，如果设置了id，可以使用dialogInstance.getButton（id）来获取按钮。  icon：可选，如果设置，指定的图标将被添加到按钮。  cssClass：可选，附加css类添加到按钮。  autospin：optinal，如果是真的，点击按钮后出现旋转图标。  action：可选，如果提供，回调将在单击按钮后被调用，并且对话框实例将被传递到回调函数。  **#dialog事件**  BootstrapDialog.show({  message: 'Hello world!',  onshow: function(dialogRef){  alert('Dialog is popping up, its message is ' + dialogRef.getMessage());  },  onshown: function(dialogRef){  alert('Dialog is popped up.');  },  onhide: function(dialogRef){  alert('Dialog is popping down, its message is ' + dialogRef.getMessage());  },  onhidden: function(dialogRef){  alert('Dialog is popped down.');  }  }); |

# Pnotify插件的使用

|  |
| --- |
| #引入依赖  <link rel="stylesheet" href="$rc.contextPath/assets/css/pnotify. custom.min.css">  <script src="$rc.contextPath/assets/js/pnotify.custom.min.js">  </script> |
| #使用例子  new PNotify({  title: '错误!',  styling : 'bootstrap3',  text: '服务器异常',  type: 'error',  delay:5000  });  new PNotify({  title: '成功!',  styling : 'bootstrap3',  text: '操作成功',  type: 'success',  delay:1500  });  #配置默认值/选项  **title**: false - 通知的标题。  **title**\_escape: false - 是否转义标题的内容。（不允许HTML。）  **text**: false - 通知的文本。  **text\_escape**: false - 是否转义文本的内容。（不允许HTML。）  **styling**: "brighttheme" - 使用什么样式的类。（可以是“brighttheme”，“jqueryui”，“bootstrap2”，“bootstrap3”，“fontawesome”或自定义样式对象。有关样式对象中的属性，请参阅pnotify.js末尾的源代码。  **addclass**: "" - 要添加到通知中的其他类。（自定义样式）。  **cornerclass**: "" - 要添加到角样式通知中的课程。  **auto\_display**: true - 创建通知时显示通知。关闭此选项可将通知添加到历史记录，而不显示它们。  **width**: "300px" - 通知的宽度。  **min\_height**: "16px" - 通知的最小高度。它将扩展以适应内容。  **type**: "notice" - 通知类型。“notice”，“info”，“success”或“error”。  **icon**: true - 将图标设置为true，以使用所选样式/类型的默认图标，无图标时使用false，或为自己的图标类使用字符串。  **animation**: "fade" - 显示和隐藏通知时使用的动画。“none”，“show”，“fade”和“slide”是内置的jQuery。其他需要jQuery UI。使用带有effect\_in和effect\_out的对象来使用不同的效果。  **animate\_speed**: "slow" - 通知动画和播放的速度。“慢”，“def”或“正常”，“快速”或毫秒数。  **position\_animate\_speed**: 500 - 指定位置动画的特定持续时间。  **opacity**: 1 - 通知的不透明度。  **shadow**: true - 显示投影。  **hide**: true - 延迟后，请删除通知。  **delay**: 8000 - 删除通知之前的延迟（以毫秒为单位）。  **mouse\_reset**: true - 如果鼠标移过通知，则重置隐藏定时器。  **remove**: true - 删除通知的DOM元素后，它被删除。  **insert\_brs**: true - 将新行更改为br标签。  **stack**: {"dir1": "down", "dir2": "left", "push": "bottom", "spacing1": 25, "spacing2": 25, "context": $("body"), "modal": false} - 将放置通知的堆叠。还控制通知堆叠的方向。 |