**E9流程表单前端接口API**

**1.说明**

**1.1 简介**

*所有接口统一封装在全局对象window.WfForm中*

*部分接口存在使用范围，最低kb版本以及是否移动端/PC端独有。没有特殊注明情况下通用*

*表单字段相关操作，不推荐使用jQuery，禁止原生JS直接操作DOM结构！*

*大家开发过程中，推荐都使用API接口操作，由产品统一运维；同时使用API才能完整的兼容移动终端*

**1.2 移动端兼容**

*WfForm对象下接口，兼容新版移动端EM7*

*由于API接口在PC端与移动端已经统一，为减少开发工作量以及后期维护成本；
故EM7表单在移动终端不再引入workflow\_base表custompage4emoble列作为自定义页面，直接引入custompage列(与PC模板一致)作为自定义页面*

*前端（JS方法）区分终端：*

可通过方法WfForm.isMobile()判断是否移动端

1. var isMobile = WfForm.isMobile(); //true表示是eMobile、微信、钉钉等移动终端，false代表PC端

*后端请求（自定义页面等）区分终端:*

1. boolean isMobile = "true".equals(request.getParameter("\_ec\_ismobile")); //true表示是eMobile、微信、钉钉等移动终端，false代表PC端

**1.3 前端代码开发方式**

*方式1：模板上代码块，针对单个节点，在显示/打印/移动模板单独配置*

*方式2：【路径管理】-打开具体路径-【基础设置】-【自定义页面】，针对此路径下所有节点所有模板生效*

*方式3：【路径管理】-【应用设置】-【流程表单自定义页面设置】，针对系统所有非模板模式的场景(PC及移动)。注意此页面为全局custompage，应避免写ready、checkCustomize等全局函数，只定义些函数体*

*特别注意：方式二、方式三禁止引入init\_wev8.js。*

*如遇配置不生效，请先将代码块/custompage仅写alert确认是否生效，再逐步排查错误原因。*

**1.4 PC端打开表单的方式**

*新建请求：传参路径id，会自动计算活动版本的路径id*

1. window.open("/workflow/request/CreateRequestForward.jsp?workflowid=747");

*查看请求：传参请求id，用户需本身具备此请求查看权限，主次账号需带入账号信息*

1. window.open("/workflow/request/ViewRequestForwardSPA.jsp?requestid=5963690");

**1.5 移动端打开表单的方式**

*移动端表单链接：*

1. *//新建链接，传参路径id*
2. *var createUrl = "/spa/workflow/static4mobileform/index.html#/req?iscreate=1&workflowid=747";*
3. *//查看链接，传参请求id*
4. *var viewUrl = "/spa/workflow/static4mobileform/index.html#/req?requestid=4503066";*

*第一种方式(推荐)：调用封装好的方法
如果是通过移动端脚手架打包的模块，可以直接调用
如果是自行开发的界面，需要引入/spa/coms/openLink.js*

*最低支持版本：KB900190601*

*openLink.openWorkflow(url, callbackFun, returnUrl)*

| **参数** | **参数类型** | **说明** |
| --- | --- | --- |
| url | String | 打开表单的链接 |
| callbackFun | Function | 仅限EM客户端，返回时的回调函数 |
| returnUrl | String | 非EM客户端，返回/提交后到指定链接 |

1. window.openLink.openWorkflow(createUrl, function(){
2. alert("E-mobile打开表单链接，返回或提交后触发此回调函数");
3. });
4. //非EM终端打开，返回或提交后返回到流程中心界面
5. window.openLink.openWorkflow(createUrl, null, "/spa/workflow/static4mobile/index.html#/center/doing");

*第二种方式：仅限于EM客户端，打开表单并可控制表单返回/提交后事件回调
使用EM-SDK，弹webview方式实现*

1. //分为两步，第一步调用SDK弹webview，第二步调用SDK控制回调刷新
2. window.em.openLink({
3. url: viewUrl,
4. openType: 2
5. });
6. window.em.ready(function(){
7. window.em.registerBroadcast({
8. name: "\_closeWfFormCallBack",
9. action: function (argument) {
10. alert("E-mobile打开表单链接，返回或提交后触发此回调函数");
11. }
12. });
13. });

*第三种方式(不推荐)： window.open或者window.location.href跳转。
此方式url需要传参returnUrl并转码，流程提交后需要关闭的情况会返回跳转到returnUrl地址上，无法监听打开-手动返回场景；
建议如果是EM客户端采用方式一或方式二！*

1. window.open(viewUrl+"&returnUrl="+window.encodeURIComponent("/test.jsp?param1=test11&param2=test22"));

**2.注册自定义事件**

**2.1 注册拦截事件，指定动作执行前触发，并可阻断/放行后续操作**

*支持多次注册，按注册顺序依次执行；支持异步ajax，避免请求卡住*

*场景1：表单提交、保存、退回、转发、强制收回等操作执行前，执行自定义逻辑并阻断/放行后续操作*

*场景2：明细添加行、删除行前，执行自定义逻辑并阻断/允许后续操作*

| **动作类型** | **说明** | **最低版本要求** |
| --- | --- | --- |
| WfForm.OPER\_SAVE | 保存 |  |
| WfForm.OPER\_SUBMIT | 提交/批准/提交需反馈/不需反馈等 |  |
| WfForm.OPER\_SUBMITCONFIRM | 提交至确认页面，如果是确认界面，点确认触发的是SUBMIT |  |
| WfForm.OPER\_REJECT | 退回 |  |
| WfForm.OPER\_REMARK | 批注提交 |  |
| WfForm.OPER\_INTERVENE | 干预 |  |
| WfForm.OPER\_FORWARD | 转发 |  |
| WfForm.OPER\_TAKEBACK | 强制收回 |  |
| WfForm.OPER\_DELETE | 删除 |  |
| WfForm.OPER\_ADDROW | 添加明细行，需拼明细表序号 |  |
| WfForm.OPER\_DELROW | 删除明细行，需拼明细表序号 |  |
| WfForm.OPER\_PRINTPREVIEW | 打印预览 | KB900190501 |
| WfForm.OPER\_EDITDETAILROW | 移动端-编辑明细 | KB900191101 |
| WfForm.OPER\_BEFOREVERIFY | 校验必填前触发事件 | KB900191201 |
| WfForm.OPER\_TURNHANDLE | 转办 | KB900201101 |
| WfForm.OPER\_ASKOPINION | 意见征询 | KB900201101 |
| WfForm.OPER\_TAKFROWARD | 征询转办 | KB900201101 |
| WfForm.OPER\_TURNREAD | 传阅 | KB900201101 |
| WfForm.OPER\_FORCEOVER | 强制归档 | KB900201101 |
| WfForm.OPER\_BEFORECLICKBTN | 点右键按钮前 | KB900201101 |
| WfForm.OPER\_SAVECOMPLETE | 保存后页面跳转前 | KB900210501 |
| WfForm.OPER\_WITHDRAW | 撤回 | KB900201101 |
| WfForm.OPER\_CLOSE | 页面关闭 | KB900201101 |

接口名称及参数说明

*registerCheckEvent:function(type,fun)*

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| type | String | 是 | 动作类型(详见上表)，多个逗号分隔 |
| fun | Function | 是 | 自定义函数，此函数入参为callback，执行自定义逻辑完成或异步ajax的success函数体内，放行需调用callback，不调用代表阻断后续操作 |

样例

1. jQuery().ready(function(){
2. WfForm.registerCheckEvent(WfForm.OPER\_SAVE, function(callback){
3. jQuery("#field27495").val("保存自动赋值");
4. callback(); //继续提交需调用callback，不调用代表阻断
5. });
6. WfForm.registerCheckEvent(WfForm.OPER\_SAVE+","+WfForm.OPER\_SUBMIT,function(callback){
7. //... 执行自定义逻辑
8. callback();
9. });
10. WfForm.registerCheckEvent(WfForm.OPER\_ADDROW+"1", function(callback){
11. alert("添加明细1前执行逻辑，明细1则是OPER\_ADDROW+1，依次类推")
12. callback(); //允许继续添加行调用callback，不调用代表阻断添加
13. });
14. WfForm.registerCheckEvent(WfForm.OPER\_DELROW+"2", function(callback){
15. alert("删除明细2前执行逻辑")
16. callback(); //允许继续删除行调用callback，不调用代表阻断删除
17. });
18. WfForm.registerCheckEvent(WfForm.OPER\_PRINTPREVIEW, function(callback){
19. alert("控制默认弹出的打印预览窗口")
20. alert("当打印含签字意见列表，此接口需要放到跳转路由前执行，组件库提供此机制");
21. window.WfForm.printTimeout = 3000; //产品是默认延时1s自动弹出，可通过此方式控制延时时间
22. callback(); //允许继续弹出调用callback，不调用代表不自动弹预览
23. });
24. WfForm.registerCheckEvent(WfForm.OPER\_EDITDETAILROW, function(callback,params){
25. alert(JSON.stringify(params)); //参数含当前点击哪个明细表哪一行
26. callback(); //允许跳转明细编辑窗口，不调用阻断跳转
27. });
28. });

**2.2 注册钩子事件，指定动作完成后触发**

*支持多次调用注册，按注册的先后顺序依次执行*

| **类型** | **说明** | **最低版本要求** |
| --- | --- | --- |
| WfForm.ACTION\_ADDROW | 添加明细行，需拼明细表序号 | KB900190407 |
| WfForm.ACTION\_DELROW | 删除明细行，需拼明细表序号 | KB900190407 |
| WfForm.ACTION\_EDITDETAILROW | 移动端-编辑明细行，需拼明细表序号 | KB900190501 |
| WfForm.ACTION\_SWITCHDETAILPAGING | 切换明细分页 | KB900191201 |
| WfForm.ACTION\_SWITCHTABLAYOUT | 切换模板布局标签页 | KB900191201 |

接口名称及参数说明

*registerAction: function(actionname, fn)*

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| actionname | String | 是 | 动作类型，详见上表 |
| fn | Function | 是 | 触发事件 |

样例

1. WfForm.registerAction(WfForm.ACTION\_ADDROW+"1", function(index){
2. alert("添加行下标是"+index);
3. }); //下标从1开始，明细1添加行触发事件，注册函数入参为新添加行下标
4. WfForm.registerAction(WfForm.ACTION\_DELROW+"2", function(arg){
5. alert("删除行下标集合是"+arg.join(","));
6. }); //下标从1开始，明细2删除行触发事件
7. WfForm.registerAction(WfForm.ACTION\_SWITCHDETAILPAGING, function(groupid){
8. alert("切换明细表"+(groupid+1)+"的页码触发事件");
9. });
10. WfForm.registerAction(WfForm.ACTION\_SWITCHTABLAYOUT, function(tabid){
11. alert("切换到标签项"+tabid+"触发事件");
12. });

**3.字段基础操作接口(不适用附件、位置字段类型)**

**3.1 将字段名称转换成字段id**

*灵活运用此方法，可实现多表单、多环境，代码块通用；解耦代码块中指定fieldid*

*convertFieldNameToId: function(fieldname,symbol,prefix)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldname | String | 是 | 字段名称 |
| symbol | String | 否 | 表单标示，主表(main)/具体明细表(detail\_1),默认为main |
| prefix | Boolean | 否 | 返回值是否需要field字符串前缀，默认为true |

样例

1. var fieldid = WfForm.convertFieldNameToId("zs");
2. var fieldid = WfForm.convertFieldNameToId("zs\_mx", "detail\_1");
3. var fieldid = WfForm.convertFieldNameToId("zs\_mx", "detail\_1", false);

**3.2 获取单个字段值**

*getFieldValue: function(fieldMark)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |

样例

1. var fieldvalue = WfForm.getFieldValue("field110");

**3.3 修改单个字段值（不支持附件类型）**

*此方法修改的字段如果涉及到触发联动、单元格格式化等，修改完值会自动触发联动/格式化*

*改值的格式在添加明细行初始化、批量修改字段等场景类同*

*changeFieldValue: function(fieldMark, valueInfo)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| valueInfo | JSON | 是 | 字段值信息，非浏览按钮字段格式为{value:”修改的值”};specialobj为浏览按钮信息，数组格式;showhtml属性只在单行文本类型且只读情况下生效； |

样例

1. //修改文本框、多行文本、选择框等字段类型
2. WfForm.changeFieldValue("field123", {value:"1.234"});
3. //修改浏览框字段的值，必须有specialobj数组结构对象
4. WfForm.changeFieldValue("field11\_2", {
5. value: "2,3",
6. specialobj:[
7. {id:"2",name:"张三"},
8. {id:"3",name:"李四"}
9. ]
10. });
11. //修改check框字段(0不勾选、1勾选)
12. WfForm.changeFieldValue("field123", {value:"1"});
13. //针对单行文本框字段类型，只读情况，支持显示值跟入库值不一致
14. WfForm.changeFieldValue("field123", {
15. value: "入库真实值",
16. specialobj: {
17. showhtml: "界面显示值"
18. }
19. });

**特别注意**：

1. 后台字段如果设置的是只读属性，changeFieldValue修改的字段值在非创建时刻是禁止入库的，属于篡改数据。
2. 此情况需要设置为可编辑属性，如果前台界面又想显示成只读效果，同时设置禁止手工编辑即可。
3. 不适用于附件字段类型。

**3.4 改变单个字段显示属性(只读/必填等)**

*changeFieldAttr: function(fieldMark, viewAttr)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| viewAttr | int | 是 | 改变字段的状态，1：只读，2：可编辑，3：必填，4：隐藏字段标签及内容，5:隐藏字段所在行(行内单元格不要存在行合并) |

样例

1. WfForm.changeFieldAttr("field110", 1); //字段修改为只读
2. WfForm.changeFieldAttr("field110", 4); //字段标签以及内容都隐藏，效果与显示属性联动隐藏一致，只支持主表字段

**3.5 同时修改字段的值及显示属性**

*changeSingleField: function(fieldMark, valueInfo, variableInfo)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| valueInfo | JSON | 否 | 字段值信息，与接口2格式一致，例：{value:”修改的值”} |
| variableInfo | JSON | 否 | 变更属性，例：{viewAttr:3} |

样例

1. WfForm.changeSingleField("field110", {value:"修改的值"}, {viewAttr:"1"}); //修改值同时置为只读

**3.6 批量修改字段值或显示属性**

*changeMoreField: function(changeDatas, changeVariable)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| changeMoreField | JSON | 是 | 修改的字段值信息集合 |
| changeVariable | JSON | 否 | 修改的字段显示属性集合 |

样例

1. WfForm.changeMoreField({
2. field110:{value:"修改后的值"},
3. field111:{value:"2,3",specialobj:[
4. {id:"2",name:"张三"},{id:"3",name:"李四"}
5. ]},
6. ...
7. },{
8. field110:{viewAttr:2},
9. field111:{viewAttr:3},
10. ...
11. });

**3.7 触发指定字段涉及的所有联动**

*说明：手动触发一次字段涉及的所有联动，包括字段联动、SQL联动、日期时间计算、字段赋值、公式、行列规则、显示属性联动、选择框联动、bindPropertyChange事件绑定等*

*场景：触发出的子流程打开默认不执行字段联动、归档节点查看表单不执行联动，可通过此接口实现*

*triggerFieldAllLinkage:function(fieldMark)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |

样例

1. //表单打开强制执行某字段的联动
2. jQuery(document).ready(function(){
3. WfForm.triggerFieldAllLinkage("field110"); //执行字段涉及的所有联动
4. });

**3.8 根据字段ID获取字段信息**

*说明：根据字段ID获取字段信息，JSON格式，包括名称、类型、只读必填属性等*

*getFieldInfo:function(fieldid)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldid | String | 是 | 字段ID，不带任何标示 |

返回值字段信息JSON重要属性说明
参数|说明
—-|—-
htmltype|字段大类型(文本/多行文本…)
detailtype|字段小类型(整数/浮点数…)
fieldname|字段数据库名称
fieldlabel|字段显示名
viewattr|字段属性(1:只读；2：可编辑；3：必填)

1. WfForm.getFieldInfo("111");

**3.9 获取字段当前的只读/必填属性**

*此方法为实时获取字段显示属性，包含显示属性联动、代码接口变更、已办、明细已有字段不可修改等可能的变更情况，不是仅仅获取后台配置的字段属性；*

*如只想获取后台配置的字段属性，调用接口3.8取返回值viewattr属性*

*getFieldCurViewAttr:function(fieldMark)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |

1. WfForm.getFieldCurViewAttr("field110\_2"); //获取明细字段属性，1：只读、2：可编辑、3：必填；已办全部为只读；

**4.表单字段事件绑定、自定义渲染**

**4.1 表单字段值变化触发事件**

*字段值变化即会触发所绑定的函数，可多次绑定*

*bindFieldChangeEvent: function(fieldMarkStr,funobj)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMarkStr | String | 是 | 绑定字段标示，可多个拼接逗号隔开，例如：field110(主字段),field111\_2(明细字段)…… |
| funobj | Function | 是 | 字段值变化触发的自定义函数，函数默认传递以下三个参数，参数1：触发字段的DOM对象，参数2：触发字段的标示(field27555等)，参数3：修改后的值 |

1. WfForm.bindFieldChangeEvent("field27555,field27556", function(obj,id,value){
2. console.log("WfForm.bindFieldChangeEvent--",obj,id,value);
3. });

**特别注意**

1. 如果字段绑定事件，事件内改变本字段的值，需要setTimeout延时下
2. WfForm.bindFieldChangeEvent("field111", function(obj,id,value){
3. window.setTimeout(function(){
4. WfForm.changeFieldValue("field111",{value:"修改本字段值需要延时"});
5. }, 10);
6. WfForm.changeFieldValue("field222",{value:"修改非本字段不需要延时"});
7. });

**4.2 明细字段值变化触发事件**

*绑定后对新添加的明细行字段以及加载的已有行明细字段，值变更触发所绑定的事件*

*bindDetailFieldChangeEvent: function(fieldMarkStr,funobj)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMarkStr | String | 是 | 绑定的明细字段标示，不能有下划线标示，可多个拼接逗号隔开，例如：field110,field111 |
| funobj | Function | 是 | 字段值变更触发自定义函数，函数默认传递以下三个参数，参数1：字段标示(field27583)，参数2：行标示，参数3：修改后的值 |

1. jQuery(document).ready(function(){
2. WfForm.bindDetailFieldChangeEvent("field27583,field27584",function(id,rowIndex,value){
3. console.log("WfForm.bindDetailFieldChangeEvent--",id,rowIndex,value);
4. });
5. });

**4.3 字段区域绑定动作事件**

*推荐使用值变化事件实现开发,因为此接口点击、双击等动作不是绑定到字段元素，是字段所在单元格区域即会触发*

*此接口所有功能都通可以新版公式实现*

| **类型** | **说明** |
| --- | --- |
| onblur | 失去焦点事件，仅支持单行文本类型 |
| onfocus | 获取焦点事件，仅支持单行文本字段类型 |
| onclick | 单击事件，字段所在单元格区域单击触发 |
| ondbclick | 双击事件，字段所在单元格区域双击触发 |
| mouseover | 鼠标移入事件，鼠标移入字段所在单元格区域触发 |
| mouseout | 鼠标移出事件，鼠标移出字段所在单元格区域触发 |

*bindFieldAction: function(type, fieldids, fn)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| type | String | 是 | 动作类型，见上表 |
| fieldids | String | 是 | 字段id集合，多个逗号分隔，明细字段不加下划线对所有行生效 |
| fn | Function | 是 | 触发函数，此函数入参接收两个参数，fieldid以及rowIndex行号 |

样例

1. WfForm.bindFieldAction("onfocus", "field111,field222", function(fieldid,rowIndex){
2. alert("单行文本字段111获取焦点触发事件");
3. alert("明细第"+rowIndex+"行字段222获取焦点触发事件");
4. });
5. WfForm.bindFieldAction("onclick", "field333", function(){
6. alert("浏览按钮字段单击触发事件，不是指点放大镜选择，是整个字段所在单元格区域单击都会触发");
7. });

**4.4 自定义代理渲染单行文本框字段**

*最低版本要求：KB900190407*

*此接口仅对单行文本框字段类型生效，即数据库字段类型为varchar*

*显示效果、事件、字段值交互都可自行控制，通过接口3.3修改的可编辑字段值也会正常入库*

*此接口传入的组件，产品会传入此字段依赖的相关props，具体可通过React Developer Tools自行抓取，按需调用*

*proxyFieldComp: function(fieldMark, el, range)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| el | React Comp | 是 | 渲染的组件 |
| range | String | 否 | 作用范围，默认全部，(1:只读、2:可编辑、3:必填),组合后逗号分隔 |

样例

1. WfForm.proxyFieldComp("field111", React.createElement("div",{
2. style:{background:"red"},
3. children:"子内容"
4. }));
5. //字段111在只读、可编辑、必填情况下自定义渲染
6. WfForm.proxyFieldComp("field222\_1", "<div>自定义渲染字段</div>", "2,3");
7. //明细某行字段222再可编辑、必填情况下自定义渲染

**4.5 自定义追加渲染表单字段**

*最低版本要求：KB900190407*

*在标准字段展现内容的基础上，after方式追加渲染自定义组件*

*此接口参数说明与用法，与接口4.4类同*

*afterFieldComp: function(fieldMark, el, range)*

样例

1. WfForm.afterFieldComp("field111",
2. React.createElement("a",{
3. href:"/test.jsp?userid="+WfForm.getFieldValue("field222"),
4. children:"自定义链接"
5. }));
6. //字段111在只读、可编辑、必填情况下,追加渲染个自定义链接，链接参数依赖表单其它字段值

**4.6 函数式自定义渲染表单字段**

*最低版本要求：KB900190701*

*以函数返回值方式自定义渲染表单字段，支持全部的字段类型，可实现基于原组件追加/复写/重新布局等等*

*建议结合ecode工具，放到模块加载前调用，使用JSX，可实现与表单字段渲染有关的二次开发*

*此接口的优先级高于4.4、4.5，即使用此接口代理的字段，如再使用4.4、4.5会直接无效*

*proxyFieldContentComp: function(fieldid,fn)*

接口参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| field | String | 是 | 主表/明细表字段ID,格式$fieldid$ |
| fn | Function | 是 | 代理的函数，此函数必须有返回值，返回自定义渲染的组件 |

代理的函数参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| info | JSON | 是 | 字段基础信息，包括字段值、字段属性等等 |
| compFn | Function | 是 | 代理前原字段组件函数，可通过此函数获取原组件 |

样例

1. WfForm.proxyFieldContentComp("111", function(info,compFn){
2. console.log("字段id：",info.fieldid);
3. console.log("明细行号：",info.rowIndex);
4. console.log("字段只读必填属性：",info.viewAttr);
5. console.log("字段值：",info.fieldValue);
6. //返回自定义渲染的组件
7. return React.createElement("div", {}, ["想怎么玩就怎么玩"]);
8. //返回原组件
9. return compFn();
10. //返回基于原组件的复写组件
11. return React.createElement("div", {}, ["前置组件",compFn(),"后置组件"]);
12. });
13. //如果此接口调用在代码块、custompage等(非模块加载前调用)，需强制渲染字段一次
14. WfForm.forceRenderField("field111");

**4.7 根据字段标识获取字段组件**

*最低版本要求：KB900190701*

*根据字段标识，获取字段组件，即字段组件可单独提取出来放在任意地方渲染。注意字段的只读可编辑属性仍需后台事先设置好*

*需要结合ecode工具，使用JSX，同时Provider 注入Store，再结合设计器自定义属性或接口4.6，可实现某一区域自定义排版布局渲染多个表单字段*

*generateFieldContentComp:function(fieldMark)*

接口参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |

样例

1. //以明细多字段、子明细的需求为例子
2. 步骤1：模板单元格给自定义class：area\_1
3. 步骤2：自定义排版渲染area\_1区域
4. const {Provider}=mobxReact;
5. const globalStore = WfForm.getGlobalStore();
6. const layoutStore = WfForm.getLayoutStore();
7. ReactDOM.render(<div>
8. <Provider globalStore={globalStore} layoutStore={layoutStore}>
9. <table>
10. <tr>
11. <td>{WfForm.generateFieldContentComp("field111\_1")}</td>
12. <td>{WfForm.generateFieldContentComp("field112\_1")}</td>
13. </tr>
14. <tr>
15. <td>{WfForm.generateFieldContentComp("field113\_1")}</td>
16. <td>{WfForm.generateFieldContentComp("field114\_1")}</td>
17. </tr>
18. </table>
19. </Provider>
20. </div>, document.getElementByclassName("area\_1")[0]); //仅供参考，参数二要区分行号定位到具体单元格

**5.明细表操作相关接口**

**5.1 添加明细行并设置初始值**

*addDetailRow: function(detailMark, initAddRowData={})*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | 是 | 明细表标示，明细1就是detail\_1，以此递增类推 |
| initAddRowData | JSON | 否 | 给新增后设置初始值，格式为{field110:{value:”11”},field112:{value:”22”},…},注意key不带下划线标示 |

样例

1. //明细2添加一行并给新添加的行字段field111赋值
2. WfForm.addDetailRow("detail\_2",{field111:{value:"初始值"}});
3. //添加一行并给浏览按钮字段赋值
4. WfForm.addDetailRow("detail\_2",{field222:{
5. value: "2,3",
6. specialobj:[
7. {id:"2",name:"张三"},
8. {id:"3",name:"李四"}
9. ]
10. }});
11. //动态字段赋值，明细1添加一行并给字段名称为begindate的字段赋值
12. var begindatefield = WfForm.convertFieldNameToId("begindate", "detail\_1");
13. var addObj = {};
14. addObj[begindatefield] = {value:"2019-03-01"};
15. WfForm.addDetailRow("detail\_1", addObj);
16. //不推荐这种动态键值写法，IE不支持，避免掉
17. WfForm.addDetailRow("detail\_1",{[begindatefield]:{value:"2019-03-01"}})

**5.2 删除明细表指定行/全部行**

*delDetailRow: function(detailMark, rowIndexMark)*

*说明：此方法会清空明细已选情况，删除时没有提示”是否删除”的确认框*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | 是 | 明细表标示，明细1就是detail\_1，以此递增类推 |
| rowIndexMark | String | 是 | 需要删除的行标示，删除全部行:all,删除部分行：”1，2，3” |

样例

1. WfForm.delDetailRow("detail\_1", "all"); //删除明细1所有行
2. WfForm.delDetailRow("detail\_1", "3,6"); //删除明细1行标为3,6的行

**5.3 选中明细指定行/全部行**

*checkDetailRow: function(detailMark, rowIndexMark,needClearBeforeChecked)*

*说明：此方法可灵活使用，依靠参数needClearBeforeChecked可实现清除选中的逻辑*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | 是 | 明细表标示，明细1就是detail\_1，以此递增类推 |
| rowIndexMark | String | 否 | 需要选中的行标示，选中全部行:all,选中部分行：”1，2，3” |
| needClearBeforeChecked | Boolean | 否 | 是否需要清除已选 |

样例

1. WfForm.checkDetailRow("detail\_2", "all"); //勾选明细2所有行
2. WfForm.checkDetailRow("detail\_2", "", true); //清除明细2所有已选
3. WfForm.checkDetailRow("detail\_2", "3,6", true); //清除明细2全部已选，再勾选行标为3,6的行
4. WfForm.checkDetailRow("detail\_2", "7", false);
5. //保持已选记录，追加选中行标为7的行

**5.4 获取明细行所有行标示**

*getDetailAllRowIndexStr： function(detailMark)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | 是 | 明细表标示，明细1就是detail\_1，以此递增类推 |

样例

1. console.log(WfForm.getDetailAllRowIndexStr("detail\_2")); //输出1，3...等等

特别注意

1. 遍历明细行的写法
2. var rowArr = WfForm.getDetailAllRowIndexStr("detail\_1").split(",");
3. for(var i=0; i<rowArr.length; i++){
4. var rowIndex = rowArr[i];
5. if(rowIndex !== ""){
6. var fieldMark = "field111\_"+rowIndex; //遍历明细行字段
7. }
8. }

**5.5 获取明细选中行下标**

*最低版本要求：KB900190501*

*getDetailCheckedRowIndexStr： function(detailMark)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | 是 | 明细表标示，明细1就是detail\_1，以此递增类推 |

样例

1. console.log(WfForm.getDetailCheckedRowIndexStr("detail\_2")); //输出选中行1，3...等等

**5.6 控制明细行check框是否禁用勾选**

*注：后台配置的置灰行(不允许删除情况)，不支持通过此API控制*

*controlDetailRowDisableCheck: function(detailMark, rowIndexMark, disableCheck)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | true | 明细表标示，明细1就是detail\_1，以此递增类推 |
| rowIndexMark | String | 是 | 需要选中的行标示，选中全部行:all,选中部分行：”1，2，3” |
| disableCheck | boolean | true | 是否禁用勾选，true:置灰禁止勾选，false:允许勾选 |

1. WfForm.controlDetailRowDisableCheck("detail\_1", "all", true); //明细1所有行check框置灰禁止选中
2. WfForm.controlDetailRowDisableCheck("detail\_1", "1,2", false);
3. //明细1行标为1,2的行释放置灰，允许勾选

**5.7 控制明细数据行的显示及隐藏**

*注：只是界面效果隐藏，序号不会变化，即被隐藏行的前后行序号会断层不连续*

*controlDetailRowDisplay: function(detailMark, rowIndexMark, needHide)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | true | 明细表标示，明细1就是detail\_1，以此递增类推 |
| rowIndexMark | String | 是 | 需要选中的行标示，选中全部行:all,选中部分行：”1，2，3” |
| needHide | boolean | true | 是否隐藏行，true:隐藏，false:显示 |

1. WfForm.controlDetailRowDisplay("detail\_1", "3,5", true); //明细1行标为3,5的隐藏不显示
2. WfForm.controlDetailRowDisplay("detail\_1", "all", false);
3. //明细1所有行不隐藏都显示

**5.8 获取明细已有行的数据库主键**

*getDetailRowKey: function(fieldMark)*

*此方法只对明细已有行生效，新增加的行/不存在的行返回-1*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号}，用于定位属于哪个明细表 |

1. WfForm.getDetailRowKey("field112\_3"); //获取明细第四行主键

**5.9 获取明细总行数**

*getDetailRowCount： function(detailMark)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | 是 | 明细表标示，明细1就是detail\_1，以此递增类推 |

样例

1. console.log(WfForm.getDetailRowCount("detail\_2")); //输出明细总行数，注意此结果只代表明细总行数，不能用作循环行

**5.10 添加行、删除行前执行逻辑或阻断事件**

*最低版本要求：KB900190501*

*场景：可实现添加行前执行自定义逻辑、限制超过多少行添加无效、不允许删除等*

*采用注册函数机制，详见接口2.1*

**5.11 添加行、删除行后触发事件**

*采用钩子机制，详见接口2.2*

**5.12 移动端跳转至明细编辑行页面执行事件**

*最低版本要求：KB900190501*

*仅应用于移动端编辑明细行，采用钩子机制，详见接口2.2*

**5.13 添加明细时默认复制最后一行记录**

*setDetailAddUseCopy: function(detailMark, needCopy)*

*说明：此方法在ready时调用，手动点添加时自动赋值最后行字段内容，覆盖节点前设置及默认值等，附件上传字段不予复制*

*注：E9是异步ready执行后点添加明细才生效，例如默认新增空明细无效;*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| detailMark | String | 是 | 明细表标示，明细1就是detail\_1，以此递增类推 |
| needCopy | Boolean | 是 | 是否需要启用复制，true：启用，false：不启用 |

1. jQuery(document).ready(function(){
2. WfForm.setDetailAddUseCopy("detail\_1", true);
3. });

**5.14 根据明细行标识获取序号(第几行)**

*最低版本要求：KB900190601*

*场景：根据明细行的下标，获取当前是第几行明细，可用于提示某某行异常等*

*getDetailRowSerailNum: function(mark, rowIndex)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| mark | String | 是 | 明细表标示，支持两种格式detail\_${dindex}或者field${fieldid}\_${rowIndex} |
| rowIndex | Int | 是 | 行标识，第一种格式detail\_${dindex}才需要传此参数 |

样例

1. WfForm.getDetailRowSerailNum("detail\_1", 3); //获取明细1下标为3的行序号
2. WfForm.getDetailRowSerailNum("field222\_3"); //获取字段222对应明细表下标为3的行序号

**6.常用全局接口(与字段无关)**

**6.1 获取当前打开请求的基础信息**

*说明：包括路径id、节点id、表单id、主次账号信息*

*getBaseInfo: function()*

1. console.log(WfForm.getBaseInfo()); //返回当前请求基础信息
2. //输出对象说明：
3. {
4. f\_weaver\_belongto\_userid: "5240" //用户信息
5. f\_weaver\_belongto\_usertype: "0"
6. formid: -2010 //表单id
7. isbill: "1" //新表单/老表单
8. nodeid: 19275 //节点id
9. requestid: 4487931 //请求id
10. workflowid: 16084 //路径id
11. }

**6.2 可控制显示时间的message信息**

*showMessage: function(msg, type, duration)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| msg | String | true | 提示信息内容 |
| type | int | false | 提示类型，1(警告)、2(错误)、3(成功)、4(一般)，默认为1，不同类型提示信息效果不同 |
| duration | Float | false | 多长时间自动消失，单位秒，默认为1.5秒 |

1. WfForm.showMessage("结束时间需大于开始时间"); //警告信息，1.5s后自动消失
2. WfForm.showMessage("运算错误", 2, 10); //错误信息，10s后消失

**6.3 系统样式的Confirm确认框**

*说明：兼容移动端，可自定义确认内容及按钮名称*

*showConfirm: function(content, okEvent, cancelEvent, otherInfo={})*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| content | String | 是 | 确认信息 |
| okEvent | Function | 是 | 点击确认事件 |
| cancelEvent | Function | 否 | 点击取消事件 |
| otherInfo | Object | 否 | 自定义信息(按钮名称) |

1. WfForm.showConfirm("确认删除吗？", function(){
2. alert("删除成功");
3. });
4. WfForm.showConfirm("请问你是否需要技术协助？",function(){
5. alert("点击确认调用的事件");
6. },function(){
7. alert("点击取消调用的事件");
8. },{
9. title:"信息确认", //弹确认框的title，仅PC端有效
10. okText:"需要", //自定义确认按钮名称
11. cancelText:"不需要" //自定义取消按钮名称
12. });

**6.4 表单顶部按钮、右键菜单置灰**

*说明：设置表单顶部按钮、右键菜单置灰不可操作和恢复操作功能参数isDisabled*

*controlBtnDisabled: function(isDisabled)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| isDisabled | boolean | 是 | true：按钮全部置灰不可操作,false：恢复按钮可操作状态 |

1. function subimtForm(params){
2. WfForm.controlBtnDisabled(true); //操作按钮置灰
3. ...
4. WfForm.controlBtnDisabled(false);
5. }

**6.5 调用右键按钮事件**

*说明：调用表单右键事件逻辑，只可调用，不允许复写*

*doRightBtnEvent: function(type)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| type | String | 是 | 按钮类型(控制台可通过mobx.toJS(WfForm.getGlobalStore().rightMenu.rightMenus)方式定位具体按钮type) |

1. WfForm.doRightBtnEvent("BTN\_SUBBACKNAME"); //触发提交需反馈
2. WfForm.doRightBtnEvent("BTN\_SUBMIT"); //触发提交不需反馈
3. WfForm.doRightBtnEvent("BTN\_WFSAVE"); //触发保存
4. WfForm.doRightBtnEvent("BTN\_REJECTNAME"); //触发退回
5. WfForm.doRightBtnEvent("BTN\_FORWARD"); //触发转发
6. WfForm.doRightBtnEvent("BTN\_REMARKADVICE"); //触发意见征询
7. WfForm.doRightBtnEvent("BTN\_TURNTODO"); //触发转办
8. WfForm.doRightBtnEvent("BTN\_DORETRACT"); //触发强制收回
9. WfForm.doRightBtnEvent("BTN\_PRINT"); //触发打印

**6.6 刷新表单页面**

*强制刷新表单页面，默认为当前requestid*

*reloadPage: function(params={})*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| params | Object | 否 | 自定义参数，覆盖默认参数 |

1. WfForm.reloadPage();
2. WfForm.reloadPage({requestid:"11"}); //覆盖参数

**6.7 移动端打开链接方式**

*仅支持移动端，特别是非表单主界面(例如：明细编辑)需要用此方式打开链接。
此方式打开链接返回不会刷新表单*

*window.showHoverWindow：function(url,baseRoute)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| url | String | 是 | 打开的链接地址 |
| baseRoute | String | 是 | 当前路由地址，具体见url地址，明细编辑打开传’/req/editDetailRow’ |

样例

1. window.showHoverWindow('/workflow/test.jsp', '/req'); //主界面打开链接
2. window.showHoverWindow('https://www.baidu.com', '/req/editDetailRow'); //明细行编辑界面打开链接

**6.8 扩展提交操作发送给服务端的参数**

*最低版本要求：KB900190801*

*自定义扩展提交/保存动作发送给服务端的参数，服务端可通过request.getParameter方式取到对应参数值*

*推荐：扩展的自定义参数都以cus\_开头，避免影响/覆盖标准产品所必需的参数，导致功能异常*

*appendSubmitParam: function(obj={})*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| obj | Object | 否 | JSON格式自定义参数 |

1. WfForm.appendSubmitParam({cus\_param1:"11", cus\_param2:"测试"}); //服务端可通过request对象取到参数值request.getParameter("cus\_param1"):

**6.9 获取校验必填逻辑第一个未必填的字段**

*最低版本要求：KB900191201*

*场景：调用产品的必填校验逻辑，获取第一个未必填字段；
例如结合接口2.1中WfForm.OPER\_BEOPER\_BEFOREVERIFY可实现自定义控制必填提示的效果*

*getFirstRequiredEmptyField: function()*

1. var emptyField = WfForm.getFirstRequiredEmptyField(); //获取调用时刻的第一个未必填字段，返回值格式是`field${fieldid}\_${rowIndex}`

**6.10 触发一次必填验证**

*最低版本要求：KB900191201*

*手动触发一次必填验证并提示，可选控制校验必须新增空明细/校验字段必填*

*verifyFormRequired: function(mustAddDetail=true, fieldRequired=true)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| mustAddDetail | Boolean | 否 | 是否校验必须新增空明细，默认为是 |
| fieldRequired | Boolean | 否 | 是否校验字段必填，默认为是 |

1. WfForm.verifyFormRequired(); //触发必填验证并提示，先校验必须新增空明细，后校验字段必填
2. var result = WfForm.verifyFormRequired(false, true); //仅校验字段必填并提示
3. if(!result)
4. alert('存在未必填情况');

**7.不同字段类型特定接口(限定指定字段类型可用)**

**7.1 扩展浏览按钮取数接口参数值**

*限定条件：仅适用非日期时间的浏览按钮类型*

*场景：控制浏览按钮可选数据范围，限定范围、依赖表单字段过滤数据范围等；对联想输入范围及弹窗选择范围都生效；*

*实现方式：接口扩充的参数会通过url参数提交到服务端接口，需结合修改浏览按钮接口类方可生效*

*appendBrowserDataUrlParam: function(fieldMark, jsonParam)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| jsonParam | JSON | 是 | 扩展的url参数，JSON中key-value格式 |

1. WfForm.appendBrowserDataUrlParam("field395",{"customerid":"2"});//给浏览按钮395请求后台数据时增加url参数customerid

**7.2 获取浏览按钮的显示值**

*限定条件：仅适用非日期时间的浏览按钮类型*

*获取浏览按钮的显示名称，多个则以splitChar字符分隔拼接成串*

*getBrowserShowName:function(fieldMark,splitChar)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| splitChar | String | 否 | 分隔符，默认以逗号分隔 |

1. WfForm.getBrowserShowName("field110"); //以逗号分隔获取浏览按钮字段显示值

**7.3 移除选择框字段选项**

*限定条件：仅适用选择框类型字段*

*removeSelectOption: function(fieldMark, optionKeys)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| optionKeys | String | 是 | 需要移除的Option选项key值，多个以逗号分隔 |

1. WfForm.removeSelectOption("field112", "3,4"); //移除选择框中id值为3/4的选项

**7.4 控制选择框字段选项**

*限定条件：仅适用选择框类型字段*

*controlSelectOption:function(fieldMark, optionKeys)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| optionKeys | String | 是 | 完全控制选择框的选项范围 |

1. WfForm.controlSelectOption("field112", "1,2,4"); //控制选择框只显示1/2/4的选项
2. WfForm.controlSelectOption("field112", ""); //清除选择框所有选项

**7.5 获取选择框字段的显示值**

*限定条件：仅适用选择框类型字段*

*获取选择框类型的显示名称，多个则以splitChar字符分隔拼接成串*

*getSelectShowName:function(fieldMark,splitChar)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| splitChar | String | 否 | 分隔符，默认以逗号分隔(只有复选框多选才会用到) |

1. WfForm.getSelectShowName("field110\_3"); //获取选择框字段显示值

**7.6 文本字段可编辑状态，当值为空显示默认灰色提示信息，鼠标点击输入时提示消失**

*限定条件：仅支持单行文本、整数、浮点数、千分位、多行文本字段(非html)字段类型；支持主字段及明细字段*

*setTextFieldEmptyShowContent:function(fieldMark,showContent)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| showContent | String | 是 | 空值时显示的提示信息，灰色 |

1. jQuery(document).ready(function(){
2. WfForm.setTextFieldEmptyShowContent("field27555", "单文本默认提示信息1");
3. WfForm.setTextFieldEmptyShowContent("field27566", "多文本默认提示2");
4. WfForm.setTextFieldEmptyShowContent("field222\_0", "明细字段提示信息"); //需要结合接口5.9添加行事件一并使用
5. });

**7.7 复写浏览按钮组件的props**

*仅支持浏览按钮类型，谨慎使用，完全重写覆盖浏览按钮的props*

*只是传递props，具体传递的属性实现何种需求由组件内部控制*

*overrideBrowserProp: function(fieldMark,jsonParam)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| jsonParam | JSON | 是 | 扩展或复写的props参数 |

样例

1. WfForm.overrideBrowserProp("field111",{
2. onBeforeFocusCheck: function(success, fail){/\*\*\*/}
3. }); //复写浏览按钮字段111的props

**7.8 控制日期浏览按钮的可选日期范围**

*最低版本要求：KB900190501*

*仅支持日期类型，控制手动选择时的可选日期范围*

*controlDateRange: function(fieldMark,start,end)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| start | String | 是 | 支持两种格式，第一种标准的日期格式，比如2019-05-28，第二种整数，相比于当前日期往前/后多少天 |
| end | String | 否 | 格式与start参数一致 |

样例

1. WfForm.controlDateRange("field111", -5, 10); //限定日期可选范围，往前5天，往后10天
2. WfForm.controlDateRange("field111", 0, '2019-12-31'); //限定今天至本年
3. WfForm.controlDateRange("field222\_0", '2019-05-01', '2019-05-31'); //明细字段，限定当月

**7.9 控制Radio框字段打印是否仅显示选中项文字**

*最低版本要求：KB900190501*

*仅支持选择框中单选框类型，打印场景，是否仅显示选中项文字，都未选中显示空*

*controlRadioPrintText: function(fieldid)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段id，格式${字段ID},支持明细字段 |

样例

1. WfForm.controlRadioPrintText("12580"); //单选框字段12580打印只显示文字

**8.签字意见接口**

**8.1 获取签字意见内容**

*最低版本要求：KB900190501*

*getSignRemark: function()*

样例

1. WfForm.getSignRemark(); //获取签字意见内容

**8.2 设置签字意见内容**

*最低版本要求：KB900190501*

*setSignRemark: function(text,isClear=true,isAfter=true,callback)*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| text | String | 是 | 需设置的内容 |
| isClear | Boolean | 否 | 是否先清除意见，默认为true，即覆盖意见内容，false为追加意见内容 |
| isAfter | Boolean | 否 | 追加的位置，默认为true，原意见内容尾部追加，false再头部追加 |
| callback | Function | 否 | 意见设置成功后的回调函数 |

样例

1. WfForm.setSignRemark("覆盖设置签字意见内容");
2. WfForm.setSignRemark("原有意见内容前追加请审批", false, false);

**8.3 扩展签字意见输入框底部按钮**

*签字意见输入框底部按钮，在产品内置的附件、文档、流程基础上，支持自定义扩充*

*appendSignEditorBottomBar: function(comps=[])*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| comps | React Comp Array | 是 | 需要扩充的React组件数组 |

1. WfForm.appendSignEditorBottomBar([
2. React.createElement("div",{className:"wf-req-signbtn",children:"自定义按钮1"}),
3. <div>自定义按钮2</div>
4. ]);

**9.历史E8代码块的相关兼容**

*以下方法不推荐使用，只为兼容历史代码块，上述WfForm中API都涵盖包括*

*建议全部改造通过WfForm接口实现，使用老版本接口、操作Dom结构等，很容易引发问题及后续版本升级不兼容*

**9.1 提交事件执行自定义函数**

*建议使用2.1 拦截提交事件代替*

*比如checkCustomize、checkCarSubmit会继续执行,根据函数返回值判断是否阻塞提交，返回值true：继续流转，flase:阻断提交*

开发示例

1. window.checkCustomize = function(){
2. var flag = true;
3. //...
4. return flag;
5. }

**9.2 字段值变化触发事件bindPropertyChange**

*建议使用4.1 监听字段变化代替*

*说明：与E8一致，依赖DOM，字段值变化时触发此函数;
自定义函数默认传递以下三个参数，参数1：触发字段的DOM对象，参数2：触发字段的标示(field27563等)，参数3：修改后的值*

样例

1. jQuery(document).ready(function(){
2. jQuery("#field27563").bindPropertyChange(function(obj,id,value){
3. console.log("tri...",obj,id,value);
4. });
5. });

**9.3 明细新增行渲染后触发事件**

*建议使用2.2钩子事件代替*

*说明：重载\_customAddFun”+groupid+”函数，groupid从0开始递增，0代表明细1；不论手动添加、联动添加、接口添加都会触发*

样例

1. function \_customAddFun1(addIndexStr){ //明细2新增成功后触发事件，addIndexStr即刚新增的行标示，添加多行为(1,2,3)
2. console.log("新增的行标示："+addIndexStr);
3. }

**9.4 明细删除行渲染后触发事件**

*建议使用2.2钩子事件代替*

*说明：重载\_customDelFun”+groupid+”函数，groupid从0开始递增，0代表明细1；不论手动删除、接口删除都会触发*

样例

1. function \_customDelFun1(){ //明细2删除成功后触发事件
2. console.log("删除明细");
3. }

**9.5 修改浏览按钮字段值window.\_writeBackData**

*此方法只为兼容E8已有代码，新开发的请使用WfForm.changeFieldValue方式赋值*

参数说明

| **参数** | **参数类型** | **必须** | **说明** |
| --- | --- | --- | --- |
| fieldMark | String | 是 | 字段标示，格式field${字段ID}\_${明细行号} |
| isMustInput | String | 是 | E8参数，E9暂时不用 |
| data | JSON | 是 | 修改的浏览按钮数据，格式{id:”***“,name:”***“} |
| \_options | JSON | 否 | 修改方式，格式{replace:false,isSingle:true}，replace或isSingle任意一个为true即为覆盖修改，否则为在原有浏览按钮数据基础上追加修改，默认为覆盖修改 |

1. \_writeBackData("field110", 1, {id:"22",name:"lzy"},{replace:false,isSingle:false}); //多人力浏览按钮追加个值lzy

**9.6 少用jQuery操作**

*不推荐使用,历史jQuery(“#field111”).val()取值、赋值等操作已兼容*

**9.7 禁止JS原生操作**

*不允许，会引发各种不可控异常*

*类似document.write(“”);document.getElementById(“field111”).value操作需要调整为WfForm接口操作*

**10 常用配置文件修改方式**

*以下配置修改都是针对当前ecology系统全部流程生效*

**10.1 修改意见默认字体**

*sysadmin账号登录，在浏览器调用接口的方式修改配置*

/api/workflow/index/updateWfConfig?name=signinput\_default\_fontfamily&value=仿宋\_GB2312/FangSong\_GB2312

*value参数(字体)范围*

| **字体值** | **—** |
| --- | --- |
| 仿宋\_GB2312/FangSong\_GB2312 |  |
| 宋体/SimSun |  |
| 微软雅黑/Microsoft YaHei |  |
| 楷体/KaiTi |  |
| 黑体/SimHei |  |
| Arial/Arial, Helvetica, sans-serif |  |
| 新宋体/NSimSun |  |
| 楷体\_GB2312/KaiTi\_GB2312 |  |

**10.2 修改意见默认字体大小**

*sysadmin账号登录，在浏览器调用接口的方式修改配置*

/api/workflow/index/updateWfConfig?name=signinput\_default\_fontsize&value=36/36px

*value参数(字体大小)范围*

| **字体大小** | **—** | **—** | **—** |
| --- | --- | --- | --- |
| 8/8px | 9/9px | 10/10px | 11/11px |
| 2/12px | 14/14px | 16/16px | 18/18px |
| 20/20px | 22/22px | 24/24px | 26/26px |
| 28/28px | 36/36px |  |  |

**10.3 流程自定义浏览框缓存功能开关**

*sysadmin访问链接修改配置，实时生效*

/api/workflow/index/updateWfConfig?name=un\_use\_customize\_browser\_cache&value=0

*value参数1关闭，0开启*

*缓存任务参数及清理数据缓存页面（任务功能：保存表单、打开表单进行自定义浏览框数据缓存及刷新、以及清理自定义浏览框缓存数据批量清理以及单个流程清理）*

/workflow/request/CustomizeBrowserCacheUtil.jsp

**10.4 非多行文本html字段类型支持html格式**

*场景：E9限制只有当字段类型为多行文本且勾选html的字段，内容才支持html格式
单行文本字段、多行文本字段是不支持，但是部分场景此类字段值是通过外面接口等情况赋值html串
此情况可通过改配置实现*

第一步：
找到需要支持html格式的字段id(可通过表单设计器模板单元格选中看右下角)
假设字段id为12345，则格式为： field12345\_1
(如是老表单，则格式为 field12345\_0，老表单指字段从字段库选择生成的表单，一般是E8系统前)
第二步：
执行SQL(不同数据库请转换拼接符)

1. update workflow\_config set value=value||',field12345\_1' where name='support\_html\_textarea\_field'

第三步：
重启resin生效

**10.5 PC端-流程表单显示底部耗时信息开个(调试分析用)**

*最低版本要求：KB900190801
sysadmin访问链接修改配置，实时生效*

/api/workflow/index/updateWfConfig?name=show\_duration\_log&value=1

*value参数1开启，0关闭，默认为关闭*

**10.6 明细开启横向滚动条情况下，首行(按钮所在行)固定不跟随滚动条滚动**

*最低版本要求：KB900191101
场景: 明细开启横向滚动条、首行不包含除按钮/文字外类型、首行含添加删除按钮且处于非列锁定区域
满足上述三条件，首行会固定，不跟随横向滚动条滚动，
此时根据模板不同可能会存在首行与下一行存在不对齐问题
sysadmin访问链接修改配置，实时生效*

/api/workflow/index/updateWfConfig?name=detail\_locked\_button\_row&value=1

*value参数1锁定，0取消锁定，默认为锁定*

**10.7 明细字段合计给主字段，当明细未添加行，赋零值或空值**

*场景: 配置列合计给主字段，当明细没有添加过行，主表合计字段，
有些场景需要赋值为空，用于校验必填
有些场景需要赋值为零，用于出口判断*

*sysadmin访问链接修改配置，实时生效*

/api/workflow/index/updateWfConfig?name=colRule\_noRow\_empty&value=1

*value参数1赋零值，0赋空值，默认为赋空值*

**10.8 pc端-手写签批按钮开关**

*最低版本要求：KB900191001
sysadmin访问链接修改配置，实时生效*

/api/workflow/index/updateWfConfig?name=handwrittensign\_switch&value=1

*value参数1开启，0关闭*

**10.9 移动端-选择框单选框类型显示成radio效果开关**

*最低版本要求：KB900191101
sysadmin访问链接修改配置，实时生效*

/api/workflow/index/updateWfConfig?name=mobile\_show\_radio&value=1

*value参数1开启，0关闭，默认为关闭*

**10.10 移动端-表单正文、附件签批功能开关**

*最低版本要求：KB900190308
修改配置文件：MobileWFOfficeSign.properties*

mobileWFOffice=1

*mobileWFOffice参数1开启，0关闭，开启后显示签批按钮，否则不显示签批按钮*

mobilePDFSign=1

*mobilePDFSign 最低版本要求：KB900190901
mobilePDFSign 配置项只对pdf文件生效
mobilePDFSign 参数1 pdf文件签批后放在签字意见附件中 参数2 pdf文件签批后直接替换原文件 参数3 由用户选择签批后放在签字意见附件中还是替换原文件*

**11 常用CSS样式案例分享**

*根据客户需求范围，灵活变通
仅对当前模板生效—->写在代码块里面
仅对当前路径所有节点生效—->写在路径基础设置-自定义页面(CSS文件)
对系统所有路径生效—->写在应用设置-全局自定义页面(CSS文件)*

*注：如写在代码块中需包一层style标签，写在CSS文件中不需要*

**11.1 实现明细添加删除按钮靠左显示**

*场景：明细添加删除按钮，始终靠右显示，如何实现靠左显示*

插入样式到代码块：

1. <style>
2. .detailButtonDiv{float:left}
3. </style>

如写到CSS文件则不需要style标签

1. .detailButtonDiv{float:left}

**11.2 实现单元格图片居中、自适应缩放**

*场景：单元格插入图片，始终按原始尺寸从左上角平铺显示，如何居中，如何自适应缩放*

*最低版本要求：KB900190901*

图片所在单元格增加自定义属性class标识：

1. imageCell

单元格宽度大于图片尺寸，实现图片居中，增加样式：

1. .imageCell\_swap{background-position:center}

单元格宽度小于图片尺寸，实现图片自适应缩放显示完整，增加样式：

1. .imageCell\_swap{background-size:100% 100%}

**11.3 控制浏览按钮链接颜色**

*场景：浏览按钮链接的颜色不受单元格颜色控制，如何修改浏览按钮链接颜色*

方案一：要求版本达到KB900190901以上

可通过登录sysadmin，访问以下链接开启配置，开启后浏览按钮链接颜色完全取单元格设置的颜色
/api/workflow/index/updateWfConfig?name=browser\_color\_controlByCell&value=1
(value参数0关闭，1开启，默认为关闭)

方案二：不限制版本

浏览按钮所在单元格增加自定义属性class标识：

1. browserColorCell

增加样式

1. .browserColorCell a{color:red !important} //强制将浏览按钮链接显示为红色

**11.4 控制主表选择框字段最小宽度**

*场景：主表选择框字段，最小宽度要求100px，当选项仅为”是/否”时，如何再减小宽度*

选择框所在单元格增加自定义属性class标识：

1. selectCell

增加样式

1. .selectCell .wea-select{min-width:50px !important}

**12 其它场景分享**

**12.1 移动端异构系统提交表单后，如何刷新流程列表**

*最低支持版本：KB900191101*

1. //提交后跳转至此页面
2. window.location.href = 当前ecology服务器地址+"/workflow/workflow/WfRefreshList.jsp"

*注意：移动端可能存在反向代理，请将当前ecology服务器地址配置成反向代理对应的地址*