客户端

Aras

每个 `Innovator` 窗体都有一个 `area` 对象，对象引用 `JavaScript` 方法，在客户端使用。客户端方法在正在查看的窗口上下文运行，可以**被更改**。通常存储在上下文信息

- `inDom` 是许多客户端方法中使用的变量，如果合适，它存储项目上下文
- `inArgs` 用于存储附加信息的变量
- `document.thisItem` 可用于查找上下文项（如果inDom为空）

如果要查找的信息存在于窗口中，无法通过上述的任何变量访问，则可以在窗口路径中获取所需内容。

在Item窗体获取选项卡 JavaScript

```
var relationshipTabInfo = parent.relationships.relTabbar;
var currentRelationshipTabId = relationshipTabInfo.GetSelectedTab();
var tabLabel = relationshipTabInfo.GetTabLabel(currentRelationshipTabId);
```

上下文项

`this` 关键字是 `Item Actions` 的Item对象。 `context Object` 是不用于表单、字段和网格事件的 Item Object。这个context 对象是用于表单和网格事件的浏览器文档(DOM)对象，并且是用于字段事件的字段对象。对应 `parent` ， `thisItem` 是一个指向文档的指针。

IOMInnovator

`newIOMInnovator` 是js获取loovator对象不包含当前 `Context Item`

`applyMethod("方法名称", "body")` ：调用后端方法

- 参数1：调用方法的名称
- 参数2：传递的参数，可以通过 `this.getProperty("name")` 来获取

`applyAML("AML语句")` 向服务端发送AML请求，返回字符串变量

`applyItem(Item)` 执行Item请求， 返回Item，字符串变量

案例，调用服务端方法 JavaScript

```javascript
var inn = aras.newIOMInnovator();
var id = this.getID();
var res = inn.applyMethod("onCompanyServerMethod","<id>"+id+"</id>");
if(res.isError()){
  return alert(res.getErrorString());
}
return res;
```

服务的方法 JavaScript

```javascript
Innovator inn=this.getInnovator();
string id = this.getProperty("id");
Item doc = inn.newItem("cn_company_item", "get");
doc.setProperty("source_id", id);

Item docItem = doc.apply();

for(int i = 0; i < docItem.getItemCount(); i++){
    string relate_id = docItem.getItemByIndex(i).getProperty("related_id")
;
    string sql = $"update document set item_number = '996{i}' where id ='"
+relate_id+"'";
    Item res = inn.applySQL(sql);
}
return this;
```

document对象

访问当前上下文项，表单和字段事件可以使用，能够获取当前 `Item` 及对应的**关系对象**。

1. `document.item` 一个 XML DOM对象，表示与Client Framework API一起使用
2. `document.thisItem` 上下文Item对象（在客户端事件中不能this）
3. `document.itemID` 获取上下项ID
4. `document.itemType` 获取ItemType
5. `window.handleItemChange("field", "value")` 更新值和 `HTML Form Field` 值函数

客户端事件和方法

客户端事件

1. `onBeforeNew` 在创建新项目之前运行，可用于取消新项请求以及阻止新项表单显示
2. `OnNew` 替换新的Item行为，必须为客户端方法提供。
3. `OnAfterNew` 在项目创建后，用于数据填充或打开自定义对话框。服务端已经处理了请求，如果更改(或添加属性)到AML响应，他们将在表单上代替显示。 **`在这个阶段不能使用 window.handleItemChange 方法，用this.setPropery("field", "value")` 代替。**
4. `OnShowItem` 用户替换显示窗口的标准逻辑，客户端方法必须提供自定义逻辑来适当显示窗口。

OnBeforeNew创建新项目 JavaScript

```javascript
// 判断当前年份的日历， 是否可以检索到，如果找到则创建，否则返回False
var item = this.newItem("Business Calendar Year", "get");
item.setProperty("year", "2023");
item = item.apply();
if(item.isError()){
    aras.AlertError("未获取到年份的日历");
    return false;
}
return true;
```

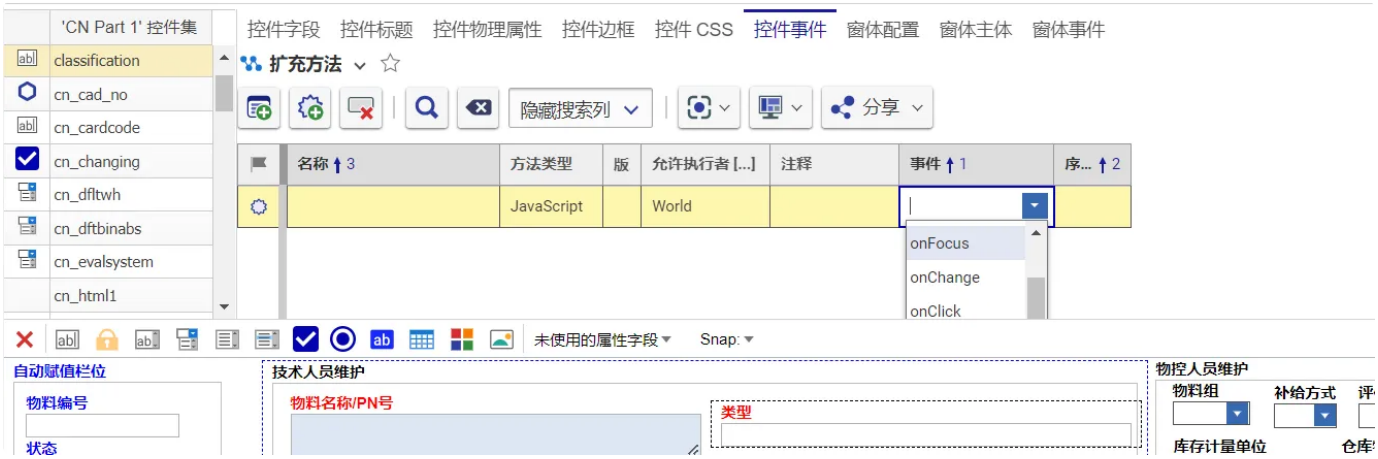
修改从服务器返回的上下文项（响应），可以在用户界面加载更多的信息，以便 `OnAfterNew` 可以和 `Method` 一起使用

OnAfterNew创建之后 JavaScript

```javascript
var d =  new Date();
var nowDate = `${d.getMonth() + 1}/${d.getDate()}/${d.getFullYear()}`;
var msg = "Item Added" + nowDate;

var rel = this.newItem("Design Request Comments", "add");
rel.setProperty("remarks", msg);
this.addRelationship(rel);
return this;
```

控件事件



1. `onFocus` 在获取焦点的时候触发。
2. `onChnage` 在内容改变时触发，通过 `this.value` 来捕获**更改**的值。 `document.thisItem.getProperty` 返回的是**原始字段值**。
3. `onClick` 在点击时触发。
4. `onBlur` 在失去焦点的时候触发。

在控件事件中，通过 `document` 对象来获取当前的节点内容是原始数据， `this.value` 获取的是最新值，这个可以判断修改的值是否与原先相同。

onChange JavaScript

```javascript
// 获取当前的字段值
let value = this.value;
// 获取原值
let oldValue = document.thisItem.getProperty("cn_name")
```

窗体事件

1. `onFormPopulated` 在使用服务器中的数据填充所有字段时运行。
2. `onLoad` 在 HTML 窗体加载到窗口时运行。可以用来加载第三方库。如 `jquery`， `Layui` 等。
3. `onBeforeUnload` 在从视图中删除表单之前运行。
4. `onUnLoad` 在从视图中删除表单时（在窗口关闭时）运行。
5. `onResize` 在修改窗口高度或宽度时运行。
6. `onBeforePrint` 在将表单发送到打印机之前运行，允许在将表单发送到打印机之前进 行更改或修改。
7. `onAfterPrint` 在将表单发送到打印机后运行，一旦将表单发送到打印机，允许将表单 还原回上一个视图。
8. `onContextMenu` 当用户单击鼠标右键以显示上下文菜单项时运行。
9. `onKeyUp，onKeyDown` 在键盘释放或按下运行
10. `onMouseOver、onMouseUp、onMouseDown、onMouseMove` 在鼠标移动到表单主体，释放左键，左键按下，移动鼠标时运行。

在创建表单的时候，生成了命名控件的HTML页面。使用客户端 `JavaScript` 方法来更改控件的**样式或可见性**。通过 `getFieldByName` 获取对应的**DIV元素**

控件隐藏及显示 JavaScript

```javascript
var el = document.getElementById("MainDataForm").address_state;
el.style.visibility = 'hidden'


var el_allow = getFieldByName("cn_allowed_chk").querySelector("input");
var el_num = getFieldByName("cn_allowed_weeknum").querySelector("input");
console.log(el_allow);
console.log(el_num);
el_allow.disabled = "disabled";
el_num.disabled = "disabled"
```

设置和获取字段属性值

`this.value` 仅在**HTML控件**中设置或获取值

`document.thisItem.getProperty` 在**XML Cache** 中检索值

`document.thisItem.setProperty` 将值设置为 **XML Cache**，HTML 控件未更新

`window.handleItemChange` 将值设置未 **XML Cache**，更新 HTML 控件。每个 Aras Innovator 表单都包含一个名为 handleItemChange 的预定义 JavaScript 窗口 函数，该函数将在**缓存中设置属性的值，并以编程方式更新 HTML 控件**。

语法 JavaScript

```javascript
var topWindow = aras.getMostTopWindowWithAras(window);
var workerFrame = topWindow.work;
var gridXmlCallback = function() {
    return workerFrame.grid.getXML(true);
};
aras.export2Office(gridXmlCallback, 'export2Excel', undefined, workerFrame.itemTypeName);
```

异步 Async

demo JavaScript

```javascript
var applyWithSpinner = async function(itm) {
  aras.browserHelper.toggleSpinner(document, true);
  var res = await itm.applyAsync("方法名");
  aras.browserHelper.toggleSpinner(document, false);
  return res;
}

// Create a query we know will take a while like querying for all Methods in the system
var longQueryItem = aras.IomInnovator.newItem("Method", "get");
var res = await applyWithSpinner(longQueryItem);
alert("Performed query and found " + res.getItemCount() + " Methods");
```

demo2 JavaScript

```javascript
try{
  var method = inn.newItem("Method", "get");
  method.setProperty("mode", sp[1]);
  method.setProperty("path", pathAry[sp[0]].label);
  method.setProperty("pathId", pathAry[sp[0]].id);
  method.setProperty("id", thisItem.getID());
  method.setProperty("type", thisItem.getType());
  method.applyAsync("SHSTaskVoteByLoginUser").then(x => {
    console.log(x);
  }, rej => {
    var msg = rej.responseXML.querySelector("faultstring").text
    aras.ALertError(msg);
  })
}catch(ex){
  console.log(ex)
  aras.ALertError(ex);
}
```

控件事件

提前筛选 `Item` 字段类型（**远程搜索**）

重写为Item字段生成预输入的默认服务器调用的示例代码 JavaScript

```javascript
// Get the field we want to populate dynamically
var item = getFieldComponentByName('owned_by_id');

// Override the default server call
item.component.request = function() {
  var itemType = this.state.itemType;
  var maxCount = this.state.maxItemsCount;
  var label = this.state.label;

  var req =
  '<Item type="' + itemType + '" select="keyed_name" maxRecords="' + maxCount + '" action="get">' +
    '<keyed_name condition="like">' + label + '*</keyed_name>' +
    // Add our extra filters to this call
    '<is_alias>1</is_alias>' +
  '</Item>';

  return ArasModules.soap(req, {async: true});
}
```

填充下拉列表

动态填充下拉列表 JavaScript

```javascript
// Get the field we want to populate dynamically
var dropdown = getFieldComponentByName('authoring_tool');

/* Construct a filtered list that contains only Microsoft Office authoring tools */
var listItm = aras.IomInnovator.newItem("List", "get");
listItm.setAttribute("select", "id");
listItm.setProperty("name", "Authoring Tools");

var itms = aras.IomInnovator.newItem("Value", "get");
itms.setAttribute("select", "value,label");
itms.setPropertyItem("source_id", listItm);
itms.setProperty("value", "Microsoft%");
itms.setPropertyAttribute("value", "condition", "like");
itms = itms.apply();

// List that will be passed into the dropdown control for typeahead functionality
listOfValues = [{
    label: '',
    value: ''
}];

// Add an entry to our list for each Office tool
for (var i = 0, itmCount = itms.getItemCount(); i < itmCount; i++) {
    var itm = itms.getItemByIndex(i);
    listOfValues.push({
        label: itm.getProperty("label"),
        value: itm.getProperty("value")
    });
}

// Pass this list into our dropdown field
dropdown.component.setState({list: listOfValues});
```

网格事件

网格事件是网格控件的事件，在关系选项卡区域下的页签。网格事件发生在行和单元格，像 `Server Events` 一样，绑定一个方法作为事件的回调，作为 `RelationshipType` 项目上的 `Grid event` 关系，以及作为 `Property` 上的Grid Event 关系

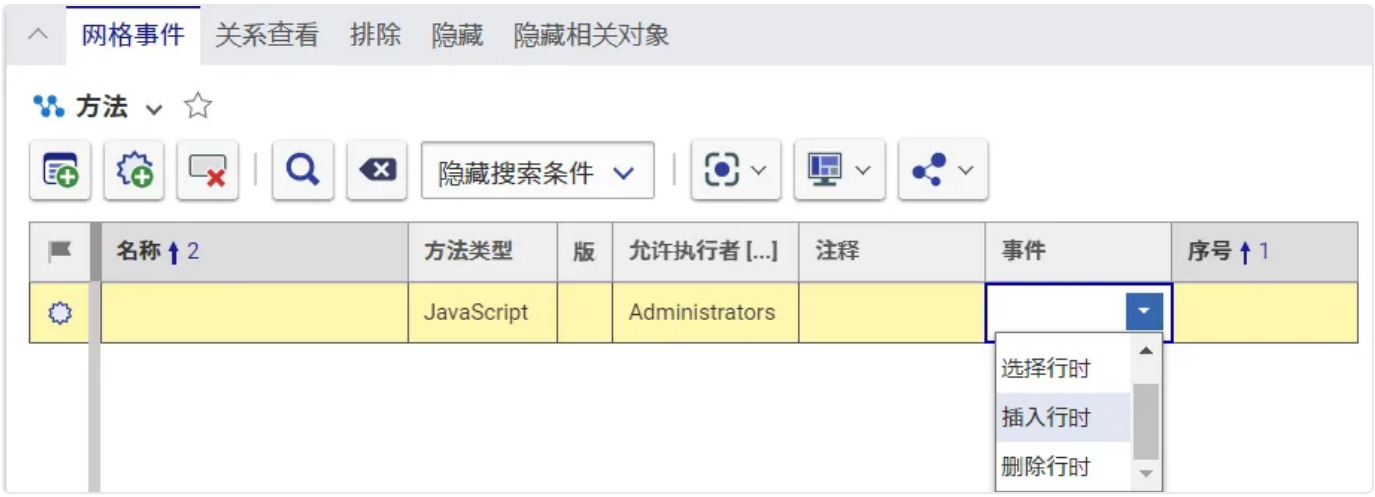
Method至少有三个参数 `relationshipID` 、 `relatedID` 、 `gridApple`

- `relationshipID` 是所选行的Item的ID

- `relatedID` 是Item的ID
- `gridApplet` 是网格控件对象的句柄

如果没有相关的Item， `relatedID` 可能为空。0

行事件

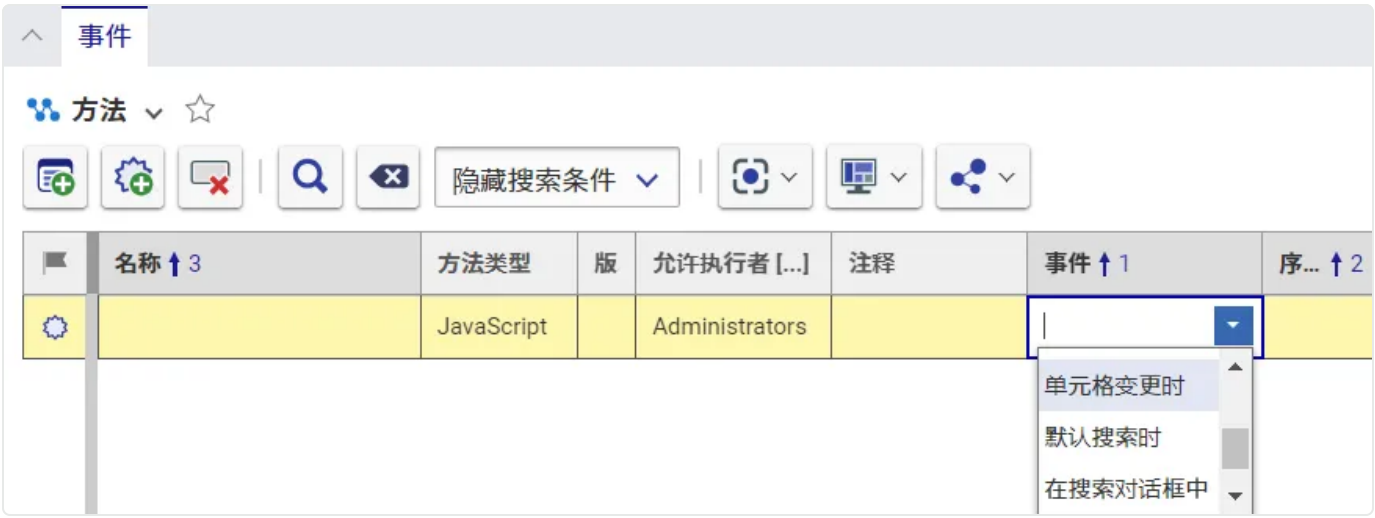


- `onSelectRow` 行被选中时触发
- `onInsertRow` 行在插入时触发
- `onDeleteRow` 行被删除时触发

在方法被调用时会携带3个参数

- `relationshipID` 关系Item的ID，也是网格控件所选择的行ID
- `relatedID` 相关Item的ID，如果关系中没有相关的Item，则 `relatedID` 可能为空
- `gridApplet` 网格控件的句柄

单元格事件



1. `onEdotStart` 当单元格获得焦点时触发
2. `onEditFinish` 当单元格失去焦点时触发
3. `onChangeCell` 当单元格值改变时触发。
4. `onSearchDialog` 打开搜索弹框时触发

5. `Default search` 默认搜索事件

事件方法被调用时携带的五个参数：

参数	类型	描述
relationshipID	String	关系Item的ID，也是网格控件的行ID
relatedID	String	相关Item的ID，如果关系行没有相关Item，则为空
gridApplet	GridControl	网格控件的句柄
propertyName	String	所选单元格的属性名称
colNumber	Integer	在网格中的列位置编号

在grid上修改值

案例 JavaScript

```javascript
var prt = typeof(parent.document.item) == "object" ? parent.document : parent.parent;
var tmpThisItem = typeof(parent.document.thisItem) == "object" ? parent.document.thisItem : parent.thisItem;
var NewRels = prt.item.selectNodes("Relationships/Item[@type='" + relationshipTypeName + "' and @isTemp='1']");
if (!NewRels || NewRels.length < 2)
  top.aras.getItemRelationshipsEx(prt.item, relationshipTypeName);

var maxVal = 0;
var rels = tmpThisItem.getRelationships(relationshipTypeName);
var count = rels.getItemCount();
for (var i = 0; i < count; i++) {
  var rel = rels.getItemByIndex(i);
  var sort_order = parseInt(rel.getProperty("sort_order"));
  if (sort_order > maxVal) {
    maxVal = sort_order;
  }
}


var thisRel = tmpThisItem.getItemsByXPath("//Item[@id='" + relationshipID + "']").getItemByIndex(0);
thisRel.setProperty("sort_order", maxVal + 1);
thisRel.setProperty("quantity", maxVal + 1);
grid.items_Experimental.set(relationshipID, "value", "sort_order_D", maxVal + 1);
grid.items_Experimental.set(relationshipID, "value", "quantity_D", 1000);
```

关系页签

获取页签标题 JavaScript

```javascript
var relationshipTabInfo = parent.relationships.relTabbar;
var currentRelationshipTabId = relationshipTabInfo.GetSelectedTab();
var tabLabel = relationshipTabInfo.GetTabLabel(currentRelationshipTabId);
```

对话框

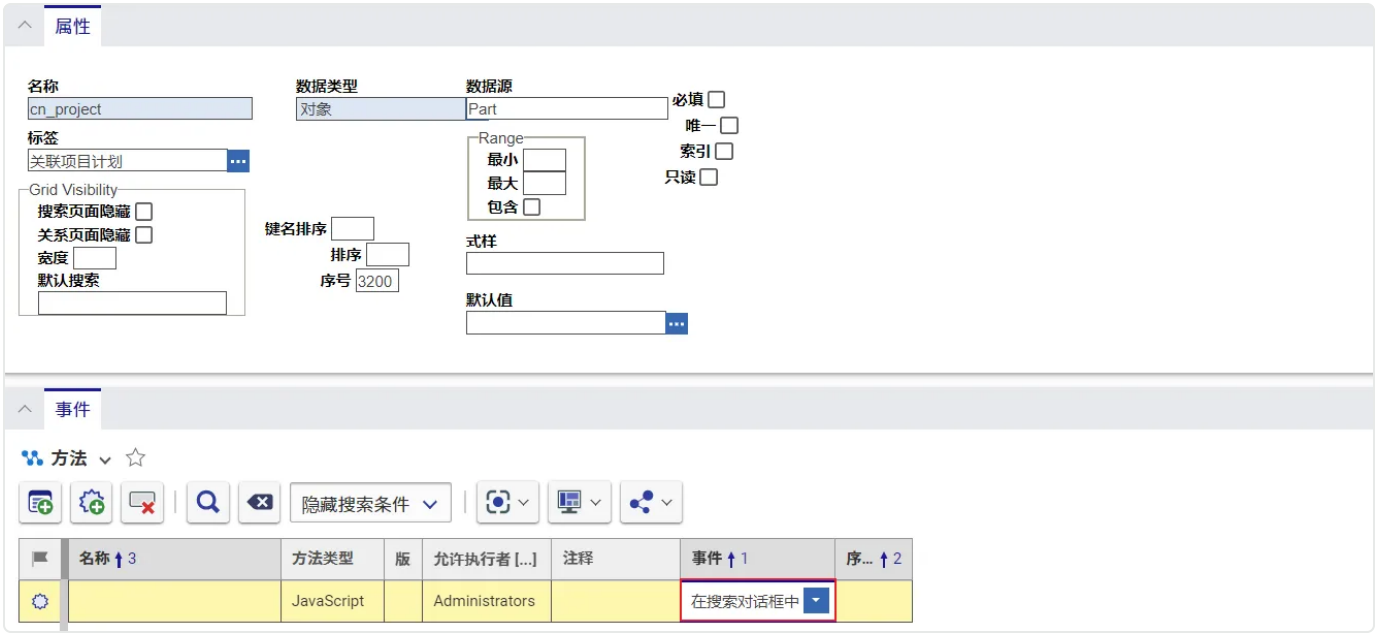
`aras` 授权对话框访问的aras函数，**每个对话框都需要这个参数。**

搜索对话框（OnSearchDialog）

搜索对话框，它允许用户搜索、选择和返回一个或多个项。通过**配置**或**代码**来打开对话框

配置

在属性为 `Item` 的字段的，可以通过事件来对搜索对话框进行卡控。



代码

参数说明：

1. `itemtypeName` 打开对话框的类型名称
2. `type` 打开对话框的；类型
3. `itemtypeID` 打开搜索框的 `itemType` 的ID
4. `handler` 处理对话框返回值的函数
5. `sourceItemTypeName` 父级ItemType
6. `sourcePropertyName` 指定父 `ItemType` 的名称和包含事件的属性名来触发属性的 `onSearchDialog` 事件
7. `multiselect` 指示用户是否可以多选

搜索对话框 JavaScript

```
var param = {
  aras: top.aras,
  type: 'SearchDialog', // 搜索对话框
  dialogWidth: 700,
  dialogHeight: 450,



  itemtypeName: 'Part', // 指定Item类型
  sourceItemTypeName:'Activity2',     // 参照父类
  sourcePropertyName: "managed_by_id" // 调用父类指定字段的onSearchDialog事件，对其进行卡控。
};

function callback(res) {
  if (res) {
    var itemNumber = res.keyed_name;
    alert("Part #" + itemNumber + " was selected");
  }
}

var topWnd = top.aras.getMostTopWindowWithAras();
var wnd = topWnd ? topWnd : window;
wnd.ArasModules.MaximazableDialog.show('iframe', param).promise.then(callback);
```

筛选

1，按列名筛选，添加指定的过滤条件，`%` 可以进行模糊匹配，`isFilterFixed` 为true时，无法对过滤条件进行修改。**无法包含 `or` 运算符。**

按列名筛选code JavaScript

```
var Filter = {};
Filter["name"] = { filterValue: "%Admin", isFilterFixed: false };
Filter["is_alias"] = { filterValue: "1", isFilterFixed: true };
return Filter;
```

2，按ID筛选，此条件的筛选，无法编辑搜索条件

按ID筛选code JavaScript

```javascript
var inn = this.getInnovator();
var ident = inn.newItem("Identity", "get");
var or = ident.newOR();

// Search for our Admin Identities...
var and1 = or.newAND();
and1.setProperty("name", "%Admin");
and1.setPropertyAttribute("name", "condition", "like");
and1.setProperty("is_alias", "1");

// ... and also search for Super User
var and2 = or.newAND();
and2.setProperty("name", "Super User");
and2.setProperty("is_alias", "1");
ident = ident.apply();

// Get an array of IDs that match our criteria
var idArray = [];
for (var i = 0; i < ident.getItemCount(); i++)
{
    idArray.push(ident.getItemByIndex(i).getID());
}

// Pass those IDs into our query
inArgs.QryItem.item.setAttribute("idlist", idArray.join(","));
return;
```

3，按AML筛选，直接传入要过滤的查询结果，可以手动编辑搜索条件中的功能

按AML筛选code JavaScript

```javascript
var criteriaAml = `
  <Item type='Identity' action='get' page='1'
        select='classification' pagesize='25' maxRecords='' returnMode='it
emsOnly'>
    <OR>
      <name condition='like'>%Admin</name>
      <name condition='like'>Super User</name>
    </OR>
    <is_alias condition='eq'>1</is_alias>
  </Item>`;
inArgs.QryItem.loadXML(criteriaAml);
return;
```

4，按ID筛选，可以继续搜寻

按ID筛选Code JavaScript

```javascript
/*
Name : cn_AdvancedFilter
Comment : 进阶控制搜索清单(控制 idList, 但可继续搜寻)
Field :
Event : OnSearchDialog
*/
var inn = this.getInnovator();
var strIdLists = "";

var criteriaAml = `<AML>
    <Item type='re_Requirement' action='get'>
    <OR>
        <bcs_ref_current_state_label>开发中</bcs_ref_current_state_label>
        <bcs_ref_current_state_label>已评审</bcs_ref_current_state_label>
    </OR>
</Item>
</AML>`;
var itmOperaions = inn.applyAML(criteriaAml);
for(var i=0;i<itmOperaions.getItemCount();i++){
  var itmOperaionTemplateId = itmOperaions.getItemByIndex(i).getID();

  if(strIdLists==="")
  {
    strIdLists+=itmOperaionTemplateId;
    continue;
  }
  strIdLists+=","+itmOperaionTemplateId;
}
inArgs.QryItem.removeAllCriterias();//clearCriterias
inArgs.QryItem.setCriteria("id", strIdLists, "in"); //use "in" to allow
```

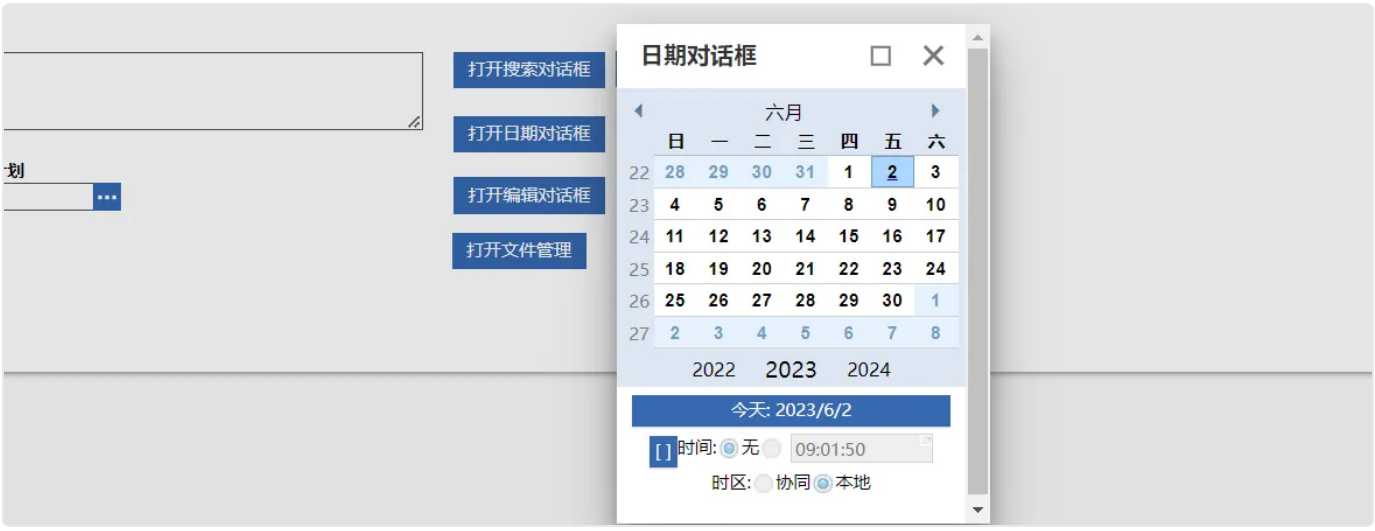
日期对话框（Date）

- `data` 打开日期对话框开始的默认日期
- `format` 将返回的字符串日期格式

```javascript
// Prompt the user for the baseline date
var dateFormat = "yyyy-MM-dd HH:mm:ss";
var baseDate = new Date().format(dateFormat);
params = {
    aras: top.aras,
    format: dateFormat,
    type: "Date",
    date: baseDate
};

// call the date dialog with defined params
return aras.getMainWindow().main.ArasModules.Dialog.show("iframe", params)
.promise.then(function(res) {
    // do something with the date on the dialog's callback
    return doSomethingWithTheDate(res);
});

// write a function to do some logic with the date
function doSomethingWithTheDate(dt) {
    alert(dt);
}
```



图片浏览（ImageBrowser）

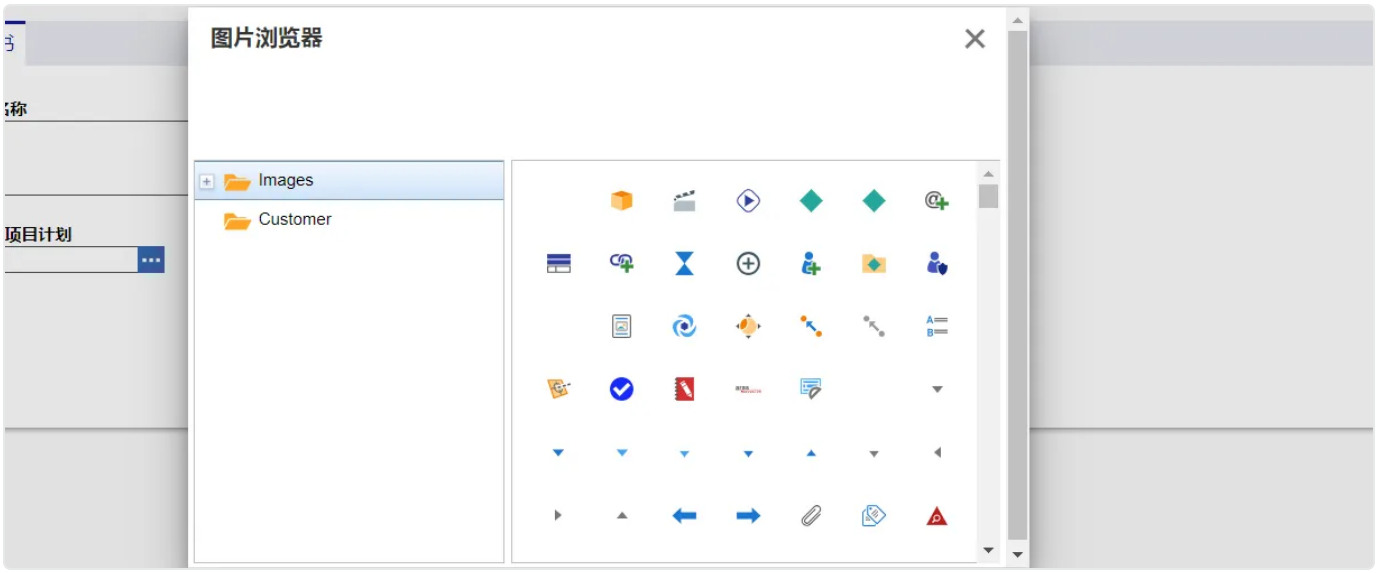
1. `showOnlyExternalFile` 一个布尔值，指示用户是否应该能够选择其中一个内部Aras图标

案例 JavaScript

```javascript
params = {
    type: "ImageBrowser",
    showOnlyExternalFile: true
};

// call the date dialog with defined params
return aras.getMainWindow().main.ArasModules.Dialog.show("iframe", params)
.promise.then(function(res) {
    // do something with the date on the dialog's callback
    return doSomethingWithTheDate(res);
});

// write a function to do some logic with the date
function doSomethingWithTheDate(dt) {
    alert(dt);
}
```



富文本对话框（HTMLEditorDialog）

此功能需要修改源文件，存在问题。

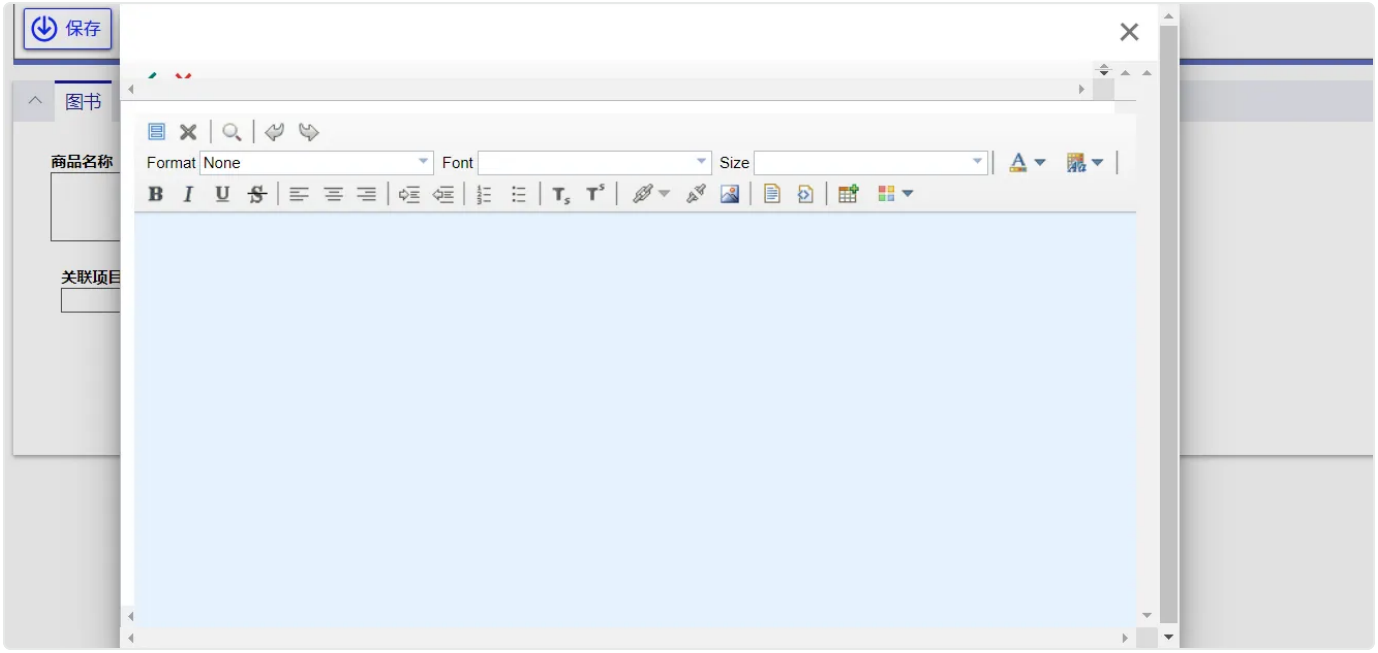
`sHtml`

编辑对话框 JavaScript

```
params = {
    aras: top.aras,
    type: "HTMLEditorDialog",
    sHtml: ''  // 指定源HTML的路径
};

return aras.getMainWindow().main.ArasModules.Dialog.show("iframe", params)
.promise.then(function(res) {
    return doSomethingWithTheDate(res);
});

function doSomethingWithTheDate(dt) {
    alert(dt);
}
```



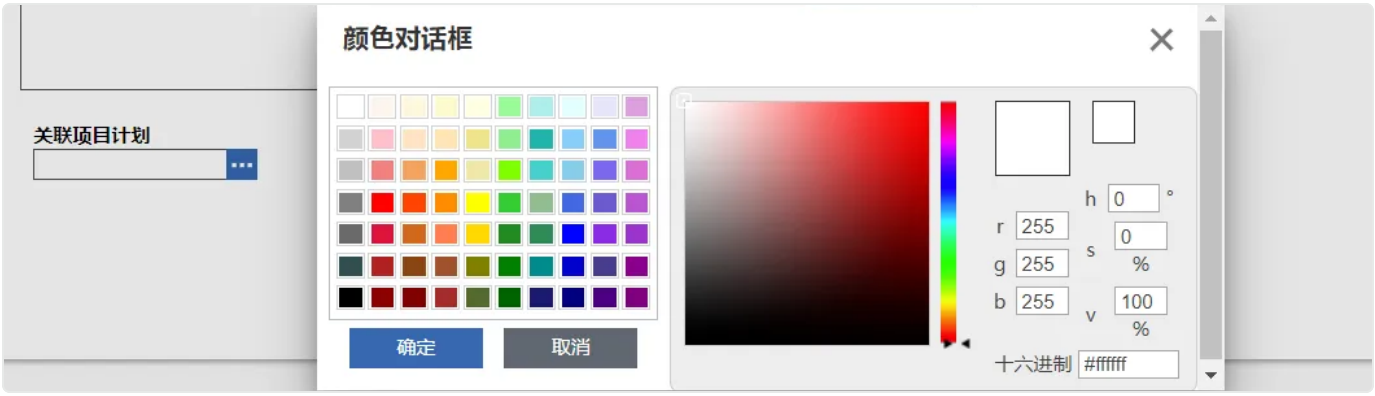
颜色选择（Color）

选择颜色 JavaScript

```javascript
params = {
    aras: top.aras,
    type: "Color",
};

return aras.getMainWindow().main.ArasModules.Dialog.show("iframe", params)
.promise.then(function(res) {
    return doSomethingWithTheDate(res);
});

function doSomethingWithTheDate(dt) {
    alert(dt);
}
```



文本对话框

1. `content` 默认显示内容
2. `isEditMode` boolean，是否可以编辑

文本对话框 JavaScript

```javascript
params = {
    aras: top.aras,
    type: "Text",
    content: '内容', // 默认显示内容
    isEditMode: false // 是否可以编辑
};

return aras.getMainWindow().main.ArasModules.Dialog.show("iframe", params)
.promise.then(function(res) {
    return doSomethingWithTheDate(res);
});

function doSomethingWithTheDate(dt) {
    alert(dt);
}
```

自定义对话框

打开窗体 JavaScript

```javascript
var formNd = top.aras.getItemByName("Form", "bwcs_UpdatePartState", 0);
// 设置参数
var param = {
  title: "Dialog Example",
  aras: top.aras,
  formId: formNd.getAttribute("id"),
  isEditMode: true,
  item: parent.thisItem,
  wnd: window,
  formType: parent.thisItem.getType()
};

var options = {
  dialogWidth: 300,
  dialogHeight: 250
};

var callback = {
  oncancel: function(dialogWrapper) {
    console.log(dialogWrapper);
  }
};

// Open the dialog
var topWnd = top.aras.getMostTopWindowWithAras();
var wnd = topWnd ? topWnd : window;
wnd.aras.modalDialogHelper.show('DefaultPopup', wnd, param, options, 'Show
FormAsADialog.html', callback);
```

打开自定义HTML JavaScript

```javascript
var inn = aras.newIOMInnovator();
var mainWindow = top.aras.getMainWindow();

var params = {
    title: '项目任务分派',
    aras: top.aras,
    content: '../customer/bwcs_Project ActiveAssign/index.html',
    item: parent.item,
    dialogWidth: 1000,
    dialogHeight: 650,
}
mainWindow.ArasModules.Dialog.show("iframe", params).promise.then(x => {

    console.log(x);
});
```

警告框

错误 `aras.AlertError("Something went very wrong");`

警告 `aras.AlertWarning("Something went only slightly wrong");`

成功 `aras.AlertSuccess("Something went right!");`

操作执行提示 将显示一个包含您的消息的窗口以及确定和取消按钮。如果用户单击“确定”，则该函数将返回 true，否则将返回 false。

执行操作 JavaScript

```javascript
aras.confirm("Are you sure you want to delete this item?");
```

UX

打开表单（uiShowItem）

使用代码的方式打开一个表单 `aras.uiShowItem('itemtype', id)`

案例 JavaScript

```javascript
var itemtype = control.datastore.get('af_itemtype').value;
var keyed_name = control.datastore.get('af_keyed_name').value;
if(keyed_name == null || keyed_name == '') return;
var inn = aras.newIOMInnovator();
var item = inn.getItemByKeyedName(itemtype, keyed_name);
if(item) aras.uiShowItem(itemtype, item.getID());
else aras.AlertWarning("Item not Found");
```

打开表单并编辑 JavaScript

```javascript
// 表单打开并编辑
aras.uiShowItem("Part", "303662F762B84046A87F14AB0FF69AD9", "tab view", false).then(x => {
  // 编辑
  x.onEditCommand()
})

// 如果是新建表单
var inn = top.aras.newIOMInnovator()
// 需要指定action = add
var part = inn.newItem("Part", "add");
part.setProperty("name", "新建表单");
aras.uiShowItemEx(part.node, "tab view", false)
```

新建表单（uiShowItemEx）

新建一张表单，可以未这张表单填写默认数据 `aras.uiShowItemEx(node, undefined, true)`

案例，新建表单并添加默认数据 JavaScript

```
var currItem=this.node;
var inn = this.getInnovator();
var id = this.getID();
if(id == "")
{
    id = thisItem.getID();
}
var itm = inn.getItemByKeyedName("ItemType","Part");
//查找当前Item对应的所有栏位
var itmProperties = inn.newItem("Property","get");
itmProperties.setProperty("source_id",itm.getID());
itmProperties = itmProperties.apply();
//不需要复制的栏位
var IgnoreProperties = ["bcs_autonumber","created_on","effective_date","modified_on","release_date","superseded_date","created_by_id","major_rev","modified_by_id","generation","state","item_number","permission_id","ctw_ljzt","ctw_datecode","ctw_wlzt","ctw_bzfs","ctw_khbin","keyed_name","id","config_id", "units", "sxw_bzff", "sxw_hbsx", "sxw_jlzt", "sxw_jygy", "description", "sxw_fzxs"];

var t_part=new Item("Part","add");
//共用属性栏位赋值
for(var i = 0;i < itmProperties.getItemCount();i++)
{
    var itmProperty = itmProperties.getItemByIndex(i);
    var strPropertyName = itmProperty.getProperty("name","");
    if(IgnoreProperties.indexOf(strPropertyName) >= 0)
    {
        continue;
    }
    var strPropertyValue = top.aras.getItemProperty(currItem,strPropertyName);//取值
     if(strPropertyName=="ctw_wfxh" && this.getProperty("ctw_wfxh","")!==""){

        var wf=new Item("Part","get");
        wf.setProperty("id",this.getProperty("ctw_wfxh"));
        wf=wf.apply();
        t_part.setPropertyItem(strPropertyName,wf); //赋值


    }else{
       t_part.setProperty(strPropertyName, strPropertyValue); //赋值
    }
}
```

```
39
40
41  var valuses =inn.newItem("value","get");
42  var listid="";
43
44  if(t_part.getProperty("classification") == "WF(晶圆)"){
45      t_part.setProperty("classification", "CP(中测)");
46  }else{
47      return aras.AlertError("料件不匹配，类型不属于WF(晶圆)");
48  }
49
50  //p料号默认值
51  listid="66BB2CD147634E0E9AAB539D770B99DC";
52  valuses.setProperty("source_id",listid);
53  valuses=valuses.apply();
54  //T料物料编码栏位赋值
55  for(var i=0;i<valuses.getItemCount();i++){
56      var proname=valuses.getItemByIndex(i).getProperty("label","");
57      var provalue=valuses.getItemByIndex(i).getProperty("value","");
58      t_part.setProperty(proname, provalue); //赋值
59
60  }
61
62  //--------------End--------------------
63  //新建PartBOM
64  var new_partbom=new Item("Part BOM","add");
65  new_partbom.setProperty("related_id", id); //赋值
    new_partbom.setProperty("permission_id", "5C07EB829D4241F6BB884952960FAF58"); //赋值
66  new_partbom.setProperty("quantity", "1"); //赋值
67  new_partbom.setProperty("sort_order", "1"); //赋值
68  //rel.appendChild(new_partbom);
69  //t_part.addRelationship(new_partbom);
70
71
72  aras.uiShowItemEx(t_part.node,undefined,true);
73
```

```C#
function onNewCommand() {
    // Calling uiNewItemEx for Project itemtype creates dialog in main window.
    // aras.utils.setFocus not working in chrome for parent window.
    // We need use window.open with empty url as 1-st argument and name of exits window as 2-nd argument. If window name is empty we need set temporary name.
    // window.open change opener property of opened window. After window.open calling we set old params to target window.
    // window.open must be called with context of the current window.
    if (itemTypeName === 'Project' && aras.Browser.isCh()) {
        var aWindow = aras.getMostTopWindowWithAras(window).opener;
        var lastOpener = aWindow.opener,
            lastName = aWindow.name;
        if (!aWindow.name) {
            aWindow.name = aras.generateNewGUID();
        }
        window.open('', aWindow.name);
        aWindow.opener = lastOpener;
        aWindow.name = lastName;
    }
    var newItm = aras.uiNewItemEx(itemTypeName);
    addRowToItemsGrid(newItm);
    return true;
}
```

文件

上传文件

客户端使用 `selectFile()` ，他根据用户选择一个File对象，也可以先将文件保存，将File的 `ID` 传递给服务端来处理。

文件上传案例 JavaScript

```javascript
var inn = aras.newIOMInnovator();

top.aras.vault.selectFile().then(async (fileObj)=>{
    let fileItem = inn.newItem('File', 'add');
    fileItem.attachPhysicalFile(fileObj);
    fileItem.setProperty('filename', fileObj.name, null);
    fileItem = await fileItem.applyAsync();

    console.log(fileItem);
});
```

文件下载

`downloadFile` 是从Aras Innovator下载文件到客户端机器。只能在客户端使用

- `fileNd` 要下载的文件XML节点，包含文件名，ID和所有相关属性
- `preferredName` 可选，下载时替换文件名的字符串

案例 JavaScript

```javascript
let item = aras.newIOMItem("File", "get");
item = item.apply();
aras.downloadFile(item.node, "sample.txt");
```

URL下载 C#

```csharp
var isSucceeded = aras.vault.downloadFile(fileURL);
if (!isSucceeded) {
    aras.AlertError(
        "item_methods_ex.failed_download_file"
    );
}
```

CUI

布局(Grid)

获取当前Grid

`aras.getMostTopWindowWithAras(window).work` 获取当前Grid

刷新当前Grid

```javascript
var topWindow = aras.getMostTopWindowWithAras(window);
var workerFrame = topWindow.work;
if (workerFrame && workerFrame.searchContainer) {
    workerFrame.searchContainer.runSearch();
}
```

刷新父窗口中的Grid

```javascript
var topWindow = parent.aras.getMostTopWindowWithAras(window);
var workerFrame = topWindow.work;
if (workerFrame && workerFrame.searchContainer) {…}
```

当前Grid内容导出Excel

```javascript
var topWindow = aras.getMostTopWindowWithAras(window);
var workerFrame = topWindow.work;
var gridXmlCallback = function() {
    return workerFrame.grid.getXML(true);
};
aras.export2Office(gridXmlCallback, 'export2Excel', undefined, workerFrame.itemTypeName);
```

Grid内容导出Word

```javascript
var topWindow = aras.getMostTopWindowWithAras(window);
var workerFrame = topWindow.work;
var gridXmlCallback = function() {
    return workerFrame.grid.getXML(false);
};
aras.export2Office(gridXmlCallback, 'export2Word');
```

菜单按钮中得到当前选中Item的ID

```javascript
var topWindow = aras.getMostTopWindowWithAras(window);

var workerFrame = topWindow.work;
alert(workerFrame.grid.getSelectedID());
// 获取所有选中的ids
alert(workerFrame.grid.getSelectedItemIDs(","));
```

关系页签

测试代码 JavaScript

```javascript
var tabbar = parent.relationships.relTabbar;
      tabID = tabbar.GetTabId("部門會簽");
var currTabID = parent.relationships.currTabID;
var FirstTabID = tabbar.GetTabId("Signoffs");

var tmpItem;
var DocApply_depRelId = top.aras.getItemFromServerByName("RelationshipType","DocApply_deplist","id").node.getAttribute('id');

var dep = "gm,ap,ep,sw,rd,hpla,po,ee,pa,cm,ed,lm,tt,pt,ad_mm,qc,ms_1,it,ad,hr,fa,fc"

if (srcElement.value !== "")
{
   var qcondition = "<itemtype>Arima_ECR_ECN</itemtype><id>"+ECNid+"</id><dep>"+dep+"</dep>";
   var Depinfo = top.aras.applyMethod("Find_Old_CountersignDep", qcondition);
   Depinfo = Depinfo.replace("<Result>","");
   Depinfo = Depinfo.replace("</Result>","");
   //return-> id:keyed_name,id:keyed_name,...

   var Dep_data = Depinfo.split(",");
   for (var i = 0;i<Dep_data.length ; i++)
   {
      var Dep_info = Dep_data[i].split(":");

      tmpItem = top.aras.newItem("DocApply_dep");
      top.aras.itemsCache.addItem(tmpItem);
      top.aras.setItemProperty(tmpItem,"dep_name",Dep_info[0]); // id

      var newRelship = top.aras.newRelationship(DocApply_depRelId,parent.item,false,null,tmpItem,"DocApply_dep",false,true,"DocApply_deplist");
   }
}

if (currTabID !== FirstTabID)
{
   top.document.frames[2].iframesCollection[tabID].contentWindow.location.reload();
}
```

aras. uiReShowItem

打开表单 C#

```
parent.findCurrentRelationshipsTab().document.getElementById('gridTD').children[1].getElementsByTagName('th')[ColumnIndex].setAttribute('style','display:none;');

top.aras.uiReShowItemEx(this.item.id,this.item);
```

aras.uiShowItemEx

通知 JavaScript

```
const itemNode = aras.newItem(target.type);
itemNode.setAttribute('id', target.id);
aras.uiShowItemEx(itemNode, 'notification');
```

aras.clearClientMetadataCache();

清楚客户端缓存

onLogoutCommand

退出登录 JavaScript

```
const topWindow = aras.getMostTopWindowWithAras(window);
if (topWindow.onLogoutCommand) {
  topWindow.onLogoutCommand();
}

```

onShowHistory

历史记录 JavaScript

```
window.tearOffMenuController.onShowHistory();
```

6，当前Grid导出World

语法 JavaScript

```
var topWindow = aras.getMostTopWindowWithAras(window);
var workerFrame = topWindow.work;
var gridXmlCallback = function() {
    return workerFrame.grid.getXML(false);
};
aras.export2Office(gridXmlCallback, 'export2Word');
```

7，循环Item对象内容

语法 JavaScript

```
if(item.isError() || item.getItemCount() == 0){
  return this;
}
for (var i = 0; i < item_color.getItemCount(); i++)
{
  //属性
  var item = item_color.getItemByIndex(i);
  alert(item.getProperty("ext_tab_color"));
  alert(item.getProperty("ext_tab_font_color"));
}
```

常用

获取主界面搜索参数

操作读取搜索界面查询参数。 C#

```
var wd = top.aras.getMainWindow()
wd.work.searchContainer._getSearchQueryAML()
```