目录

[一、 form表单组件 11](#_Toc471202641)

[1 效果图 11](#_Toc471202642)

[2 示例 11](#_Toc471202643)

[3 属性 15](#_Toc471202644)

[3.1 必填属性 15](#_Toc471202648)

[3.1.1 wings 15](#_Toc471202653)

[3.1.2 name 16](#_Toc471202654)

[3.2 非必填属性 16](#_Toc471202655)

[3.2.1 <table> 16](#_Toc471202656)

[3.2.2 <col> 16](#_Toc471202657)

[3.2.3 <colgroup> 16](#_Toc471202658)

[3.2.4 id 16](#_Toc471202659)

[3.2.5 caption,panel 16](#_Toc471202660)

[3.2.6 style 17](#_Toc471202661)

[4 可包含组件 17](#_Toc471202662)

[4.1 hidden 17](#_Toc471202663)

[4.1.1 type（必填） 17](#_Toc471202669)

[4.1.2 name（必填） 17](#_Toc471202670)

[4.1.3 defValue 17](#_Toc471202671)

[4.2 wingspsd 17](#_Toc471202672)

[4.2.1 type（必填） 17](#_Toc471202674)

[4.2.2 name（必填） 17](#_Toc471202675)

[4.2.3 defValue 17](#_Toc471202676)

[4.2.4 inputClass 17](#_Toc471202677)

[4.2.5 disable 17](#_Toc471202678)

[4.2.6 wingsReadonly 18](#_Toc471202679)

[4.2.7 css 18](#_Toc471202680)

[4.2.8 校验 18](#_Toc471202681)

[4.3 label 18](#_Toc471202682)

[4.3.1 type（必填） 18](#_Toc471202683)

[4.3.2 name（必填） 18](#_Toc471202684)

[4.3.3 defValue 18](#_Toc471202685)

[4.3.4 inputClass 18](#_Toc471202686)

[4.3.5 css 18](#_Toc471202687)

[4.4 text 18](#_Toc471202688)

[4.4.1 type（必填） 18](#_Toc471202689)

[4.4.2 name（必填） 18](#_Toc471202690)

[4.4.3 inputClass 18](#_Toc471202691)

[4.4.4 css 18](#_Toc471202692)

[4.4.5 defValue 19](#_Toc471202693)

[4.4.6 disable 19](#_Toc471202694)

[4.4.7 wingsReadonly 19](#_Toc471202695)

[4.4.8 校验 19](#_Toc471202696)

[4.5 textarea 19](#_Toc471202697)

[4.5.1 type（必填） 19](#_Toc471202707)

[4.5.2 name（必填） 19](#_Toc471202708)

[4.5.3 defValue 19](#_Toc471202709)

[4.5.4 css 19](#_Toc471202710)

[4.5.5 inputClass 19](#_Toc471202711)

[4.5.6 disable 19](#_Toc471202712)

[4.5.7 wingsReadonly 20](#_Toc471202713)

[4.5.8 rows 20](#_Toc471202714)

[4.5.9 校验 20](#_Toc471202715)

[4.6 radio 20](#_Toc471202716)

[4.6.1 type（必填） 20](#_Toc471202719)

[4.6.2 name（必填） 20](#_Toc471202720)

[4.6.3 defValue 20](#_Toc471202721)

[4.6.4 colWidth 20](#_Toc471202722)

[4.6.5 disable 20](#_Toc471202723)

[4.6.6 onClickBefore（事件） 20](#_Toc471202724)

[4.6.7 onClickAfter（事件） 20](#_Toc471202725)

[4.7 checkbox 21](#_Toc471202726)

[4.7.1 type（必填） 21](#_Toc471202738)

[4.7.2 name（必填） 21](#_Toc471202739)

[4.7.3 defValue 21](#_Toc471202740)

[4.7.4 colWidth 21](#_Toc471202741)

[4.7.5 disable 21](#_Toc471202742)

[4.7.6 onClickBefore（事件） 21](#_Toc471202743)

[4.7.7 onClickAfter（事件） 21](#_Toc471202744)

[4.8 multipleSelect 22](#_Toc471202745)

[4.8.1 type（必填） 22](#_Toc471202746)

[4.8.2 name（必填） 22](#_Toc471202747)

[4.8.3 dataName 22](#_Toc471202748)

[4.8.4 inputClass 22](#_Toc471202749)

[4.8.5 css 22](#_Toc471202750)

[4.8.6 defValue 22](#_Toc471202751)

[4.8.7 size 22](#_Toc471202752)

[4.8.8 popDisplay 22](#_Toc471202753)

[4.8.9 onSelect（事件） 22](#_Toc471202754)

[4.9 select 23](#_Toc471202755)

[4.10 date 23](#_Toc471202756)

[4.11 pulltree 23](#_Toc471202757)

[4.12 wingsValidator 23](#_Toc471202758)

[4.13 wingsSubmit 23](#_Toc471202759)

[5 接口详解 23](#_Toc471202760)

[5.1 getValue() 23](#_Toc471202761)

[5.2 setValue() 25](#_Toc471202762)

[5.3 getSubmitData() 26](#_Toc471202763)

[5.4 disable()/enable() 26](#_Toc471202764)

[5.5 resetAll() 26](#_Toc471202765)

[5.6 clearSetDefValue() 26](#_Toc471202766)

[二、 工具栏组件 26](#_Toc471202767)

[1 效果图 26](#_Toc471202768)

[2 示例 27](#_Toc471202769)

[3 属性 28](#_Toc471202770)

[3.1 必填属性 28](#_Toc471202771)

[3.1.1 wings 28](#_Toc471202772)

[3.1.2 name 28](#_Toc471202773)

[3.2 非必填属性 28](#_Toc471202777)

[3.2.1 isfixed 28](#_Toc471202778)

[3.3 按钮属性详解 28](#_Toc471202779)

[3.3.1 name(必填) 28](#_Toc471202780)

[3.3.2 caption 29](#_Toc471202781)

[3.3.3 type 29](#_Toc471202782)

[3.3.4 quickKey 29](#_Toc471202783)

[3.3.5 color 29](#_Toc471202784)

[3.3.6 enabled 29](#_Toc471202785)

[4 事件 29](#_Toc471202786)

[4.1 onclick 29](#_Toc471202790)

[5 接口详解 29](#_Toc471202791)

[5.1 setHide/setDisplay 29](#_Toc471202792)

[5.2 setDisabled/setEnabled 29](#_Toc471202793)

[三、 下拉组件 30](#_Toc471202794)

[1 属性 30](#_Toc471202795)

[1.1 必填属性 30](#_Toc471202796)

[1.1.1 name 30](#_Toc471202797)

[1.1.2 type 30](#_Toc471202798)

[1.2 非必填属性 30](#_Toc471202799)

[1.2.1 dataName 30](#_Toc471202800)

[1.2.2 inputDisplay 30](#_Toc471202801)

[1.2.3 popDisplay 30](#_Toc471202802)

[1.2.4 inputClass 31](#_Toc471202803)

[1.2.5 css 31](#_Toc471202804)

[1.2.6 defValue 31](#_Toc471202805)

[2 选项数据集 31](#_Toc471202806)

[2.1 code（必填） 31](#_Toc471202807)

[2.2 caption（必填） 31](#_Toc471202808)

[3 事件详解 31](#_Toc471202809)

[3.1 onselect 31](#_Toc471202810)

[4 接口详解 31](#_Toc471202811)

[4.1 表单组件 31](#_Toc471202812)

[4.2 表格组件 31](#_Toc471202813)

[5 前后台交互 31](#_Toc471202814)

[5.1 addSelectWithDataName 32](#_Toc471202815)

[5.2 getFormData 32](#_Toc471202816)

[四、 进度条组件 32](#_Toc471202817)

[1 属性 32](#_Toc471202818)

[1.1 必填属性 32](#_Toc471202819)

[1.1.1 wings 32](#_Toc471202820)

[1.1.2 name 32](#_Toc471202821)

[2 接口详解 32](#_Toc471202822)

[2.1 Close() 32](#_Toc471202823)

[五、 grid表格组件 32](#_Toc471202824)

[1. 属性 32](#_Toc471202825)

[1.1 表级属性 32](#_Toc471202826)

[1.1.1 wings(必填) 32](#_Toc471202827)

[1.1.2 name(必填) 32](#_Toc471202828)

[1.1.3 pageLength 32](#_Toc471202829)

[1.1.4 paging 32](#_Toc471202830)

[1.1.5 pagingType 33](#_Toc471202831)

[1.1.6 info 33](#_Toc471202832)

[1.1.7 ordering 33](#_Toc471202833)

[1.1.8 lineNumber 33](#_Toc471202834)

[1.1.9 lineColumnTitle 33](#_Toc471202835)

[1.1.10 rowSelect 33](#_Toc471202836)

[1.1.11 selectColor 33](#_Toc471202837)

[1.1.12 hideMenu 33](#_Toc471202838)

[1.1.13 hideMenu\_exclude 33](#_Toc471202839)

[1.1.15 border 33](#_Toc471202840)

[1.1.16 showHeader 33](#_Toc471202841)

[1.1.17 itemX 33](#_Toc471202842)

[1.1.18 striped 33](#_Toc471202843)

[1.1.19 nowrap 34](#_Toc471202844)

[1.1.20 contentLen 34](#_Toc471202845)

[1.1.21 checkColor 34](#_Toc471202846)

[1.1.22 dataFontSize 34](#_Toc471202847)

[1.1.23 fenye 34](#_Toc471202848)

[1.1.24 ptid（fenye=”true”） 34](#_Toc471202849)

[1.1.25 fzbtid 36](#_Toc471202850)

[1.2 列级属性 38](#_Toc471202851)

[1.2.1 name 38](#_Toc471202852)

[1.2.2 caption 38](#_Toc471202853)

[1.2.3 contentLen 38](#_Toc471202854)

[1.2.4 defaultValue 38](#_Toc471202855)

[1.2.5 x 38](#_Toc471202856)

[1.2.6 type 38](#_Toc471202857)

[1.2.6.1 hidden 38](#_Toc471202858)

[1.2.6.2 text 38](#_Toc471202859)

[1.2.6.3 a 38](#_Toc471202860)

[1.2.6.4 password 39](#_Toc471202861)

[1.2.6.5 pulltree 39](#_Toc471202862)

[1.2.6.6 select 39](#_Toc471202863)

[1.2.6.7 date 39](#_Toc471202864)

[2. 接口 39](#_Toc471202865)

[2.1 getCellEl () 39](#_Toc471202866)

[2.2 getRowEl() 39](#_Toc471202867)

[2.3 getColEl() 40](#_Toc471202868)

[2.4 getSelectedRow() 40](#_Toc471202869)

[2.5 getCheckedRow() 40](#_Toc471202870)

[2.6 setAjaxParam(param) 40](#_Toc471202871)

[2.7 getAjaxParam() 41](#_Toc471202872)

[2.8 getAllGridData 41](#_Toc471202873)

[2.9 getStatusGridData 41](#_Toc471202874)

[2.10 getAllNoDeleteGridData 41](#_Toc471202875)

[2.11 getAllGridShowData 41](#_Toc471202876)

[2.12 getCurPageGridData 41](#_Toc471202877)

[2.13 insertRow(dataObj) 41](#_Toc471202878)

[2.14 deleting（rowEls） 42](#_Toc471202879)

[2.15 updateCell(cellW,dataValue,showValue) 42](#_Toc471202880)

[2.16 updateRow (rows,dataObj) 43](#_Toc471202881)

[2.17 bind() 43](#_Toc471202882)

[3. 事件 44](#_Toc471202883)

[3.1 wingsOnclick 44](#_Toc471202884)

[3.2 onRowClick 44](#_Toc471202885)

[3.3 onBeforeCreateCell 44](#_Toc471202886)

[3.4 onAfterCreateRow 44](#_Toc471202887)

[3.5 wingsmouseover 45](#_Toc471202888)

[3.7 wingsmouseover 45](#_Toc471202889)

[六、 提交组件 46](#_Toc471202890)

[1. 属性 46](#_Toc471202891)

[1.1 必填属性 46](#_Toc471202892)

[1.1.1 wings 46](#_Toc471202893)

[1.1.2 name 46](#_Toc471202894)

[1.2 非必填属性 46](#_Toc471202895)

[1.2.1 btnName 46](#_Toc471202898)

[2. 事件 46](#_Toc471202899)

[2.1 onclick 46](#_Toc471202900)

[2.2 onSuccess 46](#_Toc471202901)

[3. 接口详解 46](#_Toc471202902)

[3.1 提交方式（默认Ajax提交） 46](#_Toc471202903)

[3.1.1 Ajax提交 46](#_Toc471202904)

[3.1.2 表单提交 47](#_Toc471202905)

[3.2 pushData 47](#_Toc471202906)

[3.2.1 传递单值 47](#_Toc471202912)

[3.2.2 传递整个form 48](#_Toc471202913)

[3.2.3 传递整个grid 48](#_Toc471202914)

[3.3 setFunction 48](#_Toc471202915)

[3.4 setOptions 49](#_Toc471202916)

[七、 树组件 50](#_Toc471202917)

[1 属性 50](#_Toc471202918)

[1.1 必填属性 50](#_Toc471202919)

[1.1.1 wings 50](#_Toc471202920)

[1.1.2 name 50](#_Toc471202921)

[1.2 非必填属性 50](#_Toc471202922)

[1.2.1 treeType 50](#_Toc471202923)

[1.2.2 css 50](#_Toc471202924)

[1.2.3 defValue 50](#_Toc471202925)

[1.2.4 inputClass 50](#_Toc471202926)

[1.2.5 height 50](#_Toc471202927)

[2 事件 50](#_Toc471202928)

[2.1 onclick 50](#_Toc471202929)

[2.2 onCheckBefore 51](#_Toc471202930)

[2.3 onLtidBefore 51](#_Toc471202931)

[2.4 onNodeClick 51](#_Toc471202932)

[3 接口详解 51](#_Toc471202933)

[3.1 initData() 51](#_Toc471202934)

[3.2 setValue() 51](#_Toc471202935)

[3.3 getValue() 51](#_Toc471202936)

[八、 日期组件 51](#_Toc471202937)

[1 效果图 51](#_Toc471202938)

[2 属性 52](#_Toc471202939)

[2.1 必填属性 52](#_Toc471202940)

[2.1.1 wings 52](#_Toc471202941)

[2.1.2 name 52](#_Toc471202942)

[2.1.3 type 52](#_Toc471202943)

[2.2 非必填属性 52](#_Toc471202944)

[2.2.1 startDate/endDate 52](#_Toc471202945)

[2.2.2 format 52](#_Toc471202946)

[2.2.3 inputClass 52](#_Toc471202947)

[2.2.4 css 52](#_Toc471202948)

[2.2.5 defValue 52](#_Toc471202949)

[2.2.6 disable 52](#_Toc471202950)

[2.2.7 wingsReadonly 52](#_Toc471202951)

[2.2.8 startDate 53](#_Toc471202952)

[2.2.9 endDate 53](#_Toc471202953)

[2.2.10 校验 53](#_Toc471202954)

[3 事件 53](#_Toc471202955)

[3.1 onDateChange 53](#_Toc471202959)

[九、 弹出框 53](#_Toc471202960)

[1 提示框 53](#_Toc471202961)

[1.1 效果图 53](#_Toc471202962)

[1.2 使用方法 53](#_Toc471202963)

[1.3 示例 53](#_Toc471202964)

[2 确认框 54](#_Toc471202965)

[2.1 效果图 54](#_Toc471202966)

[2.2 使用方法 54](#_Toc471202967)

[2.3 示例 54](#_Toc471202968)

[2.4 方法 54](#_Toc471202969)

[3 模态窗口 54](#_Toc471202970)

[3.1 效果图 54](#_Toc471202971)

[3.2 使用方法 55](#_Toc471202972)

[3.3 示例 55](#_Toc471202973)

[3.4 属性 55](#_Toc471202974)

[3.5 方法 56](#_Toc471202975)

[十、 Tab组件 56](#_Toc471202976)

[1 属性 56](#_Toc471202977)

[1.1 必填属性 56](#_Toc471202978)

[1.1.1 wings 56](#_Toc471202979)

[1.1.2 name 56](#_Toc471202980)

[1.1.3 tabTitles 56](#_Toc471202981)

[1.1.4 tabDivs 56](#_Toc471202982)

[1.2 非必填属性 56](#_Toc471202983)

[1.2.1 tabSelectedIndex 56](#_Toc471202984)

[1.2.2 inputClass 56](#_Toc471202985)

[十一、 公共方法 57](#_Toc471202986)

[1 $chk() 57](#_Toc471202987)

[2 $.wingsWidget() 57](#_Toc471202988)

[十二、 初始化组件 57](#_Toc471202989)

[1 属性 57](#_Toc471202990)

[1.1 wings 57](#_Toc471202991)

[2 事件 57](#_Toc471202992)

[2.1 onSuccess 57](#_Toc471202993)

[十三、 校验组件 57](#_Toc471202994)

[1 校验方法 58](#_Toc471202995)

[2 设置方法 58](#_Toc471202996)

[2.1 text 58](#_Toc471202997)

[2.2 wingsPwd 58](#_Toc471202998)

[2.3 wingsData 58](#_Toc471202999)

[3 rule属性 58](#_Toc471203000)

[十四、 左右选择器 59](#_Toc471203001)

[1 效果图 59](#_Toc471203002)

[2 示例 59](#_Toc471203003)

[3 属性 62](#_Toc471203004)

[3.1 wings（必填） 62](#_Toc471203005)

[3.2 name（必填） 62](#_Toc471203006)

[4 方法 62](#_Toc471203007)

[4.1 initData () 62](#_Toc471203008)

[4.2 getLeftInsertData()/getRightInsertData() 62](#_Toc471203009)

[4.3 getLeftDeleteData()/getRightDeleteData() 62](#_Toc471203010)

[5 与后台交互 63](#_Toc471203011)

[6 提交后初始化 63](#_Toc471203012)

[十五、 搜索建议组件 64](#_Toc471203013)

[1 示例 64](#_Toc471203014)

[2 属性 65](#_Toc471203015)

[2.1 必填属性 65](#_Toc471203016)

[2.1.1 wings 65](#_Toc471203017)

[2.1.2 name 65](#_Toc471203018)

[2.1.3 lctrl 65](#_Toc471203019)

[2.1.4 rowName 65](#_Toc471203020)

[2.1.5 keyField（当rowName的值为多个时） 67](#_Toc471203021)

[2.2 非必填属性 67](#_Toc471203022)

[2.2.1 showBtn 67](#_Toc471203023)

[2.2.2 showHeader,headerTitles 67](#_Toc471203024)

[3 事件 67](#_Toc471203025)

[3.1 onLtidBefore 67](#_Toc471203026)

[4 方法 67](#_Toc471203027)

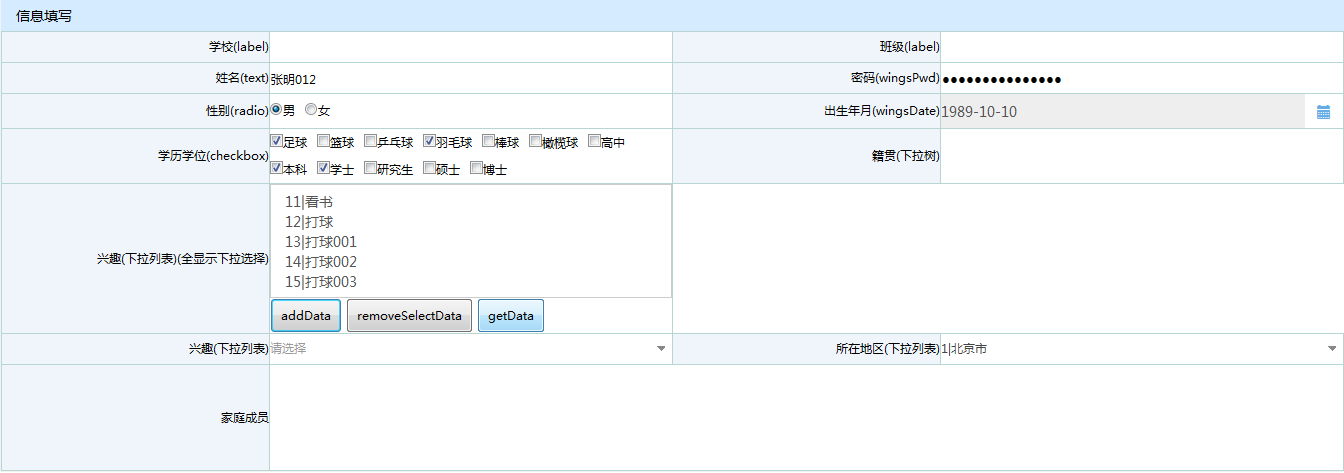
[4.1 disable()/enable() 67](#_Toc471203028)

[4.2 show()/hide() 68](#_Toc471203029)

[5 前后台交互 69](#_Toc471203030)

[5.1 后台获取 69](#_Toc471203031)

[5.2 后台到前台 70](#_Toc471203032)

1. form表单组件
2. 效果图
3. 示例

|  |
| --- |
| <html>  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <title>表单</title>  <!--引用基本组件js-->  <jsp:include page="/wings/core/common/wingsInclude.jsp">  </jsp:include>  <!--引用form组件js-->  <jsp:include page="/wings/core/common/wingsFormInclude.jsp">  </jsp:include>  <!—引用页面自己js-->  <script src="/wings/core/demoWings/wingsForm/wingsForm001.js">  </script>  </head>  <body>  <!—form组件开始-->  <div wings='WingsForm' name='studentForm' caption='信息填写'  panel="false">  <!—hidden组件-->  <div name="hidStr" type="hidden" ></div>  <!—设置布局-->  <table class="tab\_form " >  <colgroup>  <col style="width: 20%"></col>  <col style="width: 30%"></col>  <col style="width: 20%"></col>  <col style="width: 30%"></col>  </colgroup>  <tr >  <th style="text-align: right;" >学校(label)</th>  <td >  <!—label组件-->  <div name="xx" type="label" defValue="昌平一中"></div>  </td>  <th style="text-align: right;" >班级(label)</th>  <td >  <div name="bj" type="label" ></div>  </td>  </tr>  <tr>  <th style="text-align: right;" >姓名(text)</th>  <td >  <!—text组件-->  <div name="xm" type="text" required="true" ></div>  </td>  <th style="text-align: right;" >密码(wingsPwd)</th>  <td >  <!—wingsPwd组件-->  <div name="ywm" type="wingsPwd" required="true" ></div>  </td>  </tr>  <tr>  <th style="text-align: right;" >性别(radio)</th>  <td>  <!—radio组件-->  <div name="xb" type="radio" onClickBefore="radio\_clickbefore(111,222);" >  <div code="1" caption="男"></div>  <div code="2" caption="女"></div>  </div>  </td>  <th style="text-align: right;">出生年月(wingsDate)</th>  <td>  <!—日期组件-->  <div wings='WingsCalendar' name="csTime" onDateChange='onDataChange\_fn();'startDate="2016-11-11" endDate="2016-11-30"type="wingsDate" wingsReadonly="true" ></div>  </td>  </tr>  <tr>  <th style="text-align: right;" >学历学位(checkbox)</th>  <td >  <!—checkbox组件-->  <div name="xlxw" type="checkbox" defValue ='111,112' onClickAfter="alert(333);">  <div code="01" caption="高中"></div>  <div code="02" caption="本科"></div>  <div code="03" caption="学士"></div>  <div code="04" caption="研究生"></div>  <div code="05" caption="硕士"></div>  <div code="06" caption="博士"></div>  </div>  </td>  <th style="text-align: right;" >籍贯(下拉树)</th>  <td >  <!—下拉树组件-->  <div name="leiTree" type="pulltree" treeType = '1'  height='150' ></div>  </td>  </tr>  <tr>  <th style="text-align: right;" >兴趣(全显示下拉选择)</th>  <td >  <!-- multipleSelect 组件-->  <div name="xq111" type="multipleSelect" onSelect="onselect\_fn();" inputClass="form-control" size="5" popDisplay ="{code}|{caption}" >  <div code="11" caption="看书"></div>  <div code="12" caption="打球"></div>  <div code="13" caption="打球001"></div>  <div code="14" caption="打球002"></div>  <div code="15" caption="打球003"></div>  </div>  </td>  </tr>  <tr>  <th style="text-align: right;" >兴趣(下拉列表)</</th>  <td >  <!—下拉框组件（页面设置下拉选项）-->  <div name="xq" type="select" onSelect="onselect\_fn();" popDisplay ="{code}|{caption}" >  <div code="11" caption="看书"></div>  <div code="12" caption="打球"></div>  </div>  </td>  <th style="text-align: right;" >所在地区(下拉列表)</th>  <td >  <!—下拉框组件（后台设置下拉选项）-->  <div name="mySelect" type="select" onSelect="onselect\_fn();" dataName="xbSel" popDisplay ="{code}|{caption}" ></div>  </td>  </tr>  <tr>  <th style="text-align: right;" >家庭成员</</th>  <td colspan="3" >  <!—文本域组件-->  <div name="textarea001" type="textarea" required="true" >  </div>  </td>  </tr>  </table>  </div>  </div>  </body>  </html> |

1. 属性

4. 1. 必填属性

7. 1. 1. wings

wings=”wingsForm”用来声明form组件。

* + 1. name

name属性定义表单组件的名称，属性值为表单组件的唯一标识, name属性还对提交到服务器后的表单数据进行标识，后台在返回数据给表单时，通过name属性值找到对应表单。

* 1. 非必填属性
     1. <table>

<table>标签定义表单组件的布局，Web框架提供一套默认的样式：class="tab\_form"，可以根据项目的需求更改css样式。 简单的表单组件由table元素以及一个或多个tr、th、td、col、colgroup元素组成，其中: tr---定义表格行; td---定义单元格; th---定义表头; col---为一个或多个列定义属性值; colgroup---用于对表单组件中的列进行组合，以便对其进行格式化。如果没有此标签需要自己调节样式。

* + 1. <col>

<col>标签为表单组件中一个或多个列定义属性值。如需对全部列应用样式，<col> 标签很有用，这样就不需要对各个单元格和各行重复应用样式（注意: <col>标签代表每一列，设置的宽度width总和要等于100%）。如果没有此标签需要自己调节样式。

* + 1. <colgroup>

<colgroup>标签用于对表单组件中的列进行组合，以便对其进行优化。如需对全部列应用样式，<colgroup>标签很有用，这样无需对各个行和各个单元格重复应用样式（注意: <colgroup>标签只能在<table>标签中使用）如果没有此标签需要自己调节样式。。

* + 1. id

id属性定义表单组件的唯一标识（注意: 可以为html元素设置id属性，通过$("id属性值")接口获取到id对应的元素）。取值可以为任意值，不可为空，且在HTML文档中唯一。

* + 1. caption,panel

caption属性值标识表单组件的标题，取值可以为任意值，可为空。Panel属性值标识是否显示表单组件的标题栏，值为false或true。

false(默认值)---字符串类型，不显示表单组件的标题栏。

true---字符串类型，显示表单组件的标题栏。

* + 1. style

表单组件内的标签都可以使用style属性定义css样式

1. 可包含组件
   1. hidden

例：

|  |
| --- |
| //在form中  <div name="hidStr" type="hidden" defValue=“1”></div> |

4. 1. 1. type（必填）

表单中使用type="hidden"声明hidden类型的元素。

* + 1. name（必填）

name属性定义hidden元素的唯一标识。页面中hidden元素对于用户是不可见的，在表单中插入hidden元素的目的在于收集或发送信息，以利于处理表单的程序所使用。name属性还对提交到服务器后的hidden元素的数据进行标识，后台在返回数据给表单的hidden元素时，通过name属性值找到对应的hidden元素。

* + 1. defValue

defValue属性用来设置默认值。

* 1. wingspsd
  2. 1. type（必填）

表单中使用type="wingspsd"声明wingspsd类型的元素。

* + 1. name（必填）

name属性定义wingspsd元素的唯一标识。name属性还对提交到服务器后的wingspsd类型元素的数据进行标识，后台在返回数据给表单的wingspsd元素时，通过name属性值找到对应的wingspsd元素。

* + 1. defValue

defValue属性用来设置默认值。

* + 1. inputClass

inputClass属性用来定义wingspsd的类，相当于class属性。

* + 1. disable

disable属性标识wingspsd元素为禁用状态。

false(默认值)--- wingspsd元素为可编辑状态

true---password元素为禁用状态

* + 1. wingsReadonly

wingsReadonly属性标识wingspsd元素为只读状态。

false(默认值)--- wingspsd元素为可编辑状态。

true---wingspsd元素为只读状态。

* + 1. css

css属性用来定义wingspsd的样式。

* + 1. 校验

wingspsd关于校验包含required,rangelength,equalTo,

zdyRule, 具体用法参见“校验组件”一节。

* 1. label
     1. type（必填）

表单中使用type="label"声明label类型的元素。

* + 1. name（必填）

name属性定义label元素的唯一标识。

* + 1. defValue

defValue属性用来定义label的默认值。

* + 1. inputClass

inputClass属性用来定义label元素的类，相当于class属性。

* + 1. css

css属性用来定义元素的样式。

* 1. text
     1. type（必填）

表单中使用type="text"声明text类型的元素。

* + 1. name（必填）

name属性定义text元素的唯一标识。wingsReadonly属性设置是否为‘只读’。

* + 1. inputClass

inputClass属性用来定义label元素的类，相当于class属性。

* + 1. css

css属性用来定义元素的样式。

* + 1. defValue

defValue属性用来定义text的默认值。

* + 1. disable

disable属性标识text元素为禁用状态。

false(默认值)---text元素为可编辑状态。

true---text元素为禁用状态。

* + 1. wingsReadonly

wingsReadonly属性标识此文本为只读，不可进行更改删除等操作。

false(默认值)---text元素为可编辑状态。

true---text元素为只读状态。

* + 1. 校验

text关于校验包含rule，rangelength，maxnb，minnb，zdyRule，required，具体用法参见“校验组件”一节。

* 1. textarea

4. 5. 1. type（必填）

表单中使用type="textarea"声明textarea类型元素。

* + 1. name（必填）

name属性定义textarea元素的唯一标识。name属性还对提交到服务器后的textarea类型的数据进行标识，后台在返回数据给表单的textarea元素时，通过name属性值找到对应的textarea元素。

* + 1. defValue

defValue属性用来设置默认值。

* + 1. css

css属性用来定义元素的样式。

* + 1. inputClass

inputClass属性用来定义textarea元素的类，相当于class属性。

* + 1. disable

disable属性标识textarea元素为禁用状态。

false(默认值)---textarea元素为可编辑状态。

true---textarea元素为禁用状态。

* + 1. wingsReadonly

wingsReadonly属性标识此文本域为只读，不可进行更改删除等操作。

false(默认值)---textarea元素为可编辑状态。

true---textarea元素为只读状态。

* + 1. rows

rows属性定义textarea占的行数。

* + 1. 校验

text关于校验包含rule，rangelength，maxnb，minnb，zdyRule，required，具体用法参见“校验组件”一节。

* 1. radio
  3. 1. type（必填）

表单使用type="radio"声明radio类型元素。

* + 1. name（必填）

name属性定义 radio元素的唯一标识。

* + 1. defValue

defValue属性用来设置默认值。

* + 1. colWidth

定义radio类型的选项间的宽度。

* + 1. disable

disable属性标识radio元素为禁用状态。

false(默认值)---radio元素为可选中状态。

true---radio元素为禁用状态。

* + 1. onClickBefore（事件）

点击radio的选项之前触发此事件。

* + 1. onClickAfter（事件）

点击radio的选项之后触发此事件。

例：

|  |
| --- |
| <div name="xb" type="radio"  onClickBefore="radio\_clickbefore();"onClickAfter="alert(333)" >  <div code="1" caption="男"></div>  <div code="2" caption="女"></div>  </div> |

* 1. checkbox

4. 7. 1. type（必填）

表单组件使用type="checkbox"声明checkbox类型元素。

* + 1. name（必填）

name属性定义checkbox元素的唯一标识。

* + 1. defValue

defValue属性设置checkbox元素的默认选中项。如果表单初始化时服务器没有给checkbox元素传数据，那么checkbox元素值为defValue属性值。

* + 1. colWidth

定义checkbox选项间的宽度。

* + 1. disable

disable属性标识checkbox元素为禁用状态。

false(默认值)---checkbox元素为可选中状态。

true---checkbox元素为禁用状态。

* + 1. onClickBefore（事件）

点击checkbox的选项之前触发此事件。

* + 1. onClickAfter（事件）

点击checkbox的选项之后触发此事件。

例：

|  |
| --- |
| <div name="xlxw" type="checkbox" defValue ='111,112'  onClickAfter="alert(333);">  <div code="01" caption="高中"></div>  <div code="02" caption="本科"></div>  </div>  //js代码  function radio\_clickbefore(code,caption){//例如点击了男  alert("选项的code值为："+code); //1  alert("选项的caption值为："+caption);//男  } |

* 1. multipleSelect
     1. type（必填）

表单组件使用type=" multipleSelect"声明checkbox类型元素。

* + 1. name（必填）

name属性定义multipleSelect元素的唯一标识。

* + 1. dataName

当multipleSelect的选项数据从后台传送时，通过dataName属性找到相应的multipleSelect。multipleSelect的选项数据可以在页面中定义，也可以后台服务传送。当multipleSelect没有定义dataName属性时，后台服务通过name属性值传送选项数据。

* + 1. inputClass

inputClass属性用来定义multipleSelect元素的类，相当于class属性。

* + 1. css

css属性用来定义元素的样式。

* + 1. defValue

defValue属性用来设置默认值。

* + 1. size

用来定义multipleSelect的宽度显示几个选项，超出部分出现滚动条。

* + 1. popDisplay

popDisplay用来定义选项显示值的形式。

{caption}(默认值)--默认显示选项名称。

当选项显示为数据集的多个属性值时，中间用"|"分隔，例如"{code}|{caption}"，也可以包含自定义的属性。

* + 1. onSelect（事件）

当点击选项时触发此事件。

例：

|  |
| --- |
| 页面：  <div name="xq111" type="multipleSelect" onSelect="onselect\_fn();" inputClass="form-control"  dataName="xbSel" size="5" >  <div code="11" caption="看书"></div>  <div code="12" caption="打球"></div>  <div code="13" caption="打球001"></div>  <div code="14" caption="打球002"></div>  <div code="15" caption="打球003"></div>  </div>  Js代码:  function onselect\_fn(code,caption){  alert("选项的code值为："+code); //1  alert("选项的caption值为："+caption);//男  } |

* 1. select

参见wingsselect一节。

* 1. date

参见wingsdate一节。

* 1. pulltree

参见wingstree一节。

* 1. wingsValidator

参见wingsValidator一节。

* 1. wingsSubmit

参见wingsSubmit一节。

1. 接口详解
   1. getValue()

获取（）里name对应的对象的value值。

select，radio，checkbox返回值为code值。

例：

|  |
| --- |
| //页面  <div wings='WingsForm' name='studentForm' caption='信息填写'  panel="true" >  <table class="tab\_form " >  <colgroup>  <col style="width: 20%"></col>  <col style="width: 30%"></col>  <col style="width: 20%"></col>  <col style="width: 30%"></col>  </colgroup>  <tr>  <th style="text-align: right;" >姓名(text)</th>  <td >  <div name="xm" type="text" required="true" >张明  </div>  </td>  <th style="text-align: right;" >密码(wingsPwd)</th>  <td >  <div name="ywm" type="wingsPwd" required="true" ></div>  </td>  </tr>  <tr>  <th style="text-align: right;" >兴趣(下拉列表)</th>  <td >  <div name="xq" type="select" onSelect="onselect\_fn();" popDisplay ="{code}|{caption}" >  <div code="11" caption="看书"></div>  <div code="12" caption="打球"></div>  </div>  </td>  <th style="text-align: right;" >所在地区(下拉列表)</th>  <td >  <div name="mySelect" type="select" onSelect="onselect\_fn();" dataName="xbSel" popDisplay ="{code}|{caption}" ></div>  </td>  </tr>  <tr>  <th style="text-align: right;" >家庭成员</</th>  <td colspan="3" >  <div name="textarea001" type="textarea" required="true" ></div>  </td>  </tr>  </table>  </div>  //js中获取studentForm表单中name为xq的value值（结果为选中项的value值）  alert("选项的value值为："$.wingsWidget('studentForm').getValue('xq'));  //获取studentForm表单中name为xm的value值（结果显示‘张明’）  alert("value值为："$.wingsWidget('studentForm').getValue('xm')); |

* 1. setValue()

setValue()有两个参数，参数1为要改变值的对象的name,参数2为要设置的值。

select，radio，checkbox类型设置值时，参数2需传入要设置选项的code值。

例:

|  |
| --- |
| //js中设置name为xm(type=”text”)的对象的值为111  $.wingsWidget('studentForm').setValue('xm','111'); |

* 1. getSubmitData()

通过此接口获取需要提交到后台服务器的表单数据。

|  |
| --- |
| function wingsSubmit\_form(){  //获取页面表单对象。  var form = $.wingsWidget('studentForm');  var sub = new WingsSubmit();  // from刷新页面的提交方式  sub.options.postType ='form';  sub.pushData("hidStr123",form.getValue('hidStr'));  //获取表单提交数据  sub.pushData(form.getSubmitData());  sub.setCtrl('WingsFormCtrl\_formTest');  sub.submit();  } |

* 1. disable()/enable()

设置form里的项不可用/可用。参数可以为单个name值，也可以为多个name值

例：

|  |
| --- |
| //设置name为xm的项不可用  form.disable('xm');  //设置name为xm，ywm两项不可用  form.disable(['xm','ywm']);  //设置name为xm，ywm两项可用。  form.enable(['xm','ywm']); |

* 1. resetAll()

重新加载，但无法加载ajax返回的值，支持页面 defValue。

* 1. clearSetDefValue()

清除默认值。

1. 工具栏组件
2. C:\Users\3\AppData\Roaming\Tencent\Users\489614500\QQ\WinTemp\RichOle\2PIGPVWY0YR)CD5HQHR[ZCV.png效果图



1. 示例

|  |
| --- |
| <%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>  <!doctype html>  <html>  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <title>按钮</title>  <!—引入基础js-->  <jsp:include page="/wings/core/common/wingsInclude.jsp">  </jsp:include>  </head>  <body>  <div wings="WingsToolBar" name="test" isFixed="true" showType="normal">  <div name="add" type="add" quickKey=" ctrl+y "  onclick="alert('add')">增加</div>  <div name="add" type="close" quickKey="CTRL+X"  onclick="alert('close')">关闭</div>  <div name="print" onclick="alert('print')">打印</div>  <div name="export" enabled="false" onclick="alert('export')">导出</div>  <div name="import" onclick="alert('import')">导入</div>  <div name="back" onclick="alert('back')">返回</div>  <div name="preview" onclick="alert('preview')">预览</div>  <div name="inspect" onclick="alert('inspect')">启动</div>  <div name="previousPage" onclick="alert('previousPage')">上一页</div>  <div name="nextPage" onclick="alert('nextPage')">下一页</div>  <div name="firstPage" onclick="alert('firstPage')">首页</div>  <div name="endPage" onclick="alert('endPage')">末页</div>  <div name="delete" onclick="alert('delete')">删除</div>  <div name="edit" onclick="alert('edit')">修改</div>  <div name="find" onclick="alert('find')">查询</div>  <div name="save" onclick="alert('save')">保存</div>  <div name="refresh" onclick="alert('refresh')">刷新</div>  </div>  </body>  </html> |

1. 属性
   1. 必填属性
      1. wings

使用<div>标签的sword属性声明工具栏组件,wings="WingsToolBar"

* + 1. name

name属性定义工具栏组件的名称，属性值为工具栏组件的唯一标识

3. 1. 非必填属性
      1. isfixed

控制工具栏组件在页面滚动时是否产生浮动效果,置于顶部固定位置。

true(默认值)--置于顶部，false—不置于顶部。

* 1. 按钮属性详解
     1. name(必填)

工具栏按钮名称，必填当值为add,close,print,edit,delete,find,export,import,back,preview,inspect,previousPage,nextPage,firstPage,endPage,save,refresh时，组件会自动生成默认按钮。

* + 1. caption

工具栏按钮显示名称，使用此属性可以为默认按钮重新命名，也可以用于自定义按钮的显示名称

* + 1. type

定义工具栏组件的按钮类型, wingsToolBar组件中包含一些默认样式的业务按钮type样式，如add,close,print,edit,delete,find,export,import,back,preview,inspect,previousPage,nextPage,firstPage,endPage,save,refresh。当type与name值不一致时，以type值为准。

* + 1. quickKey

为toolBar组件按钮增加快捷键，格式为ctrl+任意字母。

* + 1. color

为按钮设置颜色。

* + 1. enabled

定义工具栏按钮是否可用.

true(默认值)--可用,false--不可用。

1. 事件

4. 1. onclick

点击工具栏按钮时触发此事件。

1. 接口详解
   1. setHide/setDisplay

设置按钮隐藏/显示，参数为buttonName，工具栏按钮的name属性值，用法工具栏对象.setHide(buttonName)，工具栏对象.setDisplay(buttonName)

* 1. setDisabled/setEnabled

设置按钮为不可用/可用状态，参数为buttonName，工具栏按钮的name属性值，用法工具栏对象.setDisabled(buttonName)工具栏对象.setEnabled(buttonName)。

例:

|  |
| --- |
| Js代码：  var toolbar = $.wingsWidget("toolbar");  //设置name为“find”的wingsToolbar按钮可用。  toolbar.setEnabled("find");  //设置name为“find”的wingsToolbar按钮不可用。 toolbar.setDisabled('find'); |

1. 下拉组件
2. 属性
   1. 必填属性
      1. name

name属性定义下拉列表组件的名称，属性值为下拉列表组件的唯一标识。name属性还对提交到服务器后的下拉列表数据进行标识，后台在返回数据给下拉列表时，可以通过name属性值找到对应下拉列表。只有设置了name属性的下拉列表元素，才能在提交时传递它们的值。

* + 1. type

type属性定义表格或表单中的元素为下拉列表（注意:下拉列表组件不能单独使用，必须依附于表单或者表格组件）。

* 1. 非必填属性
     1. dataName

dataName当下拉列表的选项数据从后台传送时，通过dataName属性找到相应的下拉列表，下拉列表的选项数据可以在页面中定义，也可以后台服务传送，后台服务通过res.addSelectWithDataName(key,list);传送下拉列表的选项数据，当下拉列表没有定义dataName属性时，后台服务通过name属性值传送选项数据。

* + 1. inputDisplay

inputDisplay属性用来定义下拉选中选项的显示形式。

例: inputDisplay="{caption}"。

* + 1. popDisplay

popDisplay用来定义选项显示值的形式。

{caption}(默认值)--默认显示选项名称。

当选项显示为数据集的多个属性值时，中间用"|"分隔，例如"{code}|{caption}"。

* + 1. inputClass

inputClass属性用来定义multipleSelect元素的类，相当于class属性。

* + 1. css

css属性用来定义元素样式。

* + 1. defValue

defValue属性用来定义下拉组件的默认值。

1. 选项数据集
   1. code（必填）

code属性定义下拉列表选项的code代码值，默认向后台提交下拉列表输入框显示值的code值(下拉列表的选项数据可以在页面中定义，也可以后台服务传送，后台服务通过res.addSelectWithDataName(key,list);传送下拉列表的选项数据)。code值为字符串类型，在当前数据集中code需唯一。

* 1. caption（必填）

caption属性定义下拉列表选项的名称，下拉列表的选项数据必须包含code和caption属性，选项数据可以在前台页面定义，也可以由服务器返回。

1. 事件详解
   1. onselect

当点击下拉列表的选项时触发此事件，参数为selectDiv，当前点击的选项div元素。

1. 接口详解
   1. 表单组件

表单组件中的下拉列表元素的相关接口请参考wingsForm(表单组件)接口详解

* 1. 表格组件

表格组件中的下拉列表元素的相关接口请参考wingsGrid（表格组件）接口详解

1. 前后台交互
   1. addSelectWithDataName

res的addSelectWithDataName方法向前台下拉列表元素传送下拉选项数据集。

例:

|  |
| --- |
|  |

* 1. getFormData

req的getFormData("formName")方法获取表单数据。

1. 进度条组件
2. 属性
   1. 必填属性
      1. wings

使用<div>标签的sword属性声明进度条组件,wings="wingsProgressbar"。

* + 1. name

name属性定义工具栏组件的名称，属性值为工具栏组件的唯一标识。

1. 接口详解
   1. Close()

关闭当前对象进度条。$.wingsWidget('progressName').close()。

1. grid表格组件
2. 属性
   1. 表级属性
      1. wings(必填)

固定“WingsGrid”。

* + 1. name(必填)

表格唯一标记，可用$.wingsWidget("name");获取表格操作对象。

* + 1. pageLength

默认每页条数。

* + 1. paging

分页开关。

* + 1. pagingType

分页类型‘numbers’：只显示数字，’simple’：只显示上下页，’simple\_numbers’：显示数字上下页。

* + 1. info

记录数信息开关。

* + 1. ordering

排序开关。

* + 1. lineNumber

索引列开关。

* + 1. lineColumnTitle

索引列名。

* + 1. rowSelect

行可选开关。’ noselect’：不可选，’ radio’：单选，’ multiple’：多选。

* + 1. selectColor

选中行变色。

* + 1. hideMenu

表头隐藏列控制功能开关。

* + 1. hideMenu\_exclude

不添加隐藏列控制功能的列（数组）。

* + 1. hideMenu\_unhide

添加隐藏列控制功能但是不隐藏的列（数组）。

* + 1. border

边框线是否显示。

* + 1. showHeader

显示隐藏表头。

* + 1. itemX

默认列宽。

* + 1. striped

隔行变色开关。

* + 1. nowrap

控制表格内容是否支持换行显示

true：不换行 false：换行

* + 1. contentLen

支持在表级和列级标签上添加。表级上默认全局单元格内容长度，

列级上单列单元格长度，优先度：列级>表级。

* + 1. checkColor

当checkColor设置为true时行变色和复选框将联动显示（点击一行，行变色，checkbox选中）。

* + 1. dataFontSize

设置字体大小支持px和%。

* + 1. fenye

大量数据分页开关

设置成true开启，false关闭

如果设置了ptid属性，为后台分页，初始化方法应返回1页记录信息。（用于超大量数据）

没设置初始化方法返回全部信息，一次性加载到缓存中。

* + 1. ptid（fenye=”true”）

后台分页，表级标签上加ptid为后台分页请求url。初始化方法中返回数据会被放入缓存中。

例：

|  |
| --- |
| 初始化接口示例：  public IResData initPage(IReqData req) throws Exception {  final IResData res = new SwordRes();  final PageQueryValueObject pvo =  req.getPageQueryValueObject();  FenYePageParam fy =  new FenYePageParam(pvo.getFenyePageParam());  Map<String, Object> param =  new HashMap<String, Object>();  fy.initNextQuery(new SQLParam());  fy.setPageSize(10);  fy.setRowCount(3000);  List<Map<String, Object>> grid = getGridFyData();  List<Map<String, Object>> grid1 = getGridData();  res.addTable("leiTable", grid,fy);  res.addTable("leiTable1", new ArrayList());  res.addPage("/wings/core/demoWings/wingsForm/wingsForm001.jsp");  return res;  }  分页请求接口示例：  public IResData getGridFyData1(IReqData req) throws SwordBaseCheckedException {  final IResData res = new SwordRes();  final PageQueryValueObject pvo = req.getPageQueryValueObject();  FenYePageParam fy =  new FenYePageParam(pvo.getFenyePageParam());  fy.initNextQuery(new SQLParam());  Map<String, String> map = null;  List<Map<String, Object>> gridList =  new ArrayList<Map<String, Object>>();  for(int i = 0;i<3000;i++){  Map<String, Object> row =  new HashMap<String,Object>();  row.put("bjid", i);  row.put("xh", i+1);  row.put("name", "张"+i);  row.put("school", "-10.25% ");  row.put("position", "-10.25% ");  row.put("salary", i+2);  row.put("cz", "操作"+i);  row.put("pwd", i+"1234");  row.put("xq", 2);  gridList.add(row);  }  int max = fy.getPageNum()\*fy.getRows();  int min = (fy.getPageNum()-1)\*fy.getRows()+1;  List<Map<String, Object>> fyGirdList =  new ArrayList<Map<String, Object>>();  for(int i = min-1;i<max;i++){  fyGirdList.add(gridList.get(i));  }  res.addTable("fydata", fyGirdList, fy);  return res;  } |

* + 1. fzbtid

在表级标签中添加属性：fzbtid使之与自己画的复杂表头table id相同即可。

例:

|  |
| --- |
| <table id="fzbtid">  <thead>  <tr role="row">  <th class="sorting\_disabled" rowspan="1" colspan="5">学生基本信息</th>  </tr>  <tr role="row">  <th class="sorting\_disabled" rowspan="1" colspan="1" aria-label="" style="width: 40px;"></th>  <th class="wingsDtTdLeft nowrap " tabindex="0" aria-controls="leiTable\_wingsGrid\_48" rowspan="1" colspan="1" aria-label="姓名: activate to sort column ascending" style="width: 100px;">姓名</th>  <th class="wingsDtTdLeft nowrap " tabindex="0" aria-controls="leiTable\_wingsGrid\_48" rowspan="1" colspan="1"  aria-label="学校: activate to sort column ascending"  style="width: 100px;">学校</th>  <th class="wingsDtTdLeft nowrap " tabindex="0" aria-controls="leiTable\_wingsGrid\_48" rowspan="1" colspan="1" aria-label="位置: activate to sort column ascending" style="width: 100px;">位置</th>  <th class="wingsDtTdLeft nowrap " tabindex="0"  aria-controls="leiTable\_wingsGrid\_48" rowspan="1" colspan="1" aria-label="薪资: activate to sort column ascending"style="width: 100px;">薪资</th>  </tr>  </thead  </table>  <input type="button" value="提交" onclick = "sendTo()">  <input type="button" value="增行" onclick = "addRow()">  <input type="button" value="删行" onclick = "deleting()">  <div wings='WingsGrid' name="leiTable" caption='学生列表' panel='true' border="true" rowSelect = "radio" fzbtid=“fzbtid” >  <div name="bjid" type="hidden" ></div>  <div name ="name" caption="姓名" ></div>  <div name="school" caption="学校" contentLen = "15" wingsmouseover="mouseover()" wingsmouseout="wingsmouseout()"></div>  <div name ="position" caption="位置" ></div>  <div name ="salary" caption="薪资">  </div> |

* 1. 列级属性
     1. name

与后台返回json数据对象名绑定字段（不与后台绑定不要使用此属性）。

* + 1. caption

列显示名。

* + 1. contentLen

支持在表级和列级标签上添加,列级上单列单元格长度,优先度：列级>表级。

* + 1. defaultValue

设置默认值。

* + 1. x

列宽。

* + 1. type

类型。

* + - 1. hidden

该列隐藏

* + - 1. text

单元格type设成text单元格可编辑。

* + - 1. a

该列为超链接标签，超链接文本内容和该列caption相同

1. isShowCaption

现在type="a"列级元素可以从后台取值，在a元素上添加name属性，对应后台的数据key即可

在type="a"列级元素上设置isShowCaption = "true"可以让该列内容为captionn，默认为：false 。

1. childonclick

点击事件。

* + - 1. password

密码控件，输入内容变成\*

* + - 1. pulltree

集成wings下拉树控件，详细用法参见wingsTree

* + - 1. select

集成wings下拉列表控件，详细用法参见wingsSelect控件

* + - 1. date

集成wings日期控件，详细用法参见wingsdate控件

1. 接口
   1. getCellEl ()

入参：row\_selector 行选择器（行号）。

col\_selector 列选择器（列号）。

返回值：Cell返回单元格对象（wings对象）。

1. 属性
   1. isNull判断是否为空，空返回true，存在返回false。
2. 接口
   1. getIndex() 获取元素位置。
   2. getObj()获取该元素原生dom对象。
   3. get$Obj（）获取该元素jquery对象。

d）getValue（）获取该元素的值。

* 1. getRowEl()

获取行对象。

入参：row\_selector 行选择器（行号）。

出参：row返回行对象（wings对象）。

1. 属性
   1. isNull判断是为空，空返回true，存在返回false。
2. 接口
   1. getIndex() 获取元素位置。
   2. getObj()获取该元素原生dom对象。
   3. get$Obj（）获取该元素jquery对象。
   4. getValue（colName）获取该元素的值。

参数：列名，不填返回全部。

* 1. getElement（col\_selector）获取该行指定单元格Cell（wings对象）。

1. 参数：col\_selector 列选择器（列号）。
   1. getColEl()

入参：col \_selector 行选择器（行号）。

出参：column返回列对象（wings对象）。

1. 属性

isNull判断是为空，空返回true，存在返回false。

1. 接口

a) getIndex() 获取元素位置。

b) getObj(rowNum)获取该元素原生dom对象。

参数：行号，不填返回全部。

c) get$Obj（rowNum）获取该元素jquery对象。

参数：行号，不填返回全部。

d) getValue（rowNum）获取该元素的值。

参数：行号，不填返回全部。

e) getElement（rowNum）获取该行指定单元格Cell（wings对象）。

参数：行号。

例：

|  |
| --- |
| $(function(){  var name col = $.wingsWidget('leiTable').getColEl(2). get$Obj();  var row = $.wingsWidget('leiTable').getRowEl(2).get$Obj();  var cell = $.wingsWidget('leiTable').getCellEl(1,5).get$Obj();  }) |

* 1. getSelectedRow()

获取选中变色行

* 1. getCheckedRow()

获取checkbox选中行

* 1. setAjaxParam(param)

设置后台分页翻页查询时传入的参数

参数:param:自定义传入参数对象

* 1. getAjaxParam()

获取后台分页翻页查询时传入的参数

例：

|  |
| --- |
| $(function(){  var param ={a:"1",b:"2"};  $.wingsWidget('leiTable').setAjaxParam(param);  }) |

* 1. getAllGridData

获取全部数据（带状态）。

* 1. getStatusGridData

获取带状态（增删改过的）数据。

* 1. getAllNoDeleteGridData

获取非删除数据。

* 1. getAllGridShowData

获取非隐藏数据。

* 1. getCurPageGridData

获取当页数据。

例:

|  |
| --- |
| function sendTo(){  var stu = $.wingsWidget('leiTable');  var data = stu.getAllGridData();  // var data = stu.getStatusGridData();  // var data = stu.getAllNoDeleteGridData();  // var data = stu.getAllGridShowData();  // var data = stu.getCurPageGridData();  var sub = new WingsSubmit();  sub.pushData(data);  sub.setCtrl('WingsFormCtrl\_girdTest');  sub.submit();  } |

* 1. insertRow(dataObj)

参数：dataObj,修改后的数据。

例：

|  |
| --- |
| function addRow(){  var gird = $.wingsWidget('leiTable');  var row =  {'tds':{'bjid':{value:111},  'cz':{value:"操作aaa"},  'name':{value:"ceshi"},  'position':{value:"藏品供应"},  'salary':{value:"111"},  'school':{value:"沈阳一中"},  'pwd':{value:''},  'xq':{value:"3"}  }  };  gird.insertRow(row);  } |

* 1. deleting（rowEls）

参数：rowEls：存放wings行对象数组

例：

|  |
| --- |
| function deleting(){  var gird = $.wingsWidget('leiTable');  // var rows = gird.getCheckedRow();  var rows = gird.getSelectRows();  gird.deleting(rows);  } |

* 1. updateCell(cellW,dataValue,showValue)

参数：cellW：wings行对象

dataValue：提交到后台的值

showValue：页面显示的值

例：

|  |
| --- |
| $(function(){  Var grid = $.wingsWidget('leiTable');  grid.updateCell(grid.getRowEl(1),’1’,’2’);  }) |

* 1. updateRow (rows,dataObj)

参数：rows：修改行（wings行对象数组）

dataObj：修改后的数据

例：

|  |
| --- |
| function updateRow(){  var gird = $.wingsWidget('leiTable');  var row =  {'tds':{'bjid':{value:111},  'cz':{value:"操作aaa"},  'name':{value:"ceshi"},  'position':{value:"藏品供应"},  'salary':{value:"111"},  'school':{value:"沈阳一中"},  'pwd':{value:''},  'xq':{value:"3"},  'csTime': {value:"1989-10-10"}  }  };  var rows = gird.getSelectedRows();  gird.updateRow(rows,row);  } |

* 1. bind()

绑定事件。

参数:obj:要绑定对象（cellEL，rowEl，colEl）。

event：jquery事件。

callback：回调方法。

例:

|  |
| --- |
| $(function(){  var namecol = $.wingsWidget('leiTable').getColEl(2).get$Obj();  var row = $.wingsWidget('leiTable').getRowEl(2).get$Obj();  var cell = $.wingsWidget('leiTable').getCellEl(1,5).get$Obj();  $.wingsWidget('leiTable').bind(cell,'click',function(){  alert(111);  });  }) |

1. 事件
   1. wingsOnclick

将自定义方法绑定wingsOnclick事件，在自定义方法中可以获得所选行信息。

* 1. onRowClick

表级元素上添加此属性，点击行时会触发用户定义事件,回调参数：rowData：该行数据对象（object），rowEl：该行dom对象，点击该事件元素。

例:

|  |
| --- |
| function gridRowOnClick(rowData,rowEl,e){  alert(rowData.school);  } |

* 1. onBeforeCreateCell

列级元素上添加此属性，执行该列创建前执行方法

例：

|  |
| --- |
| function onCreateBefore(renderObj){  renderObj.data = "1989-10-12";  } |

* 1. onAfterCreateRow

表级元素上添加此属性，执行每行行创建后执行的方法

例：

|  |
| --- |
| function onAfterCreateRow(rowEl){  var row =  {'tds':{'bjid':{value:111},  'cz':{value:"操作aaa"},  'name':{value:"ceshi"},  'position':{value:"藏品供应"},  'salary':{value:"111"},  'school':{value:"沈阳一中"},  'pwd':{value:''},  'xq':{value:"3"},  'csTime': {value:"1989-10-10"}  }  };  var grid = $.wingsWidget('leiTable');  grid.updateCell(rowEl.getElement('name'),'lm','lm1');  // grid.updateRow(a,row);  } |

* 1. wingsmouseover

列级元素上添加此属性，鼠标进入该列或该列单元格子元素内会触发用户自定义事件回调参数：e：点击该事件元素。

* 1. wingsmouseout

列级元素上添加此属性，鼠标离开该列或该列单元格子元素内会触发用户自定义事件回调参数：e：点击该事件元素。

* 1. wingsmouseover

列级元素上添加此属性，鼠标进入该列单元格内会触发用户自定义事件回调参数：e：点击该事件元素。

例:

|  |
| --- |
| //js代码  function mouseover(e){  alert(mouseover);  }  function wingsmouseout(e){  alert("wingsmouseout");  }  function wingsmouseout(e){  alert("wingsmouseout");  }  //页面代码  <div wings='WingsGrid' name="leiTable" caption='学生列表' panel='true' border="true" rowSelect = "radio" >  <div name="bjid" type="hidden" ></div>  <div name ="name" caption="姓名" ></div>  <div name="school" caption="学校" contentLen = "15" wingsmouseover="mouseover()"  wingsmouseout="wingsmouseout()" wingsmouseenter="wingsmouseenter()"></div>  <div name ="position" caption="位置" ></div>  <div name ="salary" caption="薪资"></div>  </div> |

1. 提交组件
2. 属性
   1. 必填属性
      1. wings

使用<div>标签的sword属性声明提交组件，wings="WingsSubmit",此组件可用来向后台提交表单、表格等组件的数据。

* + 1. name

name属性定义提交组件的名称，属性值为提交组件的唯一标识。在客户端，JavaScript通过name属性值获取提交组件对象。

* 1. 非必填属性

1. 1. 1. btnName

用来设置提交按钮的显示名称。

1. 事件
   1. onclick

点击提交组件按钮时触发此事件

* 1. onSuccess

提交成功触发此事件。此事件和服务器交互时使用，当存在返回状态时触发事件。

1. 接口详解
   1. 提交方式（默认Ajax提交）
      1. Ajax提交

Ajax提交，页面不刷新。

例:

|  |
| --- |
| function wingsSubmit\_form(){  var form = $.wingsWidget('studentForm');  var sub = new WingsSubmit();  sub.pushData("hidStr123",form.getValue('hidStr'));  sub.pushData(form.getSubmitData());  sub.setCtrl('WingsFormCtrl\_formTest');  sub.submit();  } |

* + 1. 表单提交

表单提交页面会刷新，表单提交需要在Ajax提交的基础上设置WingsSubmit对象的options.postType=‘form’。

例:

|  |
| --- |
| function wingsSubmit\_form(){  var form = $.wingsWidget('studentForm');  var sub = new WingsSubmit();  //默认ajax提交 设置这个后，是from刷新页面的提交方式  sub.options.postType ='form';  sub.pushData("hidStr123",form.getValue('hidStr'));  sub.pushData(form.getSubmitData());  sub.setCtrl('WingsFormCtrl\_formTest');  sub.submit();  } |

* 1. pushData

3. 2. 1. 传递单值

例：

|  |
| --- |
| function wingsSubmit\_form(){  var form = $.wingsWidget('studentForm');  var sub = new WingsSubmit();  //默认ajax提交 设置这个后，是from刷新页面的提交方式  sub.options.postType ='form';  sub.pushData("hidStr123",form.getValue('hidStr'));  sub.setCtrl('WingsFormCtrl\_formTest');  sub.submit();  } |

* + 1. 传递整个form

例:

|  |
| --- |
| function wingsSubmit\_form(){  var form = $.wingsWidget('studentForm');  var sub = new WingsSubmit();  //默认ajax提交 设置这个后，是from刷新页面的提交方式  sub.options.postType ='form';  sub.pushData(form.getSubmitData());  sub.setCtrl('WingsFormCtrl\_formTest');  sub.submit();  } |

* + 1. 传递整个grid

例:

|  |
| --- |
| function save(){  var stu = $.wingsWidget('testGrid');  var data = stu.getAllGridData();  var sub = new WingsSubmit();  sub.pushData(data);  sub.setCtrl('GridTestCtrl\_testData');  sub.submit();  } |

* 1. setFunction

动态设置提交组件的事件

例：

|  |
| --- |
| function e(){  var sub = new WingsSubmit();  //ajax自己的异常  sub.setFunction("onError","WingsSubmit\_onError()");  sub.setFunction("onException",  "WingsSubmit\_onException()"); //sword 响应的异常  }  function WingsSubmit\_onError(){  alert('ajax异常');  }  function WingsSubmit\_onException(req){  alert('报sword异常了');//异常处理  } |

* 1. setOptions

设置submit的一种方式。

例：

|  |
| --- |
| var sub = new WingsSubmit();  sub.setOptions({  async:"true",//选填，默认true  //选填 可用 sub.setCtrl('WingsFormCtrl\_formTest');  ctrl:"WingsFormCtrl\_formTest222",  onSuccess: function(res){  var temp = res.getAttr("leiArr");  },  onError:function(){  alert('ajax异常');  },  onException:function(req){  alert('报sword异常了');  }  }); |

1. 树组件
2. 属性
   1. 必填属性
      1. wings

使用<div>标签的wings属性声明树组件，wings="wingsTree "。

* + 1. name

name属性定义树组件的名称，属性值为树组件的唯一标识。

* 1. 非必填属性
     1. treeType

定义树组件的类型。

0：无选择框普通树(默认值)

1：单选。

2：复选。

* + 1. css

设置内联样式，相当于自定义style。

例：

|  |
| --- |
| <div name="leiTree" type="pulltree" treeType ="1"  css="font-color:red"></div> |

* + 1. defValue

defValue属性用来设置默认值。

* + 1. inputClass

inputClass属性用来定义pullTree的类，相当于class属性。

* + 1. height

设置高度，超出高度 自动出下拉滚动条。

例：

|  |
| --- |
| <div name="leiTree" type="pulltree" treeType = '1' onNodeClick="alert(123);  onCheckBefore="onCheckBefore\_fn();" height='150' ></div> |

1. 事件
   1. onclick

单击时触发此事件。

* 1. onCheckBefore

选中前触发事件。

* 1. onLtidBefore

执行ltid之前触发，方便向后台传递参数，可自定义JSON属性，默认是code caption pcode。主要作用是懒加载。

* 1. onNodeClick

点击事件。

例:

|  |
| --- |
| <div name="leiTree" type="pulltree" treeType = '2' onNodeClick="alert(123);"></div> |

1. 接口详解
   1. initData()

初始化树方法。

例：

|  |
| --- |
| var tree = $.wingsWidget(‘orgTree’);  var options = {data:znodes,treeType:0}  tree.initData(data);  znodes数据结构如下：  {code:1,pcode:0,caption:’第一个节点名字’} |

* 1. setValue()

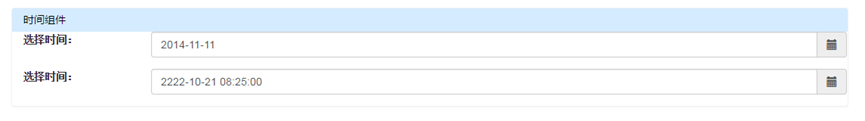
设置节点的code值。

例:

|  |
| --- |
| tree.setValue(codes);  codes数据结构:可传入多个节点，使用逗号分开，如 “1,1.1,2,2.1,2.1” |

* 1. getValue()

getValue()获取已选节点的code值。

1. 日期组件
2. 效果图
3. 属性
   1. 必填属性
      1. wings

使用<div>标签的wings属性声明日期组件，wings="WingsCalendar"

* + 1. name

name属性定义日期组件的名称，属性值为日期组件的唯一标识。

* + 1. type

type属性定义表单或表格组件中的元素为日期组件，type=*"*wingsDate*"*。

* 1. 非必填属性
     1. startDate/endDate

开始日期/结束时间, 最早/最后的日期,所有早期的日期/后期日期将被禁用。

* + 1. format

定义日期格式，如format="yyyy-mm-dd"。日期格式可以为 p, P, h, hh, i, ii, s, ss, d, dd, m, mm, M, MM, yy, yyyy 的任意组合。

* + 1. inputClass

inputClass属性用来定义时间组件的类型，相当于class。

* + 1. css

css属性定义元素样式。

* + 1. defValue

defValue属性定义元素的默认值。

* + 1. disable

disable属性标识text元素为禁用状态。

false(默认值)---text元素为可编辑状态。

true---text元素为禁用状态。

* + 1. wingsReadonly

wingsReadonly属性标识日期为只读，不可进行更改删除等操作。

false(默认值)---date元素为可编辑状态。

true---date元素为只读状态。

* + 1. startDate

定义开始时间，该日期前的日期将被禁用，只能选择开始时间之后的时间。

* + 1. endDate

定义结束时间，该日期以后的日期将被禁用,只能选择结束时间之前的时间。

* + 1. 校验

日期组件关于校验包含required,zdyRule。具体用法参见校验组件一节。

1. 事件

4. 1. onDateChange

日期改变时触发。

1. 弹出框
2. 提示框

包括正确提示和错误提示。

* 1. 效果图
  2. 使用方法

WingsDialog.alert(title,content,callback);

* 1. 示例

|  |
| --- |
| WingsDialog.alert("提示","提示信息！",{  onhidden:function(){  alert("关闭");  }  }); |

1. 确认框
   1. 效果图

|  |
| --- |
| IMG_257 |

* 1. 使用方法

WingsDialog.confirm(title, prompt,callback)

* 1. 示例

|  |
| --- |
| WingsDialog.confirm('提示', '确定继续操作吗?', **function**(result){  **if**(result) {                  alert('是');              }**else** {                  alert('否');              }          }); |

* 1. 方法

WingsDialog.alertRight()

WingsDialog.alertWrong()

1. 模态窗口
   1. 效果图

|  |
| --- |
| IMG_258 |

* 1. 使用方法

WingsDialog.showModalDialog(jsonObject);

* 1. 示例

|  |
| --- |
| WingsDialog.showModalDialog({  title:'这里是标题',  url:'/sword?ctrl=WingsFormCtrl\_initPage',  height=’800’,  width=’900’,  param:{‘data1’:data1,’data2’:data2},  onshown:function(dialog){  alert('显示完dialog');  },  onhidden:function(dialog){  alert('关闭后dialog');  }  }); |

* 1. 属性

json格式数据，包括以下属性：

title:标题

url: 请求页面访问地址

width:宽度

height:高度

onshown: 窗口显示完成回调函数

onhidden:窗口关闭完成回调函数

* 1. 方法

WingsDialog.closeModalDialog();模态窗口增加关闭窗口方法

parent.WingsDialog.closeModalDialog();在子页面中调用的关闭方法

1. Tab组件
2. 属性
   1. 必填属性
      1. wings

使用<div>标签的wings属性声明tab组件，wings="WingsTab"。

* + 1. name

name属性定义Tab切换组件的名称，属性值为Tab切换组件的唯一标识.

* + 1. tabTitles

定义Tab切换组件每个tab页的标题.

* + 1. tabDivs

定义Tab切换组件要添加的div元素的id属性值

* 1. 非必填属性
     1. tabSelectedIndex

定义默认显示第几个tab页。

* + 1. inputClass

设置tab样式，现提供两种

C:\Users\3\AppData\Roaming\Tencent\Users\489614500\QQ\WinTemp\RichOle\{@F4U(90SLZ5WAY$]IIC5Q5.pngone（默认值）

C:\Users\3\AppData\Roaming\Tencent\Users\489614500\QQ\WinTemp\RichOle\0CXRXBC5}$M[`F9%[2O)(K8.pngtwo

1. 公共方法
2. $chk()

校验是否为空。

|  |
| --- |
| if($chk(temp)){ } |

1. $.wingsWidget()

根据name属性值获取整个页面对象。

例：

|  |
| --- |
| //获取页面form表单对象  var form = $.wingsWidget('studentForm');  //获取页面toolbar对象。  var toolbar = $.wingsWidget("toolbar"); |

1. 初始化组件
2. 属性
   1. wings

使用<div>标签的wings属性声明初始化组件，wings="PageInit"。

1. 事件
   1. onSuccess

当容器成功初始化数据【后台没有抛出异常】的时候触发。

例:

|  |
| --- |
| <div wings='PageInit' onSuccess="initSuccess();"></div>  //页面初始化方法  function initSuccess(res){  var temp = res.getAttr("leiArr");  alert(temp);  } |

1. 校验组件
2. 校验方法

例：

|  |
| --- |
| var form = $.wingsWidget('studentForm');  var flag = form.formValid();//通过返回true 不通过返回false  并在不通过位置，弹出黄色提示框 |

1. 设置方法
   1. text

required="true" 必须

rangelength="6,12" 6~ 12位

maxnb = '10' 最大位数

minnb = '2' 最小位数

rule = 'email' 格式校验

* 1. wingsPwd

required="true" 必须

rangelength="6,12" 6~ 12位

equalTo="ywm2" 要和和XX一样

例：

|  |
| --- |
| <div name="ywm2" type="wingsPwd" required="true" rangelength="6,12"></div>  <div name="ywm3" type="wingsPwd" required="true" equalTo="ywm2" ></div> |

* 1. wingsData

required="true" 必须

* 1. textarea

required="true" 必须

rangelength="6,12" 6~ 12位

maxnb = '10' 最大位数

minnb = '2' 最小位数

rule = 'email' 格式校验

1. rule属性

email：电子邮箱

number：数字

numberInt：数字类-整型

numberFloat：数字类-浮点型

numberScience：数字类-科学计数法

english：字符类-英文

chinese：字符类-中文

character：字符类

numbercharacter：数字汉字加英文字母类

qq：qq类

telephone：固定电话

cellphone：手机号码

idcard：身份证号

postal：邮政编码

currency：货币格式

url：url连接

date：日期类

1. 左右选择器
2. 效果图
3. 示例

|  |
| --- |
| <%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>  <!doctype html>  <html>  <head>  <meta charset="UTF-8">  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <title>左右选择器</title>  <!—引入基础js和左右选择器js-->  <jsp:include page="/wings/core/common/wingsInclude.jsp"></jsp:include>  <jsp:include page="/wings/core/common/wingsMoveInclude.jsp"></jsp:include>  <script>  $(function() {  var data = [ {  code : 0,  caption : '大学',  url : 'www.baidu.com',  memo : '说明'  }, {  code : 1,  caption : '中学'  }, {  code : 2,  caption : '小学'  }, {  code : 3,  caption : 'invalid'  }, {  code : 4,  caption : 'wontfix'  } ]  var move = $.wingsWidget('tax2');  move.initData({}, data)  });  </script>  </head>  <body>  <div id="wrap">  <h4 id="demo-undo-redo">左右移动选择器</h4>  <div>  <div wings="wingsMove" name='tax2'>  <table style="width: 100%">  <tr>  <td style="width: 45%">  <!—左侧选择框-->  <div wings='wingsFramePanel' caption='左侧选择'>  <div type='wingsMoveLeft'></div>  </div>  </td>  <td align="center" style="width: 10%">  <!—按钮-->  <div type='wingsMoveButton'>  <div name="allRight" type="allRight"></div>  <div name="right" type="right"></div>  <div name="left" type="left"></div>  <div name="allLeft" type="allLeft"></div>  <div name="undo" type="undo"></div>  <div name="redo" type="redo"></div>  </div>  </td>  <td style="width: 45%">  <!—右侧选择框-->  <div wings='wingsFramePanel' caption='右侧选择'>  <div type='wingsMoveRight'></div>  </div>  </td>  </tr>  </table>  </div>  </div>  </div>  </body>  </html> |

1. 属性
   1. wings（必填）

wings='wingsMove' 用来声明左右选择器组件。

* 1. name（必填）

name属性定义Tab切换组件的名称，属性值为Tab切换组件的唯一标识。

1. 方法
   1. initData ()

initData()初始化数据方法 注：俩个参数一个左侧数据一个右侧数据。

例：

|  |
| --- |
| var move = $.wingsWidget('tax2');  move.initData({}, data) |

* 1. getLeftInsertData()/getRightInsertData()

获取左侧/右侧新增数据。(在初始化数据基础上新增的数据)。

例：

|  |
| --- |
| var move = $.wingsWidget('tax2');  move.getRihtInsertData()； |

* 1. getLeftDeleteData()/getRightDeleteData()

获取左侧/右侧删除数据(在初始化数据基础上删除的数据)。

例：

|  |
| --- |
| var move = $.wingsWidget('tax2');  move. getLeftDeleteData ()； |

1. 与后台交互

左右选择器 与后台交互 支持页面submit组件提交，和后台 addtable回传。

例：

|  |
| --- |
| 页面：  <div type = 'wingsMoveLeft' dataName='move\_left' showName="name" codeName="position"></div>  <div type = 'wingsMoveRight' dataName='move\_right' showName="name" codeName="position">  js:  $("#submitTest").click(function(){  var myMove = $.wingsWidget('tax2')  var sub = new WingsSubmit();  //默认ajax提交 设置这个后，是from刷新页面的提交方式  sub.pushData(myMove.getLeftSubmitData());//左  sub.pushData(myMove.getRightSubmitData());//右  sub.setCtrl('WingsFormCtrl\_wingsMoveSubmitTest');  sub.submit();  }) |

1. 提交后初始化

例:

|  |
| --- |
| 页面:  <div type = 'wingsMoveLeft' dataName='move\_left' ></div>  <div type = 'wingsMoveRight' dataName='move\_right' ></div>  java:  //左侧  List<Map<String, Object>> grid = getGridData();  res.addTable("move\_left", grid);//  //右侧  List<Map<String, Object>> grid1 = getGridData();  res.addTable("move\_right", grid1);// |

1. 搜索建议组件
2. 示例

|  |
| --- |
| <%@ page language="java" contentType="text/html; charset=UTF-8"  pageEncoding="UTF-8"%>  <!doctype html>  <head>  <meta charset="utf-8">  <title>Bootstrap 搜索建议插件</title>  <jsp:include page="/wings/core/common/wingsInclude.jsp">  </jsp:include>  <!-- 本页js -->  <script src="/wings/core/demo/wingsSuggest/suggestTest.js">  </script>  </head>  <body>  <div wings='PageInit' onSuccess="initSuccess();"></div>  <h3>测试wingsSuggest</h3>  <p>各种功能</p>  <p id="methodTest">  <button class="btn btn-success btn-success-outline"  onclick = "sugDisable();" >disable</button>  <button class="btn btn-success btn-success-outline"  onclick = "sugEnable();" >enable</button>  <button class="btn btn-success btn-success-outline"  onclick = "sugShow();">show</button>  <button class="btn btn-success btn-success-outline"  onclick = "sugHide();">hide</button>  </p>  <div class="row">  <div class="col-lg-6" >  <div wings="wingsSuggest" name = "leiSuggest"  lctrl="WingsSuggestCtrl\_suggestTest" rowName = "name"></div>  </div>  </div>  <h3>例子二</h3>  <div class="row">  <div class="col-lg-6" >  <div wings="wingsSuggest" name = "leiSuggest002"  lctrl="WingsSuggestCtrl\_suggestTest"rowName = "id,name,school" keyField="name" showBtn="false" showHeader ="true"  headerTitles = "{id:'序号',name:'姓名',school:'学校'}">  </div>  </div>  </div>  </div>  </body>  </html> |

1. 属性
   1. 必填属性
      1. wings

使用<div>标签的wings属性声明搜索建议组件,wings="wingsSuggest"。

* + 1. name

name属性定义搜索建议组件的名称，属性值为搜索建议组件的唯一标识。

* + 1. lctrl

lctrl属性定义搜索建议组件访问后台的ctrl，即 数据来源。

* + 1. rowName

rowName属性定义显示哪些字段。以及显示顺序，rowName定义的显示字段的值须与后台对应。rowName的值可以为单个字段，也可以为多个字段，当结果为多个字段时，用“，”隔开。

例：

|  |
| --- |
| 页面：  <!—声明一个搜索建议组件-->  <div wings="wingsSuggest" name = "leiSuggest"  lctrl="WingsSuggestCtrl\_suggestTest"  rowName = "name" ></div>  ctrl:  //设置数据，由于页面中rowName = "name",后台数据中row.put("name", i)，rowName属性的值必须是后台传入的数据包含的。  public List<Map<String, Object>> getGridData() {  Map<String, String> map = null;  List<Map<String, Object>> gridList =  new ArrayList<Map<String, Object>>();  for(int i = 0;i<5;i++){  Map<String, Object> row =  new HashMap<String,Object>();  row.put("code", i);  row.put("caption", "昌平一中" + i);  row.put("bjid", i);  row.put("name", i);  row.put("school", "昌平一中");  row.put("position", i+1);  row.put("salary", i+2);  gridList.add(row);  }  return gridList;  } |

* + 1. keyField（当rowName的值为多个时）

keyField属性用来定义选中rowName的哪个字段。

* 1. 非必填属性
     1. showBtn

showBtn属性用来定义是否显示下拉按钮。

true(默认值)—显示,false--不显示。

* + 1. showHeader,headerTitles

showHeader属性用来定义是否显示表头。

true—显示,false(默认值)--不显示。

headerTitles属性用来定义有效字段别名。当showHeader为false时，headerTitles属性无效，headerTitles默认显示后台传过来的标题。

例：

|  |
| --- |
| //页面代码  <!—声明一个搜索建议组件，设置headerTitles="{id:'序号',name:'姓名',school:'学校'}，表头显示序号，姓名和学校，如果不设置则默认显示id，name，school-->  <div wings="wingsSuggest" name = "leiSuggest"  lctrl="WingsSuggestCtrl\_suggestTest"  rowName= "id,name,school" keyField="name" showBtn="false" showHeader ="true" headerTitles="{id:'序号',name:'姓名',school:'学校'}">  </div> |

1. 事件
   1. onLtidBefore

在执行lctrl的方法触发。

1. 方法
   1. disable()/enable()

设置搜索建议组件是否可用。

例：

|  |
| --- |
| //页面代码  <p id="methodTest">  <button class="btn btn-success btn-success-outline"  onclick = "sugDisable();" >disable</button>  <button class="btn btn-success btn-success-outline"  onclick = "sugEnable();" >enable</button>  </p>  <div wings="wingsSuggest" name = "leiSuggest"  lctrl="WingsSuggestCtrl\_suggestTest"  rowName= "id,name,school" keyField="name" showBtn="false" showHeader ="true" headerTitles="{id:'序号',name:'姓名',school:'学校'}">  </div>  js代码  var sug = $.wingsWidget('leiSuggest');  function sugDisable(){  var sug = $.wingsWidget('leiSuggest');  //设置无效  sug.disable();  }  function sugEnable(){  var sug = $.wingsWidget('leiSuggest');  //设置有效  sug.enable();  } |

* 1. show()/hide()

设置搜索建议组件显示/隐藏

例：

|  |
| --- |
| 页面代码：  <p id="methodTest">  <button class="btn btn-success btn-success-outline"  onclick = "sugShow();">show</button>  <button class="btn btn-success btn-success-outline"  onclick = "sugHide();">hide</button>  </p>  <div wings="wingsSuggest" name = "leiSuggest"  lctrl="WingsSuggestCtrl\_suggestTest"  rowName= "id,name,school" keyField="name" showBtn="false" showHeader ="true" headerTitles="{id:'序号',name:'姓名',school:'学校'}">  </div>  js代码  function sugShow(){  var sug = $.wingsWidget('leiSuggest');  //展开  sug.show();  }  function sugHide(){  var sug = $.wingsWidget('leiSuggest');  //隐藏  sug.hide();  } |

1. 前后台交互
   1. 后台获取

req.getAttr("wingsSuggestName");

例：

|  |
| --- |
| 页面代码：  <!—声明一个搜索建议组件-->  <div wings="wingsSuggest" name = "leiSuggest"  rowName="name"l ctrl="WingsSuggestCtrl\_suggestTest" onLtidBefore ="onLtidBefore\_fn();" ></div>  ctrl代码：  public IResData suggestTest(final IReqData req) throws Exception {  final IResData res = new SwordRes();  //获取name为"leiSuggest"的值  String keyword = (String)req.getAttr("leiSuggest");  return res;  } |

* 1. 后台到前台

res.addTable("任意值", 结果集);