GARGANTUA DATABASES - SCRIPTS

ADMINISTRATION TOOL

TABLE OF CONTENTS

INTRODUCTION

GARGANTUA Administration Tool allows you to create scripts to obtain custom features.

There are 4 types of scripts:

- Form scripts;
- Process scripts;
- IntelliScan scripts;
- Other scripts

Form scripts use JavaScript syntax, while process scripts use JavaScript syntax and Java objects. This document is intended for users with the necessary programming knowledge in these languages.

Creating a script at Gargantua database level

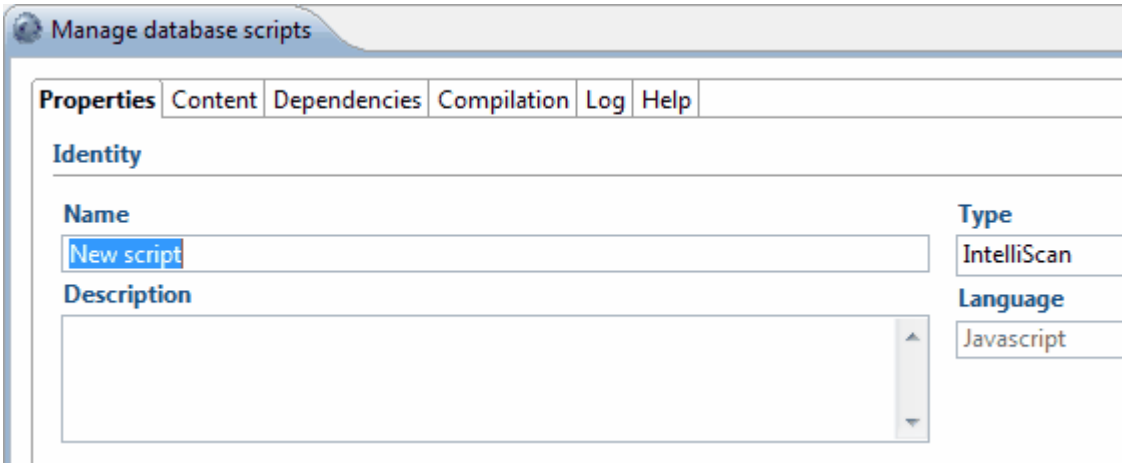
To create a centralized script at Gargantua database level, proceed as follows:

1. In the left pane of Gargantua Administration Tool, deploy the sub-branch corresponding to the Gargantua database where you want to create the script.
2. Click the **Scripts** sub-branch.

   The **Manage database scripts** screen displays in the right pane.

3. In the bottom left corner of the right pane, click the **New** button.

   A new script has been created and its management screen displays in the right pane.



4. In the **Properties** tab, enter a name and, if you want, a description for the script.
5. Choose a script type.
6. In the **Content** tab, enter the script code.
7. At the bottom of the right pane, click the **Compile** button to make sure that the script does not contain any errors.

   The **Compilation** tab opens.

8. If the **Compilation** tab contains error messages or warnings, you need to correct the script, for a script containing errors cannot be deployed. To correct an error, click the corresponding link in the list: you are redirected to the line of the script where the error is.

   If the **Compilation** tab does not contain any error messages or warnings, you can take the next step of the current procedure.

9. At the bottom of the right pane, click the **Deploy** button.

   The script is now available in the Gargantua database:
   - **Workflow** scripts are available in the **Scripts** tab of the processes management screens.

- **Form** scripts are available in the **Scripts** tab of the forms management screens.

Scripts management screen - Available tabs

The **Manage scripts** screen, which lists all the scripts in use in the current GARGANTUA database, contains the following tabs:

- The **Properties** tab allows you to view the main features of a script: name, description, type, and history.
- The **Content** tab contains a script editor which allows you to view and modify a script.
- The **Dependencies** tab lists the items used by a script.
- The **Compilation** tab lists the potential errors in a script.
- The **Summary** tab allows you to view a summary of a script.
- The **Log** allows you to view all the operations performed on a script, and to filter them according to some criteria.
- The **Help** tab allows you to access the contextual help.

"Properties" tab

The **Properties** tab allows you to name or rename a script (**Name** field), to type a description (**Description** field), to add it to a package (**Package** field), to define a type of script (**Type** drop-down menu) and to view the history of its deployed versions (**Deployment history** table).

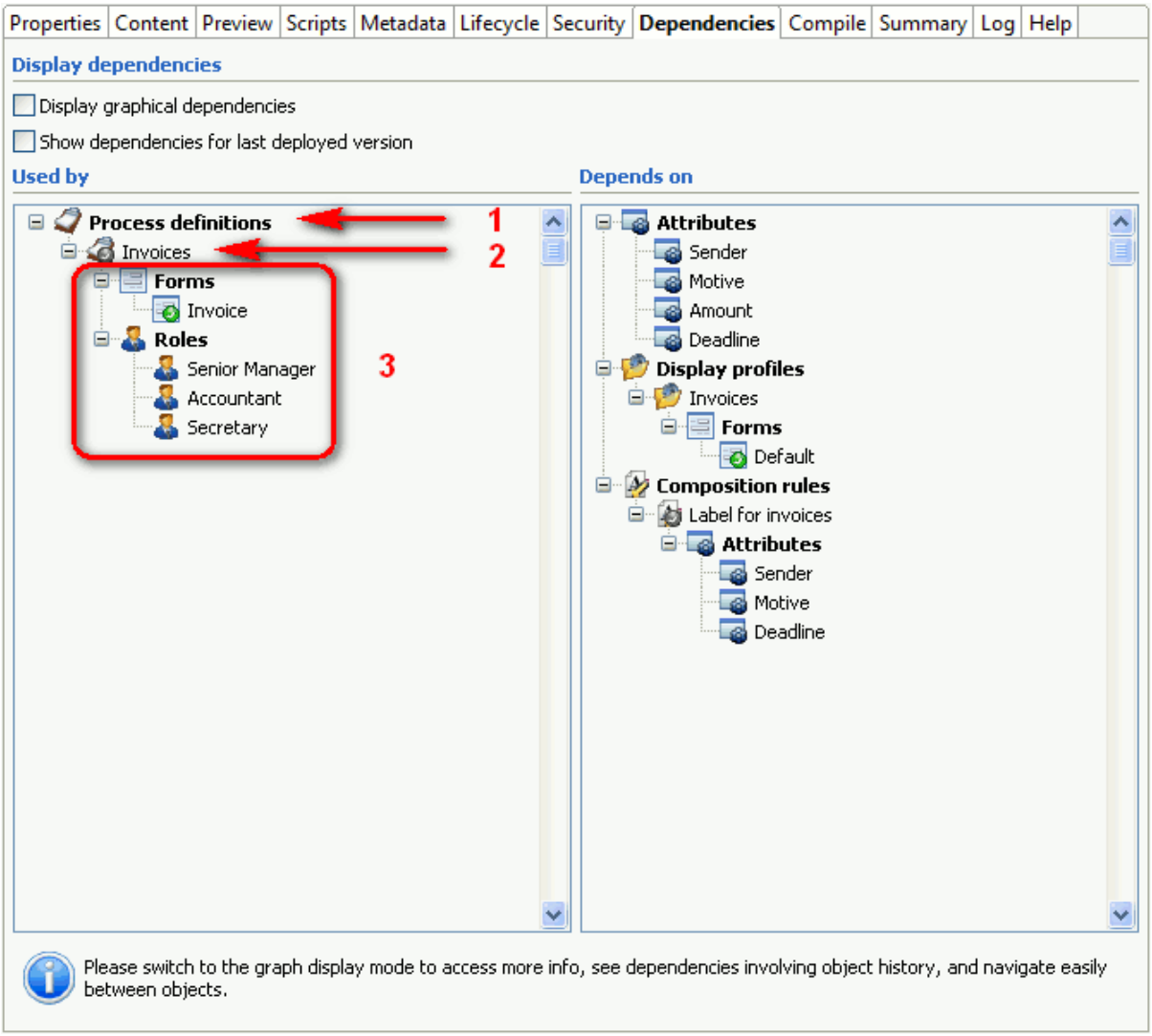
"Dependencies" tab

The **Dependencies** tab allows you to know which other objects rely on or are used by the current design object.

Displaying dependencies in list mode

By default, dependencies are displayed in **list mode** and the **working copies** of the objects are taken into account.

Depending on whether the current object is used by and/or relies on other objects, dependencies are displayed in 1 or 2 columns. If there are 2, the objects that rely on the current object are listed in the left column, and the objects that are used by the current object appear in the right column:



1. Some objects in this category have dependencies to the current object.
2. This object has a dependency to the current object, that is the “Invoices” process uses the “Invoice” form.
3. Here is the list of all objects that have a dependency to the “Invoices” process.

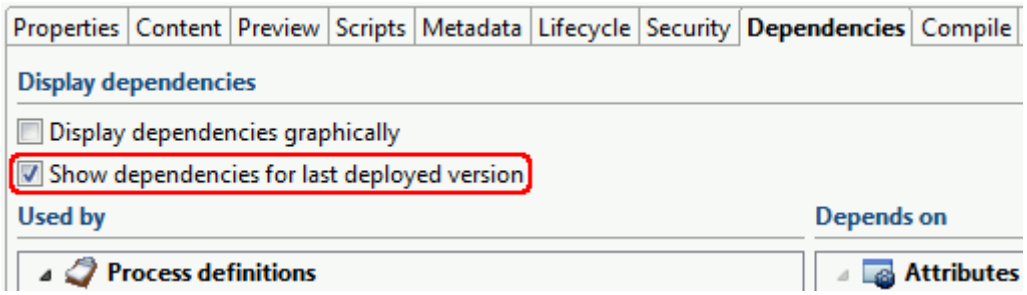
In the dependencies analysis screens as well as everywhere else in the interface, the icon indicates whether an object is currently deployed and, if so, whether the deployed version is identical to the working copy:

- No icon - The object is not deployed.
- - The object is deployed and there is no difference between its last deployed version and its working copy.
- - The object is deployed but its last deployed version is not identical to its working copy.

What is the "Show dependencies for last deployed version" option for?

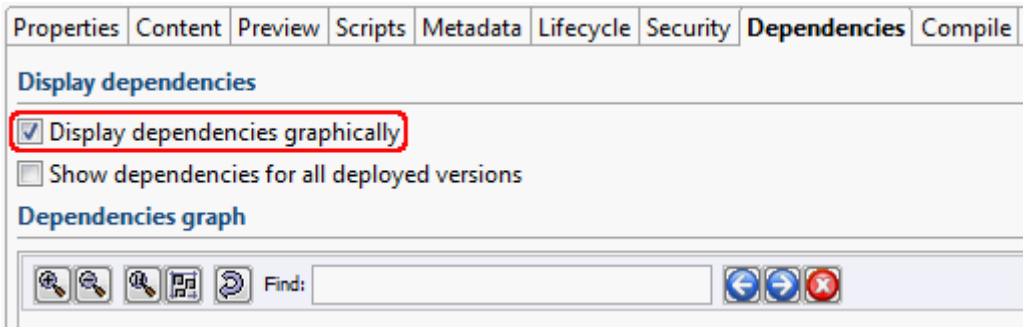
By checking the **Show dependencies for last deployed version** checkbox, you can choose to display the dependencies for the version of the object that is currently deployed, instead of the dependencies for the working copy.

This option is only available in list mode, for objects that support the **Deployment history** feature:

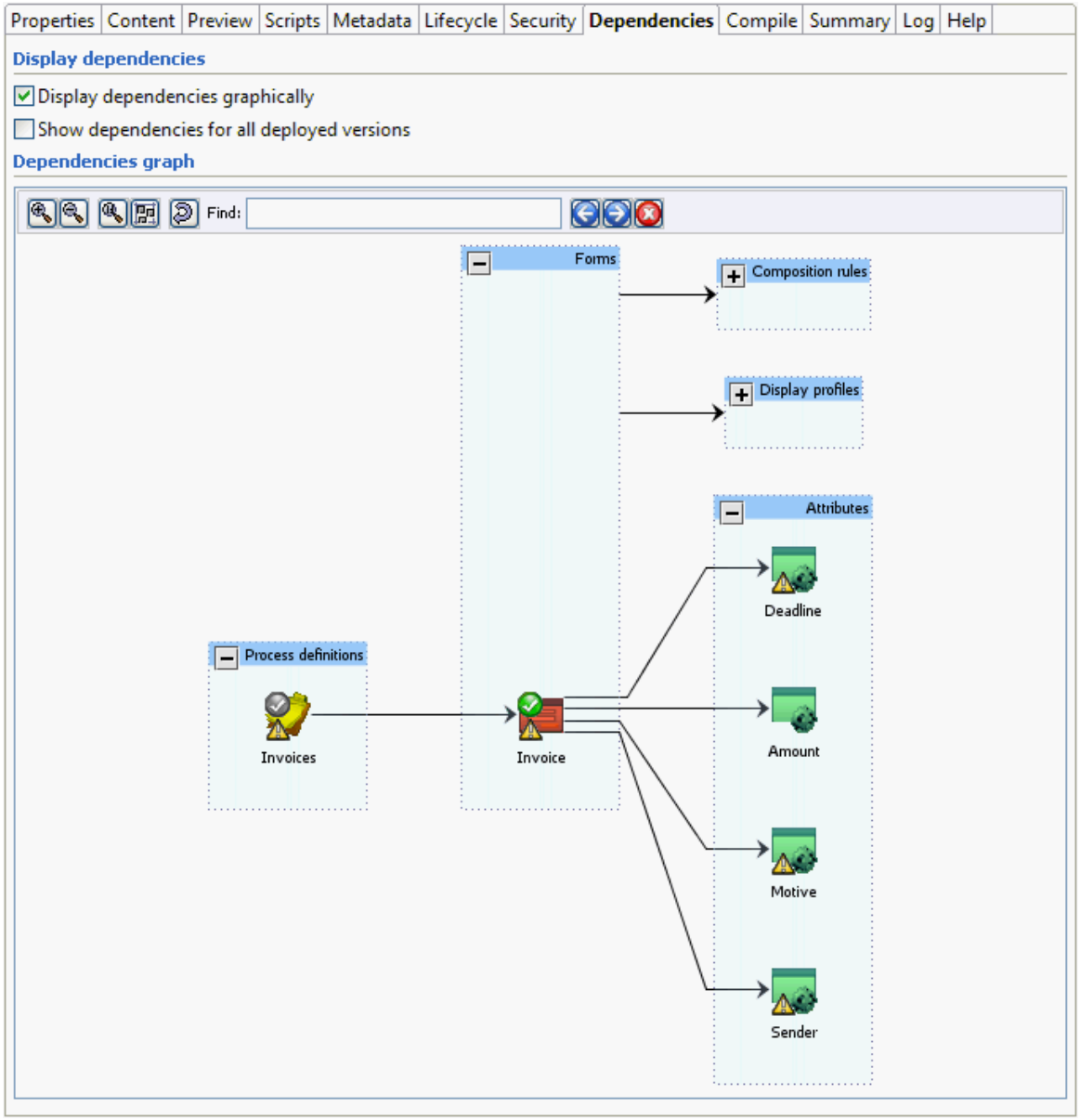


Displaying dependencies graphically

In order to switch from the list mode to the graphical display, check the **Display dependencies graphically** option:



The dependencies between the working copies of the various objects now take the form of a graph:



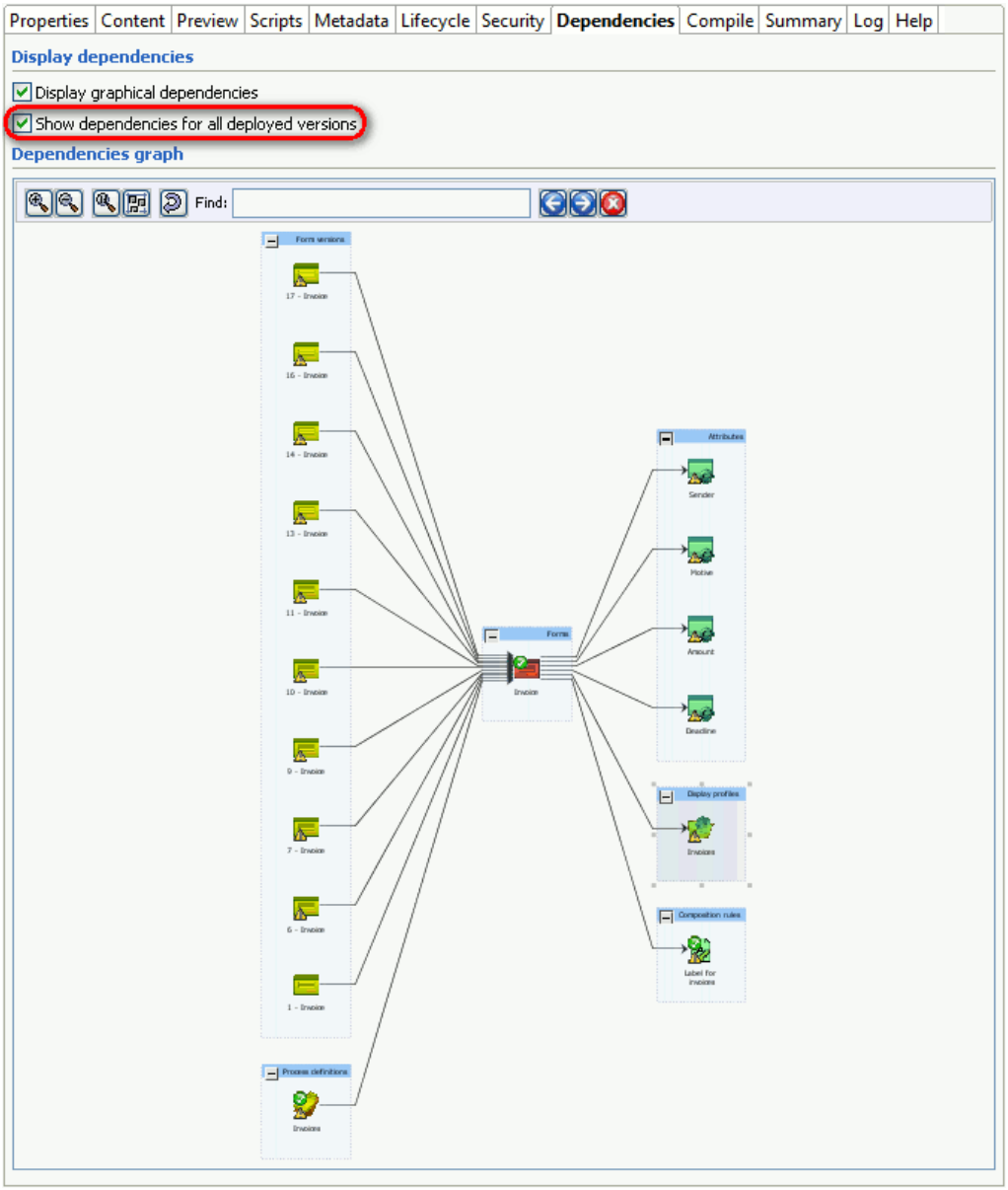
The toolbar above the graph allows you to:

- - Enlarge the graph;
- - Make the graph smaller;
- - Restore the default size for the graph;
- - Optimize the size of the graph;
- - Refresh the graph;
- Find: - Search the names of the objects in the graph for a particular string; the objects that match the search criterion are highlighted in pale blue;
- - - Navigate between the objects that contain the search string;
- - Reset the search results so that no more objects are highlighted.

What is the "Show dependencies for all deployed versions" option for?

If you check the **Show dependencies for all deployed versions** box, the dependencies graph will include all previously deployed versions of the objects, instead of just their working copies. If the current object supports the **Deployment history** function, then the list of all its previously deployed versions will also appear in the graph.

This option is only available when dependencies are displayed graphically:



"COMPILATION" TAB

The **Compilation** tab makes it possible to diagnose and solve errors and inconsistencies before an object is published. This feature is also available at database level, so that all objects can be compiled in a single move.

HOW AND WHEN TO USE THE GARGANTUA COMPILER

Before you deploy or redeploy a design object, you need to compile it in order to solve any potential errors or problems:

1. Click the name of the object you want to deploy in the left pane of the Administration Tool - or the name of a database, if you want to deploy all the objects it contains in a single move.

   The corresponding management screen appears in the right pane.

2. In the lower part of the right pane, click the **Compilation** button.

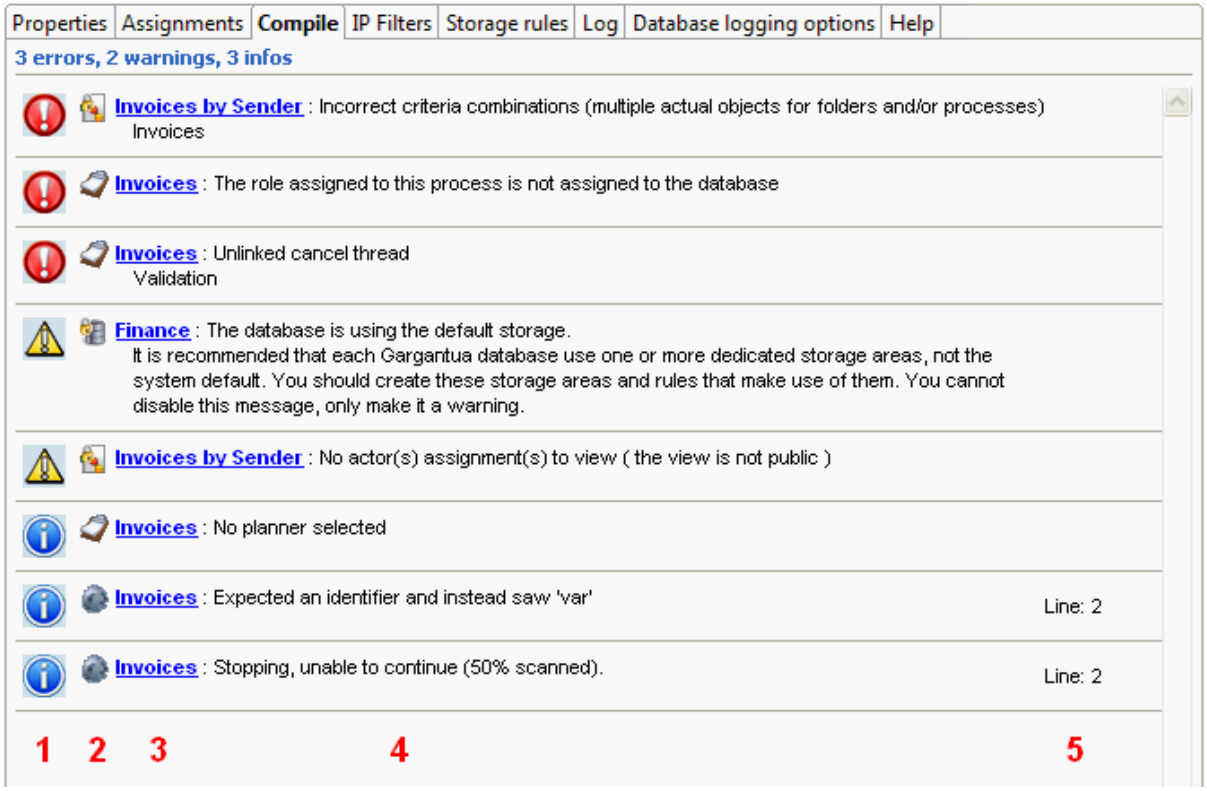
   The Compile tab opens and a list of messages appears.

3. If the list only contains success and/or information messages (see Structure of the messages list), you can proceed and deploy the object.

   On the other hand, if the list contains error and/or warning messages (see Structure of the messages list), you must solve the corresponding problems before you deploy the object.

Structure of the messages list

The list of the compiler's messages contains the following information:



1. Icons indicating the type of each message:

   - Success;

   - Information;

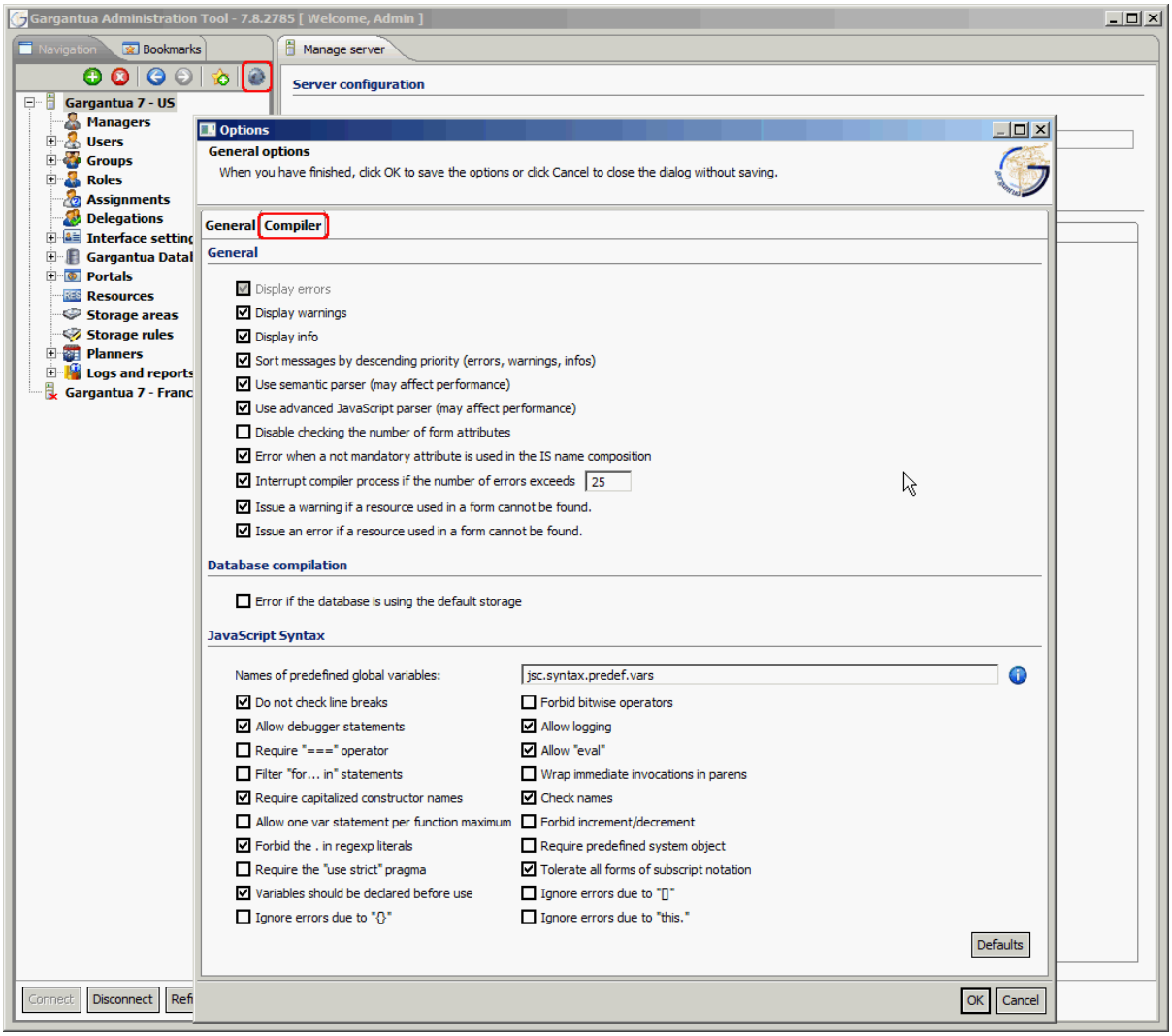
   - Warning;

   - Error;

2. Icons indicating the type of the objects the messages are about;
3. Links leading to the management screens where the problems can be solved;
4. Details about the current problem;
5. If the object is a script, line where the problem occurs.

CUSTOMIZING THE COMPILER'S BEHAVIOR

If you wish, you can customize the messages list (order, contents...) and the behavior of the JavaScript parser with the help of the **Options** dialog:



See the "General – Connection to Server" chapter of this manual for more information on the different options available.

"SUMMARY" TAB

The **Summary** tab allows you to display a summary of an item that presents its main features, like its manager, form, priority, content, etc.

The **Summary** tab is particularly helpful to have a general insight of the item, without having to browse through each tab.

The **Summary** tab also has a saving option in PDF format, which allows to view the summary of an item without

the GARGANTUA Administration tool.

"LOG" TAB

The **Log** tab allows you to view a list of all the operations performed in one of the branches of GARGANTUA Administration Tool.

You can filter your search through the following criteria:

- Start date, end date;
- User;
- Operation.

To filter by dates:

1. Fill in the date from which you want to view the operations performed, in the **Start date** field.
2. Fill in the date until which you want to view the operations performed, in the **End date** field.
3. Click on **Display**.
   The list of the operations performed between the two dates is displayed in the **Log entries** field.

> ***You can also click on the "Week audits", "Month audits", "Year audits" and "All audits" icons, so as to view the operations performed in larger periods of time.***

To filter by users:

1. Click on the magnifying glass, close to the **User** field.
2. Fill in the user name from whom you want to view the performed operations.
3. Click on **Display**.
   The list of the operations performed by the user is displayed in the Log entries field.

To filter by operation:

1. Click on the arrow of the **Operation** field.
   A list of the possible operations is displayed.
2. Click on the type of operation desired.
3. Click on **Display**.
   The list of the operations performed in relation with the selected type of operation is displayed in the **Log entries** field.

> ***You can tick the "Show login/logout events" box to display the users connections and disconnections. This functionality is only available for the "Users" and "Server log" branches.***
>
> ***You can tick the "Only Show error audits box" to only display the failed operations.***

The "Reset" button allows you to cancel the last filtering.

You can also export the list of the performed operations, in .csv format.

To export a list of the performed operations:

1. Generate a list, depending on the previous criteria.
2. Click on the **CSV Export** button.
3. Choose the place where you want to save the export.
4. Click on **Save**.
   You have exported a list of the performed operations.

Form scripts

What is a form script?

Forms in Gargantua 7 are generated in HTML.

A Gargantua form is a set of attributes whose presentation can be customized. Forms are designed using Gargantua 7 Administration Tool.

Each attribute of a form has a unique ID. The ID structure is: `DFATBBBBNNNNNNNN00000000000000000000000000`.

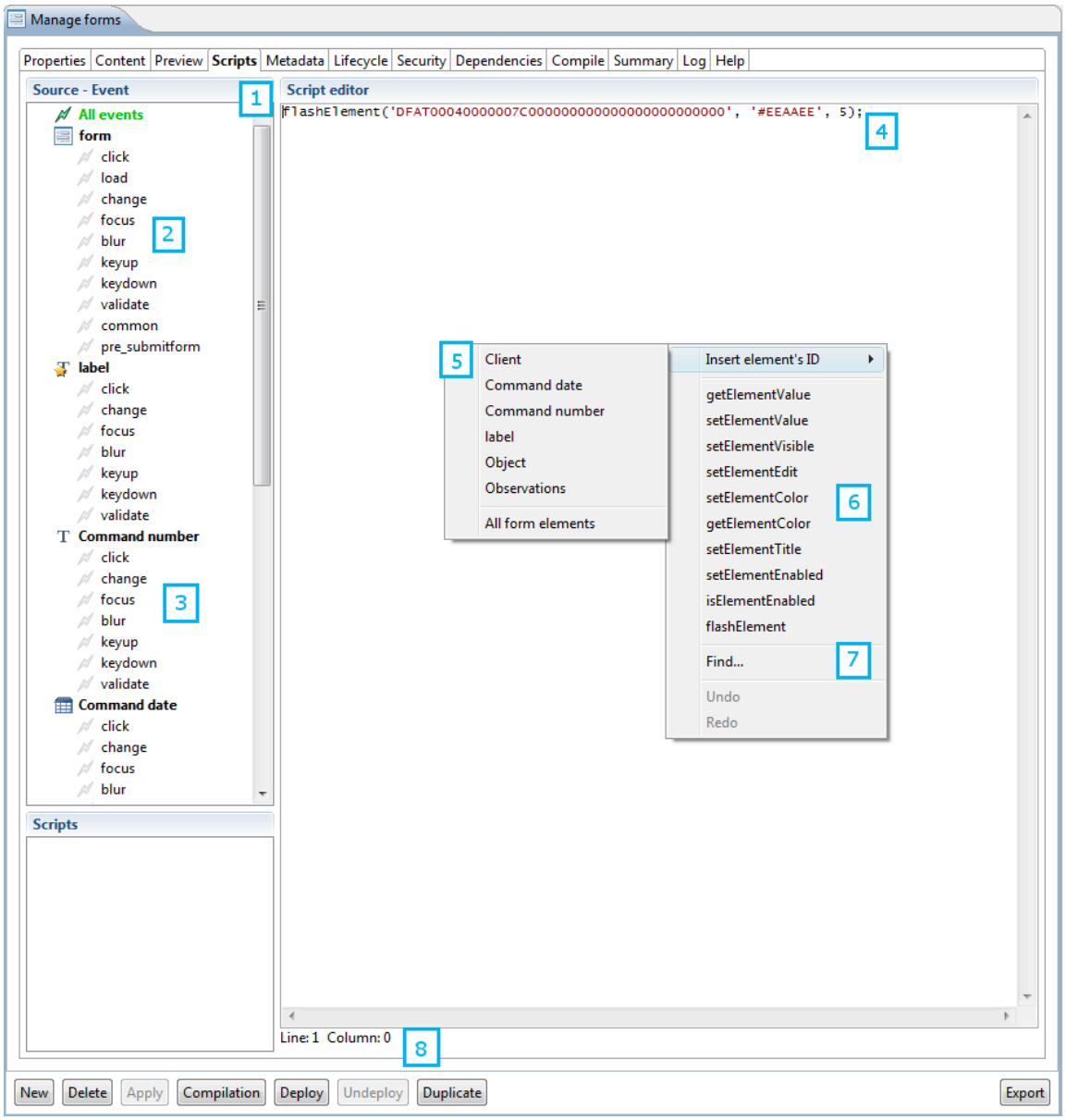
- `BBBB` is the database ID.
- `NNNNNNNN` is the unique ID of the attribute in the Gargantua database.

Gargantua forms are displayed in a browser. Thus, it is possible to run scripts written in JavaScript to manipulate the attributes through their IDs.

Form script editor

Gargantua Administration Tool includes a script editor with a wizard providing the JavaScript functions of the Gargantua API.

The right click opens a context menu that lists the JavaScript functions of the GARGANTUA API.



1. List of JavaScript events grouped by object (form + attributes);
2. List of JavaScript events for the form;
3. List of JavaScript events for a form attribute;
4. Script editor containing a GARGANTUA JavaScript API function call;
5. Wizard allowing to retrieve attribute IDs, accessible from the editor context menu;
6. Wizard allowing to generate GARGANTUA JavaScript API function calls, accessible from the editor context menu;
7. Text search function;
8. Line and column number indicator.

GARGANTUA API FOR FORMS

The GARGANTUA 8 server provides a set of JavaScript functions that allow you to modify the contents of a form or to perform a number of actions.

SCRIPT.NAME

```
//@script.name " <script_name>"
```

Allows to include a form script within another one, created in the **Scripts** tab.

For instance, we created a script named "library.script.message" in the Scripts tab. To link this script to the form script, we will enter the following code:

```
//@script.name "library.scripts.message".
```

SCRIPT.NO_CHECK

```
//@script.no_check
```

Requests the compiler to ignore compiler errors. The compiler uses a very strict grammar; some code, even if correct, may raise errors due to this.

When using this directive, please make sure you validate the script yourself or be prepared for u7nexpected errors.

JAVASCRIPT FUNCTIONS FOR FORM ELEMENTS

GFXGETATTRIDBYLABEL

```
function gfxGetAttrIdByLabel(label)
```

Retrieves the given attribute's ID based on the label.

Parameters

label

The label of the attribute whose id must be returned.

Return value

ID of the given attribute.

Exceptions

n/a

Example

```
window.alert(gfxGetAttrIdByLabel("label"));
```

will display

```
DFATxxxx00000001000000000000000000000000
```

gfxGetAttrLabelById

```
function gfxGetAttrLabelById(attributeId)
```

Retrieves the given attribute's label using it's ID.

Parameters

attributeId

The ID of the attribute whose label must be returned.

Return value

The label of the given attribute.

Exceptions

n/a

Example

```
window.alert(gfxGetAttrLabelById("DFATxxxx00000001000000000000000000000000"));
```

will display

```
label
```

getElementValue

```
function getElementValue(elementID)
```

Returns the value of the specified element.

This API function has been replaced by the gfxVal function; please use the new function whenever possible.

Parameters

elementID

The ID of the attribute whose value must be returned.

Return value

Value of the specified attribute.

Exceptions

n/a

Example

```
window.alert(getElementValue("DFATxxxx00000001000000000000000000000000"));
```

will display the current value of the label attribute.

getElementKey

```
function getElementKey(elementID)
```

Returns the key of the specified element if it is a keyed (translated) list or `null` otherwise.

Parameters

elementID

The ID of the attribute whose key must be returned.

Return value

Key of the specified attribute. If the attribute is not a keyed list, the current element value is returned.

Exceptions

n/a

Example

Let's assume we have the following two lists and two attributes of these types.

Simple list :

```
1
   1.1
   1.2
   1.3
2
   2.1
   2.2
   2.3
3
4
5
```

Keyed (translated) list :

```
1              one                 premier
   1.1            one.one             premier.premier
   1.2            one.two             premier.deuxième
   1.3            one.three           premier.troisième
2              two                 deuxième
   2.1            two.one             deuxième.premier
   2.2            two.two             deuxième.deuxième
   2.3            two.three           deuxième.troisième
3              three               troisième
4              four                quatrième
```

```
5               five                    cinquième
```

When the first value is selected, then

```
window.alert(getElementKey("DFAT00070000066000000000000000000000000"));
```

will return `1` for the first attribute and `1` for the second attribute, while

```
window.alert(getElementValue("DFAT00070000066000000000000000000000000"));
```

will return `1` for the first attribute and `premier` for the second attribute.

gfxVal

```
function gfxVal(element [, value])
```

`gfxVal` will either read or write the given attribute's value depeding on the presence of the `value` parameter

Parameters

element

The label of the attribute whose value must be returned or set.

value

(optional) The value to be set

Return value

Value of the specified attribute or undefined if the set variant is used.

Exceptions

n/a

Example

```
gfxVal("label", "test");
window.alert(gfxVal("label"));
```

will display `test`.

GFXENABLE

```
function gfxEnable(element)
```

Enables the control for the given attribute.

Parameters

element

The label of the attribute whose control must be enabled

Return value

None

Exceptions

n/a

Example

```
gfxEnable('liste simple');
```

gfxDisable

```
function gfxDisable(element)
```

Disables the control for the given attribute.

Parameters

element

The label of the attribute whose control must be disabled

Return value

None

Exceptions

n/a

Example

```
gfxDisable('liste simple');
```

GFXVISIBLE

```
function gfxVisible(element, visible)
```

Changes the visibility of the control for the given attribute.

Parameters

element

The label of the attribute whose control must be shown or hidden

visible

Boolean value to indicate visibility status

Return value

None

Exceptions

n/a

Example

```
gfxVisible('liste simple', false);
```

will hide the form control for the given attribute.

```
gfxVisible('liste simple', true);
```

will show the form control for the given attribute.

getElementValueEx

```
function getElementValueEx(elementID)
```

Returns the key of the specified element if it is a keyed (translated) list or the current value otherwise.

Parameters

elementID

The ID of the attribute whose key must be returned.

Return value

Key of the specified attribute. If the attribute is not a keyed list, the current element value is returned.

Exceptions

n/a

Example

Let's assume we have the following two lists and two attributes of these types.

Simple list :

```
1
   1.1
   1.2
   1.3
2
   2.1
   2.2
   2.3
3
4
5
```

Keyed (translated) list :

```
1              one                 premier
   1.1            one.one             premier.premier
   1.2            one.two             premier.deuxième
   1.3            one.three           premier.troisième
2              two                 deuxième
   2.1            two.one             deuxième.premier
   2.2            two.two             deuxième.deuxième
   2.3            two.three           deuxième.troisième
3              three               troisième
4              four                quatrième
```

```
5               five                    cinquième
```

When the first value is selected, then

```
window.alert(getElementValueEx("DFAT00070000066000000000000000000000000"));
```

will return `1` for the first attribute and `1` for the second attribute.

setElementValue

```
function setElementValue(elementID, value)
```

Changes the value of the specified element.

This API function has been replaced by the gfxVal function; please use the new function whenever possible.

Parameters

elementID

The ID of the attribute whose value must be changed.

value

New value.

Return value

None.

Example

```
setElementValue("DFATxxxx00000001000000000000000000000000", "test"));
```

will set the label attribute to the value `test`.

setElementKey

```
function setElementKey(elementID, value)
```

Sets the value of an keyed (translated) list attribute to the given key

Parameters

elementID

The ID of the attribute whose value must be changed.

value

New value.

Return value

None.

Example

Let's assume we have the following keyed (translated) list and an attributea of these type.

```
1               one                 premier
   1.1             one.one             premier.premier
   1.2             one.two             premier.deuxième
   1.3             one.three           premier.troisième
2               two                 deuxième
   2.1             two.one             deuxième.premier
   2.2             two.two             deuxième.deuxième
   2.3             two.three           deuxième.troisième
3               three               troisième
4               four                quatrième
5               five                cinquième
```

When the following code is executed

```
setElementKey("DFAT00070000006600000000000000000000000000", "1");
```

The value `premier` will be selected in the form control.

getElementColor

```
function getElementColor(elementID)
```

Returns the color of the specified element.

Parameters

elementID

The ID of the attribute whose value must be returned.

Return value

Color of the specified attribute or `undefined` if there is no custom color set for the control.

Example

```
window.alert(getElementColor("DFAT0007000000660000000000000000000000000"));
```

will display the color of the element or `undefined` if the color is not set.

setElementValueEx

```
function setElementValueEx(elementID, value)
```

Changes the value of the specified element using either the value or the key depending if it is a keyed (translated) list or not.

This API function has been replaced by the gfxVal function; please use the new function whenever possible.

Parameters

elementID

The ID of the attribute whose value must be changed.

value

New key or value.

Return value

None

Example

Let's assume we have the following keyed (translated) list and an attribute of this type.

```
1               one                 premier
   1.1             one.one             premier.premier
   1.2             one.two             premier.deuxième
   1.3             one.three           premier.troisième
2               two                 deuxième
   2.1             two.one             deuxième.premier
   2.2             two.two             deuxième.deuxième
   2.3             two.three           deuxième.troisième
3               three               troisième
4               four                quatrième
5               five                cinquième
```

```
setElementValueEx("DFAT00070000006700000000000000000000000000", "1");
```

will set the attribute to the value `one`.

setElementColor

```
function setElementColor(elementID, color)
```

Changes the color of the specified element.

Parameters

elementID

The ID of the attribute whose color must be changed.

color

New color.

Return value

None.

Example

```
setElementColor("DFAT00070000066000000000000000000000000", "#ff0000");
```

After executing the previous line, the following line

```
window.alert(getElementColor("DFAT00070000066000000000000000000000000"));
```

will display `#ff0000`

setElementColor2

```
function setElementColor2(elementID)
```

This API function has been deprecated in GARGANTUA 8. Please replace it when porting existing forms from older GARGANTUA versions.

Parameters

elementID

The ID of the attribute whose color must be changed.

color

New color.

all

Boolean indicating whether to change the background for all radio inputs in case of a radio element

Return value

None

setElementTitle

```
function setElementTitle(elementID, title)
```

Modifies the tooltip of the specified element.

Parameters

elementID

The ID of the attribute whose tooltip must be modified.

title

New tooltip.

Return value

None

Example

```
setElementTitle("DFAT00070000066000000000000000000000000", "TEST !");
```

setElementVisible

```
function setElementVisible(elementID, value)
```

Changes the visibility of the specified element (visible or invisible).

This API function has been replaced by the gfxVisible function; please use the new function whenever possible.

Parameters

elementID

The ID of the attribute whose visibility needs to be changed.

value

Visibility indicator (”true” or “false”).

Return value

None

Example

```
setElementVisible("DFAT0007000006600000000000000000000000", false);
```

setElementVisible2

```
function setElementVisible2(elementID)
```

This API function has been deprecated in Gargantua 8. Please replace it when porting existing forms from older Gargantua versions.

Parameters

elementID

The ID of the attribute whose visibility needs to be changed.

Return value

None

setElementEdit

```
function setElementEdit(elementID, value)
```

Modifies the edit state of the specified element (editable or read-only).

This API function has been replaced by the gfxEnable and gfxDisable functions; please use the new functions whenever possible.

Parameters

elementID

The ID of the attribute whose edit state must be changed.

value

Edit state indicator (”true” or “false”).

Return value

None.

Example

```
setElementEdit('DFAT0007000000010000000000000000000000000', false);
```

setElementEdit2

```
function setElementEdit2(elementID)
```

This API function has been deprecated in Gargantua 8. Please replace it when porting existing forms from older Gargantua versions.

Parameters

elementID

The ID of the attribute whose edit state must be changed.

value

Edit state indicator (”true” or “false”).

Return value

None

Example

```
setElementEdit2('DFAT00070000001000000000000000000000000', false);
```

setElementEnabled

```
function setElementEnabled (elementID, bValue)
```

Modifies the enable state of the specified element (enabled or disabled).

This API function has been replaced by the gfxEnable and gfxDisable functions; please use the new functions whenever possible.

Parameters

elementID

The ID of the attribute whose enable state must be changed.

value

Enable state indicator (”true” or “false”).

Return value

None.

Example

```
setElementEnabled('DFAT0007000000010000000000000000000000000', false);
```

isElementEnabled

```
function isElementEnabled (elementID)
```

Returns the enable state of the specified element (enabled or disabled).

Parameters

elementID

The ID of the attribute whose enable state must be returned.

Return value

The enable state of the attribute.

Example

```
window.alert(isElementEnabled('DFAT00070000006600000000000000000000000000'));
```

Will display either `true` or `false`.

disabledEditor

```
function disabledEditor(elementID)
```

This API function has been deprecated in GARGANTUA 8. Please replace it when porting existing forms from older GARGANTUA versions.

Parameters

elementID

The ID of the attribute whose edit state must be changed.

Return value

None.

Example

```
disabledEditor('DFAT0007000000010000000000000000000000000');
```

enabledEditor

```
function enabledEditor(elementID)
```

This API function has been deprecated in Gargantua 8. Please replace it when porting existing forms from older Gargantua versions.

Parameters

elementID

The ID of the attribute whose edit state must be changed.

Return value

None.

Example

```
enabledEditor('DFAT000700000001000000000000000000000000');
```

FORM_GETVALUES

```
function form_getValues()
```

Retrieves all the form values as a JSON object. The returned value also includes hidden attributes.

PARAMETERS

None.

RETURN VALUE

A JSON object with all form values.

EXAMPLE

```
window.alert(JSON.stringify(form_getValues()));
```

will display an output similar to this :

```
{
    "formid":"DFFR0007000000070000000000000000000000000",
    "pageid":"Défaut","parentid":"DATA0007000000000000000000000000000000000",
    "objectid":"",
    "command":"view",
    "target":"",
    "multi":"0",
    "search":"0",
    "param1":"",
    "param2":"",
    "param3":"",
    "param4":"",
    "param5":"",
    "DFAT0007000000010000000000000000000000000":"",
    "DFAT0007000000660000000000000000000000000":"",
    "DFAT0007000000670000000000000000000000000":""
}
```

form_getAttributeType

```
function form_getAttributeType(elementID)
```

Retrieves the numeric attribute type for the specified attribute.

Parameters

elementID

The ID of the attribute for which the numeric type ID must be retrieved

Return value

The numeric representation of the type ID for the specified attribute.

Example

```
window.alert(form_getAttributeType('DFAT0007000000660000000000000000000000000'));
```

will display `100`.

FORM_GETATTRIBUTETYPEID

```
function form_getAttributeTypeId(elementID)
```

Retrieves the attribute type for the specified attribute.

PARAMETERS

elementID

The ID of the attribute for which the type ID must be retrieved

RETURN VALUE

The string representation of the type ID for the specified attribute.

EXAMPLE

```
window.alert(form_getAttributeTypeId('DFAT0007000000660000000000000000000000000'));
```

will display `DFTY0007000000640000000000000000000000000`.

form_getPage

```
form_getPage()
```

Retrieves the name of the current form page.

Parameters

None

Return value

The name of the form page.

Example

```
var page = form_getPage();
```

MISCELLANEOUS JAVASCRIPT FUNCTIONS

FORM_GETVERSION

```
function form_getVersion()
```

Retrieves the version of the GARGANTUA server.

PARAMETERS

None

RETURN VALUE

A string representation of the GARGANTUA 8 server version.

EXAMPLE

```
window.alert(form_getVersion());
```

will display a string like `8.0.7955`.

FORM_LANG

```
function form_lang()
```

Returns the language code of the user interface.

PARAMETERS

None.

RETURN VALUE

Language code. fr: French; ar: Arabic; en: English; es: Spanish; ru: Russian; ro: Romanian; hu: Hungarian.

The GARGANTUA 8 does no longer implement arabic translations, so the "ar" return value is no longer used, even for forms that have been migrated from older versions of GARGANTUA.

EXAMPLE

FORM_GETDISPLAYMODE

```
function form_getDisplayMode()
```

Retrieves the current display mode of the form.

PARAMETERS

None.

RETURN VALUE

The current display mode of the form, which can be one of the following values (both symbolic and numeric constants can be used):

```
0 = DISPLAY_NORMAL
1 = DISPLAY_READONLY
2 = DISPLAY_DISABLEDINPUTS
4 = DISPLAY_DISABLEDBUTTONS
```

EXAMPLE

```
window.alert(form_getDisplayMode());
```

form_enableGroup

```
function form_enableGroup(group, enabled);
```

Enables or disables a group of controls at once.

The group property can be set for individual attributes or other form controls in the form designer.

The effect is the same as enabling or disabling individual controls by using gfxEnable and gfxDisable.

Parameters

group

The name of the group.

enabled

Enabled indicator (”true” or “false”).

Return value

None.

Example

```
form_enableGroup("myGroup", false);
```

FORM_SETGROUPVISIBLE

```
function form_setGroupVisible(group, visible);
```

Sets the visibility of a group of controls at once.

The group property can be set for individual attributes or other form controls in the form designer.

The effect is the same as showing or hiding individual controls by using gfxVisible.

PARAMETERS

group

The name of the group.

visible

Visibility indicator (”true” or “false”).

RETURN VALUE

None.

EXAMPLE

```
form_setGroupVisible("myGroup", false);
```

form_expandSection

```
function form_expandSection(id);
```

Expands a section in grid layout mode.

Parameters

id

The ID of the section.

Return value

None.

Example

```
form_expandSection("theSection");
```

FORM_COLLAPSESECTION

```
function form_collapseSection(id);
```

Collapses a section in grid layout mode.

PARAMETERS

id

The ID of the section.

RETURN VALUE

None.

EXAMPLE

```
form_collapseSection("theSection");
```

form_toggleSection

```
function form_toggleSection(id);
```

Toggles (expands or collapses) a section in grid layout mode.

Parameters

id

The ID of the section.

Return value

None.

Example

```
form_toggleSection("theSection");
```

form_toggleAllSections

```
form_toggleAllSections(group, showOrHide)
```

Toggles (expands or collapses) all sections (or the sections having the same group) in grid layout mode.

The group property can be set for sections in the form designer.

Parameters

group

(optional) The name of the group

showOrHide

Visibility indicator (”true” or “false”)

Return value

None

Example

```
form_toggleAllSections("myGroup", false);
form_toggleAllSections(null, true);
```

form_isSectionExpanded

```
function form_isSectionExpanded(id);
```

Retrieves the expanded status of the specified section.

Parameters

id

The ID of the section.

Return value

The expanded status of the specified section.

Example

```
var expanded = form_isSectionExpanded("mySection");
```

FORM_ADDSECTIONICON

```
function form_addSectionIcon(id, options)
```

Function that adds an icon to the form section identified by the given id.

PARAMETERS

id

The ID of the section.

options

A JSON object containing the following key-pairs:

- id - the icon id
- src - the icon source
- position - `start | end`
- className - a class to add to the icon
- title - the icon tooltip
- hidden - boolean indicating if the icon should not be visible
- width - the icon width
- height - the icon height.

RETURN VALUE

None.

EXAMPLE

```
form_addSectionIcon('section',{
    id : 'icon',
    src : '/images/plugins/iconpack/svg/bubble.svg#bubble',
    position : 'end',
    hidden : false,
    width : '3rem',
    height : '3rem'
});

form_setSectionIconStyle('icon3', {
    'fill' : 'green'
});
```

form_toggleSectionIcon

```
function form_toggleSectionIcon(id, showOrHide)
```

Function that toggles the visibility of the section icon.

Parameters

id

The ID of the section icon.

showOrHide

Boolean indicating if the icon should be visible or not.

Return value

None.

Example

```
form_toggleSectionIcon('icon');
```

form_removeSectionIcon

```
function form_removeSectionIcon(id)
```

Function that removes the section icon.

Parameters

id

The ID of the section icon.

Return value

None.

Example

```
form_removeSectionIcon('icon');
```

form_setSectionIconStyle

```
function form_setSectionIconStyle(id, cssData)
```

Function that sets the section icon's style.

Parameters

id

The ID of the section icon.

cssData

A JSON object containing the styles to be applied to the icon.

Return value

None.

Example

```
form_addSectionIcon('section',{
    id : 'icon',
    src : '/images/plugins/iconpack/svg/bubble.svg#bubble',
    position : 'end',
    hidden : false,
    width : '3rem',
    height : '3rem'
});

form_setSectionIconStyle('icon3', {
    'fill' : 'green'
});
```

FORM_STICKYSECTION

```
function form_stickySection(id [, options])
```

Positions a section as sticky at the top of the form.

Parameters

id

The section ID.

options

An object containing the following possible options:

- `top` or `stickyTop` parameter indicating the viewport top position when the section should become sticky. By default this is 0.
- `zIndex` or `stickyZIndex` parameter indicating the z-index property for the section. By default this is 10.

Return value

None.

Example

```
form_stickySection('s1');
```

or

```
form_stickySection('s1', { zIndex: 25 });
```

FORM_SECTIONSTOTABS

```
function form_sectionsToTabs(ids, tabId [, options])
```

Transforms the sections identified by the given IDs array into tabs, each section becoming a tab. The section headers will be removed.

Parameters

ids

An array of section IDs.

tabId

The ID of the newly created tabs wrapper.

options

An object containing the following possible options:

- `border` (true/false) parameter indicating weather the tab content should or not have a visible border. By default this is `true`.
- `sticky` (true/false) parameter indicating weather the tab navigation should be rendered as sticky. By default this is `false`.
- `top` or `stickyTop` parameter indicating the viewport top position when the tab navigation should become sticky. Should be specified only in combination with sticky true. By default this is 0.
- `zIndex` or `stickyZIndex` parameter indicating the z-index property for the tab navigation. Should be specified only in combination with sticky true. By default this is 10.
- `height` parameter allowing you to control the height of the tab content. When the content exceeds this height, the content area becomes scrollable.
- `minHeight` parameter allowing you to control the minimum height of the tab content.
- `maxHeight` parameter allowing you to control the maximum height of the tab content. When the content exceeds this height, the content area becomes scrollable.

Return value

None.

Example

```
form_sectionsToTabs(['s1', 's2', 's3', 's4'], 'mytab');
```

or

```
form_sectionsToTabs(['s1', 's2', 's3', 's4'], 'mytab', { border: false });
```

FORM_SECTIONGROUPSTOTABS

```
form_sectionGroupsToTabs(group_names_array, tab_id [, options])
```

Transforms the sections that have the groups specified in the groups array into tabs. This way a tab can contain multiple sections. The group names array should be unique (no duplicates). If any of the sections in a group has the visible mandatory marker then the tab will also have one.

Parameters

group_names_array

An array of group names.

tabId

The ID of the newly created tabs wrapper.

options

An object containing the following possible options:

- `border` (true/false) parameter indicating weather the tab content should or not have a visible border. By default this is `true`.
- `sticky` (true/false) parameter indicating weather the tab navigation should be rendered as sticky. By default this is `false`.
- `top` or `stickyTop` parameter indicating the viewport top position when the tab navigation should become sticky. Should be specified only in combination with sticky true. By default this is 0.
- `zIndex` or `stickyZIndex` parameter indicating the z-index property for the tab navigation. Should be specified only in combination with sticky true. By default this is 10.
- `height` parameter allowing you to control the height of the tab content. When the content exceeds this height, the content area becomes scrollable.
- `minHeight` parameter allowing you to control the minimum height of the tab content.
- `maxHeight` parameter allowing you to control the maximum height of the tab content. When the content exceeds this height, the content area becomes scrollable.
- `tabs` (array of objects) parameter provides custom options for each tab:
  - `id` - the id of the tab (if not provided it will be generated)
  - `name` - the name/label of the tab (if not provided it will be copied from the first section)
  - `onlyContent` (true/false) - indicates weather only the content of the section or the entire section (with header) should be added in the tab content (by default this is `false`).

Return value

None.

Example

```
form_sectionGroupsToTabs(['g1', 'g2', 'g3'], 'mytab', { border: false });
```

or

```
form_sectionGroupsToTabs(['g1', 'g2', 'g3'], 'mytab', {
  border: true,
  sticky: true,
  maxHeight: '200px',
  tabs: [{
    id: 'tab1',
    name: 'First tab'
  }, {
    id: 'tab2',
    name: '<strong>Second tab</strong>',<
    onlyContent: true
  }, {
    id: 'tab3',
    name: '<strong>Third tab</strong>'
  }]
});
```

FORM_REFRESH

```
function form_refresh();
```

Refreshes the entire iframe that contains the form.

PARAMETERS

None.

RETURN VALUE

None.

EXAMPLE

```
form_refresh();
```

form_escapeHtml

```
function form_escapeHtml(text);
```

Escapes a block of text according to HTML escaping rules.

Parameters

text

The text to escape.

Return value

The escaped text.

Example

```
window.alert(form_escapeHtml("\"this is a text");
```

will display

```
&quot;this is a &lt;bracketed&gt; text
```

form_refreshViewer

```
function form_refreshViewer();
```

Refreshes the viewer that may be displayed next to the form. This is useful when form actions trigger document changes and the document must be re-displayed to reflect those changes.

This API function has effect only in the inbox component, where a form may be displayed side by side with a viewer panel.

Parameters

None.

Return value

None.

Example

```
form_refreshViewer();
```

form_cleanupHtml

```
function form_cleanupHtml(text);
```

Removes <p> and
 tags from the input text.

Also replaces any
 tags by a hard line break (\r).

Parameters

text

The text to cleanup.

Return value

The cleaned-up text.

Example

```
window.alert(form_escapeHtml( "<p>this is a paragraph<br>and a line break</p>");
```

will display

```
this is a paragraph
and a line break
```

FORM_LOG

```
function form_log(text);
```

Outputs a log message to the browser console.

PARAMETERS

text

The message to write to the console.

RETURN VALUE

None.

EXAMPLE

```
form_log("message");
```

FORM_INFO

```
function form_info(text);
```

Outputs an info message to the browser console.

Parameters

text

The message to write to the console.

Return value

None.

Example

```
form_info("message");
```

FORM_WARN

```
function form_warn(text);
```

Outputs a warning message to the browser console.

PARAMETERS

text

The message to write to the console.

RETURN VALUE

None.

EXAMPLE

```
form_warn("message");
```

form_error

```
function form_error(text);
```

Outputs an error message to the browser console.

Parameters

text

The message to write to the console.

Return value

None.

Example

```
form_error("message");
```

FORM_WAITSTART

```
function form_waitstart(inForm);
```

Displays a 'wait' rotating glyph to indicate the application is busy.

PARAMETERS

inForm

Indicates whether the wait rotating glyph should block the form or the entire view.

RETURN VALUE

None.

EXAMPLE

```
form_waitstart(true);
```

FORM_WAITSTOP

```
function form_waitstop(inForm, immediately);
```

Closes the ‘wait’ rotating glyph used to indicate the application is busy.

Parameters

inForm

Indicates whether the wait rotating glyph is from the form or the entire view.

immediately

(optional) Indicates whether the unblock should be performed immediately or with the system predefined timeout.

Return value

None.

Example

```
form_waitstop(true);
```

form_formatDate

```
function form_formatDate(date);
```

Formats a date with the “dd/MM/yyyy” pattern.

Parameters

date

the date object to format

Return value

The formatted date.

Example

```
var date = new Date();
var formattedDate = form_formatDate(date); // something like "05/05/2024"
```

form_parseDate

```
function form_parseDate(date);
```

Parses a date formatted with the “dd/MM/yyyy” pattern and returns a valid date object.

Parameters

date

A string representation of a date in the “dd/MM/yyyy” pattern

Return value

A date object.

Example

```
var formattedDate = "05/05/2024";
var date = form_parseDate(formattedDate);
```

FORM_FORMATUSERNAME

```
function form_formatUserName(userName, realName)
```

Formats the given user name and real name based on the siatel.config.user.name.display.format server option.

PARAMETERS

userName

The user name

realName

The real name of the user

RETURN VALUE

The formatted user name.

EXAMPLE

```
var name = form_formatUserName("jeanw", "Jean Winters"); // can return something
like "jeanw (Jean Winters)"
```

Depending on the siatel.config.user.name.display.format server option the result will differ. In the example above the option is set to {{userName}} ({{realName}}).

form_hasDocuments

```
function form_hasDcuments(form, callback)
```

Checks to see if the process has documents or not that have the given form.

This API function has effect only in the inbox component.

Parameters

form

The name or the id of the form

callback

A callback function that receives the result as a parameter (optional)

Return value

If the callback option is set, then the operation will be asynchronous and the result will be returned through it, otherwise the operation is synchronous and the result will be returned by the function its self.

The result will be `true` or `false`.

Example

```
var hasDocs = form_hasDocuments("Default"); // sync call
form_hasDocuments("Default", function(hasDocs) {
    ...
}); // async call
```

form_getDocuments

```
function form_getDocuments(form, callback)
```

Returns the ids of the documents in a process that have the given form.

Parameters

form

The name or the id of the form

callback

A callback function that receives the result as a parameter (optional)

Return value

If the callback option is set, then the operation will be asynchronous and the result will be returned through it, otherwise the operation is synchronous and the result will be returned by the function its self.

The result will be an array of ids.

Example

```
var ids = form_getDocuments("Default"); // sync call
form_getDocuments("Default", function(ids) {
    ...
}); // async call
```

form_filterList

```
function form_filterList(id, arguments)
```

Filters the options of a list element.

Parameters

id

The id of the list element

arguments

There are 3 possibilities:

- `regexp` - a regular expression either as object or string
- `callback` - a function callback that receives the current option and level as parameters and must return true if the option should be kept or false if it should be removed
- `searchOptions, contains` - an array of options and a boolean indicating whether to keep only the options contained in the searchOptions array (if `true`) or if to remove any of the options that are contained in the array (if `false`)

Return value

None

Example

```
form_filterList(id, /^test/); // keeps only options who's value starts with test and
removes all others
form_filterList(id, function (option, level) {
    return option.startsWith("test");
});
form_filterList(id, ["test element 1", "test element 2"], true); // keeps only the
options who's values appear in the search array
```

FORM_SETDOCTEMPLATEOPTIONS

```
function form_setDocTemplateOptions(options)
```

Specifies the options used by the document template selection dialog.

PARAMETERS

options

an object that can contain the following options:

default

displays default templates; possible values: `0/1, true/false`

custom

displays custom templates(defined in administration console); possible values: `0/1, true/false`

plugin

displays plugin templates; possible values: `0/1, true/false`

tagNames

filter plugin templates by tag name; value: array of tag names

RETURN VALUE

None.

EXAMPLE

```
form_setDocTemplateOptions({ default: true, custom: 0, plugin: 1, tagNames: ['tag1',
'tag2'] });
```

JavaScript functions for customisizing the user interface

flashElement

```
function flashElement(elementID, color, count)
```

Makes the specified element flash. Used to draw the attention of a user.

Parameters

elementID

The ID of the attribute to be flashed.

color

The color used for flashing.

count

Number of flashes.

Return value

None.

Example

form_displayButtonBar

```
function form_displayButtonBar(bDisplay)
```

Shows/hides the toolbar. Useful if the forms contain custom buttons.

Parameters

bDisplay

Visibility indicator (”true” or “false”).

Return value

None.

Example

FORM_DISPLAYDOCUMENTLISTTOGGLE

This function has been deprecated; it exists only for compatibility reasons and has no effect. The user interface of GARGANTUA 8 no longer displays the document list at the same time as the task form.

```
function form_displayDocumentListToggle()
```

Shows/hides the container of the process documents list.

PARAMETERS

None

RETURN VALUE

None.

EXAMPLE

form_displayDocumentList

This function has been deprecated; it exists only for compatibility reasons and has no effect. The user interface of Gargantua 8 no longer displays the document list at the same time as the task form.

```
function form_displayDocumentList(bDisplay)
```

Shows/hides the process documents list.

Parameters

bDisplay

Visibility indicator (”true” or “false”).

Return value

None.

Example

form_displaySelectorArea

```
function form_displaySelectorArea(bDisplay)
```

Shows/hides the form selection bar. Useful if changing the form is not required.

Parameters

bDisplay

Visibility indicator (”true” or “false”).

Return value

None.

Example

form_displayDefaultButtonBar

```
function form_displayDefaultButtonBar()
```

Displays the default buttons below the form; the default buttons depend on the context; for instance, when the form is displayed for a folder these buttons are different than the buttons displayed in the inbox.

Parameters

None

Return value

None.

Example

FORM_SETCUSTOMBUTTONENABLED

```
function form_setCustomButtonEnabled(id, enabled)
```

Enables or disables one of the customs buttons previously created by calling `form_displayCustomButtonBar`.

PARAMETERS

id

The ID of the button to show or hide

enabled

The new enabled/disabled status of the button

RETURN VALUE

None.

EXAMPLE

form_displayCustomButtonBar

```
function form_displayCustomButtonBar(data)
```

Shows a customized button bar below the form.

Parameters

data

A JSON object describing the button structure to create.

Return value

None.

Example

form_setCustomButtonVisible

```
function form_setCustomButtonVisible(id, visible)
```

Shows or hides one of the customs buttons previously created by calling `form_displayCustomButtonBar`.

Parameters

id

The ID of the button to show or hide

visible

The new visible/hidden status of the button

Return value

None.

Example

form_setCustomButtonHidden

This function has been deprecated; it exists only for compatibility reasons and has no effect.

```
function form_setCustomButtonHidden(id, visible)
```

Hides one of the customs buttons previously created by calling `form_displayCustomButtonBar`.

Parameters

id

The ID of the button to show or hide

visible

The new visible/hidden status of the button

Return value

None.

Example

form_refreshDocumentList

This function has been deprecated; it exists only for compatibility reasons and has no effect. The user interface of GARGANTUA 8 no longer displays the document list at the same time as the task form.

```
function form_refreshDocumentList()
```

Refreshes the document list.

Parameters

None

Return value

None.

Example

form_displayDocumentListOnReturn

This function has been deprecated; it exists only for compatibility reasons and has no effect. The user interface of Gargantua 8 no longer displays the document list at the same time as the task form.

```
function form_displayDocumentListOnReturn()
```

Parameters

None

Return value

None.

Example

FORM_PROCESSAREAVERTICALMAXIMIZE

This function has been deprecated; it exists only for compatibility reasons and has no effect. The user interface of GARGANTUA 8 no longer displays the document list at the same time as the task form.

```
function form_processAreaVerticalMaximize(bMaximize)
```

Expands/shrinks vertically the display area of the process form.

PARAMETERS

bMaximize

Expand/Shrink the form area.

RETURN VALUE

None.

EXAMPLE

form_processAreaResizeToggle

This function has been deprecated; it exists only for compatibility reasons and has no effect. The user interface of GARGANTUA 8 no longer displays the document list at the same time as the task form.

```
function form_processAreaResizeToggle()
```

Extends/shrinks the display area of the form process.

Parameters

None.

Return value

None.

Example

FORM_PROCESSAREARESIZE

This function has been deprecated; it exists only for compatibility reasons and has no effect. The user interface of GARGANTUA 8 no longer displays the document list at the same time as the task form.

```
function form_processAreaResize(bResize)
```

Resizes the display area of the form process.

PARAMETERS

bResize

Expand/Shrink the form area.

RETURN VALUE

None.

EXAMPLE

form_displayAlert

```
function form_displayAlert(data)
```

Displays an alert for the user to notice.

Parameters

data

A JSON object describing the alert to display.

Return value

The jQuery alert object.

Example

```
form_displayAlert({
    type : 'error',
    message : 'Error alert without a title ...'
});
```

```
form_displayAlert({
    type : 'info',
    title : 'Not neccesary',
    message : 'Info message here ...'
});
```

```
var $alert = form_displayAlert({
    type : 'success',
    keys : {
        message : 'common.edit.success'
    }
});

_.delay(function() {
    form_closeAlert($alert);
}, 500);
```

FORM_CLOSEALERT

```
function form_closeAlert($alert)
```

Closes an alert previously displayed by a call to `form_displayAlert`.

PARAMETERS

$alert

The jQuery object that represents the alert to close; created by a call to `form_displayAlert`.

RETURN VALUE

None.

EXAMPLE

```
var $alert = form_displayAlert({
    type : 'success',
    keys : {
          message : 'common.edit.success'
    }
});

_.delay(function() {
    form_closeAlert($alert);
}, 500);
```

form_displayDialog

```
function form_displayDialog(data)
```

Displays a dialog box for the user to enter data or perform actions.

Parameters

data

A JSON object describing the dialog to display.

Return value

The jQuery dialog object.

Example

```
form_displayDialog({
    type : 'error',
    message : 'Dialog error message here ...',
});
```

```
form_displayDialog({
    type : 'info',
    title : 'Dialog title', // defaults to Gargantua if not specified
    message : 'Dialog info message here ...'
});
```

```
// Button codes :
// var Button = {
//   ESCAPE : 0,
//   OK : 1,
//   CANCEL : 2,
//   YES : 3,
//   NO : 4,
//   CLOSE : 5
// }

// Button combination codes :
// var Buttons = {
//   OK : 1,
//   OK_CANCEL : 2,
//   YES_NO : 3,
//   YES_NO_CANCEL : 4
```

```
//   CANCEL : 5
//   CLOSE : 6
// }

form_displayDialog({
     type : 'question',
     title : 'Survey',
     message : 'What do you want to do ? ...',
     buttons : Buttons.YES_NO_CANCEL,
     callback : function(e, key, btn) {
          console.log('pressed : ' + btn);

          switch (btn)
          {
               case Button.YES:
                    // do something
                    break;
               case Button.NO:
                    // do something
                    break;
               default:
                    break;
          }

     }
});
```

```
var $dialog = form_displayDialog({
     type : 'success',
     keys : {
          message : 'common.edit.success'
     }
});


_.delay(function() {
     form_closeDialog($dialog);
}, 500);
```

```
form_displayDialog({
     type : 'warning',
     keys : {
          title : 'common.edit.title',
          message : 'common.edit.denied'
     }
});
```

form_displayCustomDialog

```
function form_displayCustomDialog(data)
```

Displays a custom dialog box for the user to enter data or take actions.

Parameters

data

A JSON object describing the dialog to display.

Return value

The jQuery dialog object.

Example

```
form_displayCustomDialog({
    title : 'Sélection d\'une fiche de traitement',
     body : function() {
          var htmlTemplate = '<table class="table table-hover">'
               + '<thead class="thead-primary font-weight-bold">'
               + '<tr><th class="w-5"> </th><th
class="w-10">Référence</th><th>Noms et descriptions des fiches</th></tr>'
               + '</thead>'
               + '<tbody>{{#each objects}}'
               + '<tr><td class="text-center"><input type="checkbox"
/></td><td>{{reference}}</td><td>'
               + '<div>{{name}}</div>'
               + '<div class="text-muted">{{description}}</div>'
               + '</td></tr>'
               + '{{/each}}</tbody>'
               + '</table>';

          var template = Handlebars.compile(htmlTemplate);
          var objects = [
               {
                    reference : 'TRT-0001',
                    name : 'Gestion des fiches du personnel',
                    description : 'Gestions des informations sur les fiches'
               },
               {
                    reference : 'TRT-0209',
                    name : 'Traitement pour les souris',
                    description : 'Description'
               },
               {
```

```
                    reference : 'TRT-0213',
                    name : 'Traitement des ânes',
                    description : 'A'
               }
          ];
          return template({
               objects : objects
          });
     },
     buttons : {
          validate : {
               label : 'Valider',
               className : 'btn-primary',
               callback : onValidate
          },
          close : {
               label : 'Fermer'
          }
     }
});
```

```
form_displayCustomDialog({
     id : 'attach-dlg',
     title : 'Ajouter une annexe',
     body : '<form role="form">'
          + '<div class="form-group">'
          + '<label class="form-control-label" for="attach-dlg-
file">Fichier</label>'
          + '<input class="form-control" id="attach-dlg-file" name="attach-dlg-
file" type="file" required="required" data-controltype="file">'
          + '</div>'
          + '<div class="form-group">'
          + '<label class="form-control-label" for="attach-dlg-
name">Libellé</label>'
          + '<input class="form-control" id="attach-dlg-name" name="attach-dlg-
name" type="text">'
          + '</div>'
          + '<div class="form-group form-check">'
          + '<input class="form-check-input" id="attach-dlg-printable"
name="attach-dlg-printable" type="checkbox">'
          + '<label class="form-check-label" for="attach-dlg-
printable">Imprimable</label>'
          + '</div>'
          + '</form>',

     buttons : {
          save : {
               label : 'Sauvegarder',
               className : 'btn-primary',
               // callback : onSave
          },
          cancel : {
               label : 'Annuler
          }
     },
```

```
     callback : function(e, key, type) {
          switch (key)
          {
               case 'save':
                    onSave.call(this);
                    break;
               default:
                    break;
          }
     },
     controls : true
});
```

```
form_displayCustomDialog({
     title : 'Sélection d\'une fiche de traitement',
     body : {
          template : '<table class="table table-hover">'
               + '<thead class="thead-primary font-weight-bold">'
               + '<tr><th class="w-5"> </th><th
class="w-10">Référence</th><th>Noms et descriptions des fiches</th></tr>'
               + '</thead>'
               + '<tbody>{{#each objects}}'
               + '<tr><td class="text-center"><input type="checkbox"
/></td><td>{{reference}}</td><td>'
               + '<div>{{name}}</div>'
               + '<div class="text-muted">{{description}}</div>'
               + '</td></tr>'
               + '{{/each}}</tbody>'
               + '</table>',
          source : function() {
               var objects = [ {
                         reference : 'TRT-0001',
                         name : 'Gestion des fiches du personnel',
                         description : 'Gestions des informations sur les fiches'
                    },  {
                         reference : 'TRT-0209',
                         name : 'Traitement pour les souris',
                         description : 'Description'
                    },  {
                         reference : 'TRT-0213',
                         name : 'Traitement des ânes',
                         description : 'A'
                    } ];

                    return {
                         objects : objects
                    };
               }
          },
          buttons : {
               validate : {
                    label : 'Valider',
                    className : 'btn-primary',
                    callback : onValidate
```

```
            },
            close : {
                label : 'Fermer'
            }
    }
});
```

```
form_displayCustomDialog({
      url : '/app/cmds/document/replace/dialog',
      callback : function(e, key, btn) {
            window.console.log(btn);
      }
});
```

```
// Button codes :
// var Button = {
//   ESCAPE : 0,
//   OK : 1,
//   CANCEL : 2,
//   YES : 3,
//   NO : 4,
//   CLOSE : 5
// }

// Button combination codes :
// var Buttons = {
//   OK : 1,
//   OK_CANCEL : 2,
//   YES_NO : 3,
//   YES_NO_CANCEL : 4
//   CANCEL : 5
//   CLOSE : 6
// }

form_displayCustomDialog({
      title : 'Custom dialog',
      body : function() {
            return '<p>Test body</p>';
      },
      // title : function() {
      //   return 'Custom dialog';
      //},

      // body : '<p>Test body</p>',
      buttons : Buttons.OK,
      callback : function(e, key, btn) {
            window.console.log(btn);
      }
});
```

form_closeDialog

```
function form_closeDialog($dialog)
```

Closes a dialog box previously displayed by a call to `form_displayDialog` or `form_displayCustomDialog`.

Parameters

$dialog

The jQuery object that represents the dialog to close; created by a call to `form_displayDialog` or `form_displayCustomDialog`.

Return value

None.

Example

```
var $dialog = form_displayDialog({
     type : 'success',
     keys : {
          message : 'common.edit.success'
     }
});

_.delay(function() {
     form_closeDialog($dialog);
}, 500);
```

FORM_GETDIALOGCONTROL

PARAMETERS

...

...

RETURN VALUE

...

EXAMPLE

...

FORM_GETDIALOGELEMENT

PARAMETERS

...

...

RETURN VALUE

...

EXAMPLE

...

BUTTON

BUTTONS

form_loadScript

Parameters

...

...

Return value

...

Example

...

FORM_LOADSCRIPTS

PARAMETERS

...

...

RETURN VALUE

...

EXAMPLE

...

form_loadStylesheet

Parameters

...

...

Return value

...

Example

...

FORM_LOADSTYLESHEETS

PARAMETERS

...

...

RETURN VALUE

...

EXAMPLE

...

JavaScript functions for triggering actions

resetElement

```
function resetElement(id)
```

Resets the value of the given attribute input control.

Parameters

id

The ID of the attribute whose input control must be reset

Return value

None.

Example

runServerScript

```
function runServerScript(scriptName, parameters, callback, bAsync)
```

Starts the script execution at the server level. The script that is called must begin by the “execute” function:

```
function execute(params) {
    // processing
}
```

Paramètres et valeur retour :

scriptName

The name of the script to execute.

parameters

Parameters for the script. They must include the “databaseId” parameter, that indicates the database in which the script to run is located.

For instance: “databaseId=DATA000800000000000000000000000000000000”.

callback

Function that must be executed upon receiving the response from the server.

bAsync

A boolean that indicates if the request must be made in synchronous or asynchronous mode. The asynchronous mode is preferred, as it allows for a more responsive user interface.

Return value

Server response in the indicated format of the invoked function. In case of an async call the response will be received and processed in the callback function.

This API function has been replaced by the form_runScript function; please use the new function whenever possible.

runServerScript2

```
function runServerScript2(scriptName, functionName, parameters, callback, bAsync)
```

Starts the script execution at the server level.

Paramètres et valeur retour :

scriptName

The name of the script to execute.

functionName

The name of the function to execute.

parameters

Parameters for the script. They must include the “databaseId” parameter, that indicates the database in which the script to run is located.

For instance: “databaseId=DATA0008000000000000000000000000000000000”.

callback

Function that must be executed upon receiving the response from the server.

bAsync

A boolean that indicates if the request must be made in synchronous or asynchronous mode. The asynchronous mode is preferred, as it allows for a more responsive user interface.

Return value

Server response in the indicated format of the invoked function. In case of an async call the response will be received and processed in the callback function.

This API function has been replaced by the form_runScript function; please use the new function whenever possible.

FORM_RUNSCRIPT

```
function form_runScript(options)
```

Starts the script execution at the server level.

Parameters and return value:

options

Key-value object that specifies the following options:

- `script` ou `scriptName` - the name of the script to be executed
- `function` ou `functionName` - the name of the function to be executed
- `acync` - boolean indicating if the request should be synchronous (false) or asynchronous (true)
- `data` - object with aditional data to be sent
- `callback` - function to be called when the result is available.

Return value

None.

Example :

```
form_runScript({
    script: 'server.queries',
    'function': 'getValues',
    data: {
        docId: ids[0]
    },
    callback: function (result) {
        try {
           var json = JSON.parse(result);
           form_info(json);
           ....
        }
        catch (e) {
            form_error(e);
        }
    }
});
```

FORM_APIREQUEST

```
function form_apiRequest(options)
```

Used to make calls to outside sources (and avoid cross-origin request problems). The call will go through the GARGANTUA server, who will actually make the request on your behalf.

Parameters and return value:

options

Key-value object that contains the following options:

- `url` - the URL to call
- `method` - HTTP method (defaults to GET)
- `headers` - HTTP headers (as a key-value object)
- `params` - HTTP parameters (as a key-value object)
- `body` - HTTP body (as string or object)
- `contentType` - the content type
- `async` - boolean indicating if the request should be synchronous (false) or asynchronous (true)
- `callback` - function to be called when the result is available

Return value

None.

Example :

```
form_apiRequest({
    url :  'https://thequoteshub.com/api/random-quote',
    method : 'GET',
    callback : function(result) {
        form_info(JSON.parse(result));
    }
});
```

form_scanDocuments

This function has been deprecated; it exists only for compatibility reasons. Please replace it with "form_addDocuments". Currently, this function call defaults to "form_addDocuments"

```
function form_scanDocuments(direct)
```

Launches the scanning operation if the execution context is properly initialized.

Parameters

direct

Indicator for direct scanning. See `form_addDocuments` for more information.

Return value

None.

Example

FORM_SCANDOCUMENTSEXT

This function has been deprecated; it exists only for compatibility reasons. Please replace it with "form_addDocuments". Currently, this function call defaults to "form_addDocuments"

```
function form_scanDocumentsExt(direct, label)
```

Launches the scanning operation if the execution context is properly initialized.

PARAMETERS

direct

Indicator for direct scanning. See `form_addDocuments` for more information.

label

The label to use for the newly scanned documents.

RETURN VALUE

None.

EXAMPLE

form_scanDocumentsExt2

This function has been deprecated; it exists only for compatibility reasons. Please replace it with "form_addDocuments". Currently, this function call defaults to "form_addDocuments"

```
function form_scanDocumentsExt2(direct, form)
```

Launches the scanning operation if the execution context is properly initialized.

Parameters

direct

Indicator for direct scanning. See `form_addDocuments` for more information.

form

The Form to use for the newly scanned documents.

Return value

None.

Example

form_scanDocumentsExt3

This function has been deprecated; it exists only for compatibility reasons. Please replace it with "form_addDocuments". Currently, this function call defaults to "form_addDocuments"

```
function form_scanDocumentsExt3(direct, form, label)
```

Launches the scanning operation if the execution context is properly initialized.

Parameters

direct

Indicator for direct scanning. See `form_addDocuments` for more information.

form

The Form to use for the newly scanned documents.

label

The label to use for the newly scanned documents.

Return value

None.

Example

form_createDocument

```
function form_createDocument(template, label, form, fulltext, markers)
```

Creates a new document based on a template and text replacements,

Parameters

template

The template ID

label

The label to use for the new document.

form

The form to use for the new document.

fulltext

Boolean to indicate whether the document has to be indexed for fulltext search

markers

A map of (<placeholder>, <value>) elements to replace in the new document

Return value

The new document ID.

Example

FORM_ADDDOCUMENTS

```
function form_addDocuments(options)
```

Launches the document import operation if the execution context is properly initialized.

PARAMETERS

options

An object describing the addDocument operation options.

direct

Indicator for direct scanning; otherwise a form selector is shown first. Supported values are: **DIRECT_NONE** (0), **DIRECT_SCAN** (1), **DIRECT_SCANWITHPROFILE** (2), **DIRECT_IMPORT** (3), **DIRECT_EMAIL** (4) or **DIRECT_IMPORT_ANY** (5). You can use either the constant or its value.

form

The name of the form to use for the new documents; takes precedence over the formId, if both are specified

formId

The id of the form to use for the new documents.

label

The label for the new documents; takes precedence over the name, if both are specified

name

The name for the new documents; used as a fallback if label is not specified

RETURN VALUE

None.

EXAMPLE

form_addDocumentsDirect

This function has been deprecated; it exists only for compatibility reasons. Please replace it with "form_addDocuments". Currently, this function call defaults to "form_addDocuments"

```
function form_addDocumentsDirect(direct)
```

Launches the add document operation if the execution context is properly initialized.

Parameters

direct

Indicator for direct add (no form selector). See `form_addDocuments` for more information.

Return value

None.

Example

form_certifyDocument

```
function form_certifyDocument(params)
```

Shows the document certification user interface elements for the given document.

Parameters

params

An object describing the operation options.

id

The document id to display the certification UI for.

Return value

None.

Example

form_editDocument

```
function form_editDocument(id, create, callback)
```

Triggers the 'edit native' operation for a given document

Parameters

id

The document ID to edit

create

Indicates whether this is a new document or an existing document; for new documents, when the editing operation ends, there will be no dialog box asking for versioning information

callback

The callback used when the operation opens the native application.

Return value

None.

Example

FORM_VIEWDOCUMENT

```
function form_viewDocument(id, options)
```

Displays the view document user interface elements according to the global layout and settings; has the same effect as the user clicking on the document label or double-clicking on the document line or thumbnail.

PARAMETERS

id

The document ID to display

options

A JSON object specifying the operation options.

The options parameter is a JSON object that needs to follow the following structure; one or more members may be present.

```
{
    viewer : {
        toolbar : {
            close : false,
            zapper : false,
            pinZapper : false,
            editNative : false,
            closeOnEditNative : false,
            switchView : false
        },
        label : true,
        panel : {
            display : 'none',
            fragment : true,
            notes : true,
            ocr : true,
            links : false,
            actions : false,
            sysinfo : true
```

```
        }
    },
    getDisplayData : $.noop,
    onZap : $.noop,
    onClose : $.noop,
    onActionFinished : $.noop
}
```

viewer/toolbar

individual commands for the viewer panel : close button, zapper, zapper for pins, edit native, close viewer on editnative and switch view.

label

boolean to indicate if label should be displayed or not

panel

what the footer panel shoulr display : default display, fragment, notes, ocr, links, actions, system information

getDisplayData, onZap, onClose, onActionFinished

various callbacks for the viewer panel

Return value

None.

Example

form_signDocument

```
function form_signDocument(params)
```

Shows the document signing user interface elements for the given document.

Parameters

params

An object describing the operation options.

id

The document id to display the signing UI for.

Return value

None.

Example

form_viewDocuments

```
function form_viewDocuments()
```

Displays the view documents user interface elements according to the global layout and settings.

This API function call is used exclusively in inbox context.

Parameters

None.

Return value

None.

Example

...

form_downloadDocuments

```
function form_downloadDocuments(ids)
```

Function that downloads the documents identified by the ids array.

Parameters

ids

Array of document ids.

Return value

None

FORM_EDITOBJECT

```
function form_editObject(id, options)
```

Function that opens the view/edit attributes dialog for the specified object.

PARAMETERS

id

The ID of the object to edit.

options

Edit options.

RETURN VALUE

None.

EXAMPLE

...

FORM_SAVE

```
function form_save()
```

Saves the form content of the displayed process or task.

Parameters and return value:

No parameter

Return value

None.

FORM_SAVEIMMEDIATE

```
function form_saveImmediate()
```

Saves the form content of the displayed process or task immediately and silently (no success alert will be shown).

Parameters and return value:

No parameter

Return value

None.

FORM_SAVEDRAFT

```
function form_saveDraft()
```

Saves the form of the displayed draft process.

Parameters and return value:

No parameter

Return value

None.

This API function has been replaced by the form_save function; please use the new function whenever possible.

form_saveDraftImmediate

```
function form_saveDraftImmediate()
```

Saves all the form content (all attribute values) immediately and silently.

Parameters and return value:

No parameter

Return value

None.

This API function has been replaced by the form_saveImmediate function; please use the new function whenever possible.

FORM_SAVETASKIMMEDIATE

```
function form_saveTaskImmediate()
```

Saves all the form content (all attribute values) immediately and silently.

Parameters and return value:

No parameter

Return value

None.

This API function has been replaced by the form_saveImmediate function; please use the new function whenever possible.

form_saveTask

```
function form_saveTask()
```

Saves the form of the displayed task.

Parameters and return value:

No parameter

Return value

None.

This API function has been replaced by the form_save function; please use the new function whenever possible.

FORM_CANCELDRAFT

```
function form_cancelDraft()
```

Cancels the draft and thus the process start operation.

Parameters and return value:

No parameter

Return value

None.

FORM_CANCEL

```
function form_cancel()
```

Cancels the process draft (and by consequence the process start) or the current task if the execution context is properly initialized, depending on the case.

Parameters and return value:

No parameter

Return value

None.

FORM_FINISH

```
function form_finish()
```

Starts a new process if the current process is a draft or sends the current task (validate task - finish task) if the execution context is properly initialized, depending on the case.

Parameters and return value:

No parameter

Return value

None.

FORM_STARTPROCESS

```
function form_startProcess()
```

Starts a new process if the current process is a draft and the execution context is properly initialized.

Parameters and return value:

No parameter

Return value

None.

FORM_CANCELTASK

```
function form_cancelTask()
```

Sends the task (canceled task) if the execution context is properly initialized.

Parameters and return value:

No parameter

Return value

None.

FORM_FINISHTASK

```
function form_finishTask()
```

Sends the task (validate task - finish task) if the execution context is properly initialized.

Parameters and return value:

No parameter

Return value

None.

FORM_EXECUTESEARCH

```
function form_executeSearch()
```

Starts the execution of a search if the execution context is properly initialized.

This API function call is used exclusively for forms available for the search context.

PARAMETERS

None.

RETURN VALUE

None.

EXAMPLE

FORM_ESCALATETASK

```
function form_escalateTask()
```

Escalates the task according to the rules set by the process definition.

doValidateForm

```
function doValidateForm()
```

Performs the form fields validation test. Disables the send button and highlights in red the background of all the fields that contain errors.

Parameters

None.

Return value

None.

Example

handler_onSwitchPage

```
function form_switchPage(page)
```

Switch to the given form page.

Parameters

page

the name of the page to display.

Return value

None.

Example

form_switchForm

```
function form_switchForm(form, page)
```

Switch to the given form and form page.

Parameters

form

the name of the form that contains the page to display.

page

the name of the page to display.

Return value

None.

Example

FORM_GOTOTASKS

```
function form_gotoTasks()
```

Returns to the task list from the form/document display.

Parameters

None

Return value

None

Exceptions

n/a

Example

```
form_gotoTasks();
```

form_invokePlugin

```
function form_invokePlugin(pluginId, command, params, format)
```

Executes the given command from a server plugin, using the provided parameters and output format.

Parameters

pluginId

the ID of the plugin to invoke.

command

the name of the plugin command to execute.

params

A JSON objects containing command-specific parameters.

format

A string specifying the expected return format; can be either xml or text.

Return value

The server response in the requested format, if the plugin can handle it; otherwise, it is a text response.

Example

Advanced JavaScript functions

FORM_NEWCONTROL

```
function form_newControl(id, type, options)
```

Creates a new input control based on the given parameters.

PARAMETERS

RETURN VALUE

EXAMPLE

form_destroyControl

```
function form_destroyControl(id)
```

Destroys a previously manually created control

Parameters

id

The form element control ID

Return value

None

Example

FORM_GETCONTROL

```
function form_getControl(id)
```

Retrieves the control for the given input ID

PARAMETERS

id

The form element control ID

RETURN VALUE

The new control object.

EXAMPLE

form_getElement

```
function form_getElement()
```

Retrieves the jQuery object associated with the given element ID

Parameters

id

The page element ID

Return value

The jQuery object associated with the given element ID

Example

getElementKeyAndValue

```
function getElementKeyAndValue()
```

Retrieves the selected (key, value) pair from the autocomplete control as a JSON object.

Parameters

id

The form element control ID

Return value

A JSON object that represents the (key, value) pair for the given selector.

Example

setElementKeyAndValue

```
function setElementKeyAndValue(id, key, value)
```

Sets the key and value for the given autocomplete control (a list)

Parameters

id

The form element control ID

key

The key to be set

value

The value to be set

Return value

None

Example

FORM_SAVEGLOBALGRIDS

```
function form_saveGlobalGrids()
```

Saves the values of all grids in the form.

This API function has been deprecated in GARGANTUA 8. You may safely remove it when porting existing forms from older GARGANTUA versions.

PARAMETERS

None

RETURN VALUE

None

EXAMPLE

form_validateLatency

```
function form_validateLatency(timeout)
```

Modifies the form validation latency; this affects how the validation is performed on the form values.

Parameters

timeout

The validation timeout to be used; effective from the function call

Return value

None

Example

FORM_GETVALIDATELATENCY

```
function form_getValidateLatency()
```

Retrieves the current form validation latency value.

PARAMETERS

RETURN VALUE

newMode

EXAMPLE

form_validateLatencyOffset

```
function form_validateLatencyOffset(offset)
```

Modifies the form validation latency offset; this affects how the validation is performed on the form values.

Parameters

offset

The validation timeout offset to be used; effective from the function call

Return value

None

Example

form_validateMode

```
function form_validateMode(newMode)
```

Changes the validation mode for the form.

Parameters

newMode

The new validation mode for the form; it can be either `VALIDATE_ONSUBMIT` or `VALIDATE_CONTINUOUS`

Return value

None

Example

form_getValidateMode

```
function form_getValidateMode()
```

Retrieves the current validation mode for the form.

Parameters

None

Return value

The current validation mode. It can be either VALIDATE_ONSUBMIT or VALIDATE_CONTINUOUS

Example

FORM_GETVALIDATIONMESSAGE

```
function form_getValidationMessage(failedIds, failedReasons, failedCodes [, options])
```

Returns a standard validation message with the form errors.

Parameters

failedIds

An array with the failed attribute IDs.

failedReasons

An array with the failure reasons.

failedCodes

An array with the failure codes.

options

An object containing the following possible options:

- `multiline` parameter indicating if the different errors should be rendered on separate lines or not. By default this is `false`.

Return value

None.

Example

```
function fun_form_post_validatereponse(failedIds, failedReasons, failedCodes)
{
    var message = form_getValidationMessage(failedIds, failedReasons, failedCodes, {
multiline: true });
    if (message)
        form_displayAlert({ type : 'error', message: message });
}
```

FORM_SETSUBMITONENTER

```
function form_setSubmitOnEnter(bEnable)
```

Enables or disables the use of the **Enter** key for submitting the form.

By default, the "Enter" key is enabled for submitting the form.

Parameters

bEnable

Boolean to enable or disable the use of the **Enter** key for submitting the form..

Return value

None.

Example

```
form_setSubmitOnEnter(false);
```

CONTROLS

GENERAL BEHAVIOR

getValue

```
getValue()
```

Gets the value of the control as a string.

Parameters

None.

Return value

The current value of the control.

Example

```
var control = form_getControl(attrId);
var value = control.getValue();
```

Equivalent to:

```
var value = getElementValue(attrId);
```

setValue

```
setValue(value)
```

Sets the value of the control as a string.

Parameters

value

The value to set.

Return value

None

Example

```
var value = "...";
var control = form_getControl(attrId);
control.setValue(value);
```

Equivalent to:

```
setElementValue(attrId, value);
```

enable

```
enable()
```

Enables the control.

Parameters

None

Return value

None

Example

```
var control = form_getControl(attrId);
control.enable();
```

Equivalent to:

```
setElementEnabled(attrId, true);
```

disable

```
disable()
```

Disables the control.

Parameters

None

Return value

None

Example

```
var control = form_getControl(attrId);
control.disable();
```

Equivalent to:

```
setElementEnabled(attrId, false);
```

isEmpty

```
isEmpty()
```

Checks if the control value is empty.

Parameters

None

Return value

`true` if the value is empty, `false` otherwise

Example

```
var control = form_getControl(attrId);
if (control.isEmpty())
{ ... }
```

reset

```
reset()
```

Resets the value of the control to the initial value.

Parameters

None

Return value

None

Example

```
var control = form_getControl();
control.reset();
```

isEnabled

```
isEnabled()
```

Checks if the control is enabled.

Parameters

None

Return value

`true` if the control is enabled, `false` otherwise

Example

```
var control = form_getControl(attrId);
if (control.isEnabled())
{ ... }
```

Equivalent to:

```
isElementEnabled(attrId)
```

hasError

```
hasError()
```

Checks if the control value is correct or not (if it respects the control's pattern).

Parameters

None

Return value

`true` if the value is correct, `false` otherwise

Example

```
var control = form_getControl(attrId);
if (control.hasError())
{ ... }
```

SIMPLE INPUT FIELDS

Simple string input

To create a string control:

```
var control = form_newControl(attrId, 'string');
```

Numeric input

To create a numeric control:

```
var control = form_newControl(attrId, 'number', options);
```

The options include:

type

The type of number `[integer|float]`

negative

Boolean indicating if the number can be negative or not

min

The minimum value (any value below it will be considered invalid)

max

The maximum value (any value above it will be considered invalid)

maxUnits

The maximum number of units

maxSubunits

The maximum number of subunits

getNumber

```
getNumber()
```

Gets the value of the control as a number.

Parameters

None

Return value

The value as a number.

Example

```
var control = form_getControl(attrId);
var number = control.getNumber();
```

setNumber

```
setNumber(number)
```

Sets the value of the control as a number.

Parameters

number

The value as a number.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setNumber(250);
```

Date input

To create a date control:

```
var control = form_newControl(attrId, 'date', options);
```

The options include:

mouseWheel

Boolean indicating if the mouse wheel should or not be used to alter the input's value

getDate

```
getDate()
```

Gets the value of the control as a date (without the hour, minutes, seconds and milliseconds).

Parameters

None

Return value

The value as a date.

Example

```
var control = form_getControl(attrId);
var date = control.getDate();
```

setDate

```
setDate(date)
```

Sets the value of the control as a date (without the hour, minutes, seconds and milliseconds).

Parameters

date

The value as a date.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setDate(new Date());
```

Time input

To create a time control:

```
var control = form_newControl(attrId, 'time', options);
```

The options include:

mouseWheel

Boolean indicating if the mouse wheel should or not be used to alter the input's value

getTime

```
getTime()
```

Gets the value of the control as an array with 2 elements (hours and minutes).

Parameters

None

Return value

The value as an array with 2 elements (hours and minutes).

Example

```
var control = form_getControl(attrId);
var timeArray = control.getTime();
```

setTime

```
setTime(time)
```

Sets the value of the control as an array with 2 elements (hours and minutes).

Parameters

time

The value as an array with 2 elements (hours and minutes).

Return value

None

Example

```
var control = form_getControl(attrId);
control.setTime([10, 25]);
```

Date & time input

To create a date-time control:

```
var control = form_newControl(attrId, 'datetime', options);
```

The options include:

mouseWheel

Boolean indicating if the mouse wheel should or not be used to alter the input's value

getTimestamp

```
getTimestamp()
```

Gets the value of the control as a date.

Parameters

None

Return value

The value as a date.

Example

```
var control = form_getControl(attrId);
var date = control.getTimestamp();
```

setTimestamp

```
setTimestamp(date)
```

Sets the value of the control as a date.

Parameters

date

The value as a date.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setTimestamp(new Date());
```

Time delay input

To create a time delay control:

```
var control = form_newControl(attrId, 'timedelay', options);
```

The options include:

mouseWheel

Boolean indicating if the mouse wheel should or not be used to alter the input's value

getDelay

```
getDelay()
```

Gets the value of the control as a number (in seconds).

Parameters

None

Return value

The value as a number.

Example

```
var control = form_getControl(attrId);
var delayInSeconds = control.getDelay();
```

setDelay

```
setDelay(delay)
```

Sets the value of the control as a number (in seconds).

Parameters

delay

The value as a date.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setDelay(3600); // 60 minutes
```

Email input

To create an email control:

```
var control = form_newControl(attrId, 'email', options);
```

The options include:

multiple

Boolean indicating if the value can contain multiple emails

source

The source URL

LISTS

Autocomplete

To create an autocomplete control:

```
var control = form_newControl(attrId, 'autocomplete', options);
```

The source of the autocomplete is an array of objects. Each object should have at least two properties: a unique value that identifies the object and a display value (for example an id and a name).

The options include:

multiple

Boolean indicating if the control allows multiple values

displayKey

The key identifying the value to display in the control element (for instance the name in the example above)

valueKey

The key that uniquely identifies the selected value (for instance the id in the example above)

freeInput

Boolean indicating if the autocomplete should allow the user to write values that do not exist in the source

source

The source of the autocomplete (can be a URL, an object or a function)

minLength

The minimum number of characters the user has to type before the autocomplete is triggered

prepareSuggestion

A callback function allowing you to customize the display of the autocomplete suggestions

Example

```
form_newControl(attrId, 'autocomplete', {
    displayKey: 'value',
    valueKey: 'key',
    source: {
        values : [
            { key : 'k1', value : 'New York' },
            { key : 'k2', value : 'Chicago' },
            { key : 'k3', value : 'Geneva' },
            { key : 'k4', value : 'London' },
            { key : 'k5', value : 'Helsinki' },
            { key : 'k6', value : 'Boston' },
            { key : 'k7', value : 'Paris (France)' },
```

```
            { key : 'k8', value : 'Paris (Texas)' },
            { key : 'k9', value : 'Memphis (Egypt)' },
            { key : 'k10', value : 'Memphis (Tennessee)' },
            { key : 'k11', value : 'Madrid' },
             { key : 'k12', value : 'Vienna' },
             { key : 'k13', value : 'Sydney (Australia)' },
             { key : 'k14', value : 'Sydney (Canada)' }
        ]
    }
});
```

The source should always look like the example above even if it's a function or an URL, the returned value must follow the same pattern.

getKey

```
getKey()
```

Gets the key of the control.

Parameters

None

Return value

The key of the control.

Example

```
var control = form_getControl(attrId);
var key = control.getKey();
var value = control.getValue();
```

For the example of cities above, if the user selected “Paris (France)”, the key will be `k7` and the value will be `Paris (France)`.

setKey

```
setKey(key)
```

Sets the key of the control.

Parameters

key

The key to set.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setKey('k7');
```

getKeyAndValue

```
getKeyAndValue()
```

Gets an object containing the key and value of the control (the property names will be those configured as displayKey and valueKey).

Parameters

None

Return value

An object containing the key and value of the control.

Example

```
var control = form_getControl(attrId);
var obj = control.getKeyAndValue();
```

For the example of cities above, if the user selected "Paris (France)", the object will be `{ 'key' : 'k7', 'value' : 'Paris (France)' }`.

Equivalent to:

```
getElementKeyAndValue(attrId);
```

setKeyAndValue

```
setKeyAndValue(object)
```

Sets the key and value of the control.

Parameters

object

The key and value to set.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setKeyAndValue({ 'key' : 'k7', 'value' : 'Paris (France)' });
```

Equivalent to:

```
setElementKeyAndValue(attrId, { 'key' : 'k7', 'value' : 'Paris (France)' });
```

Simple list

To create a simple list control:

```
var control = form_newControl(attrId, 'list', options);
```

The options include:

flags

`[m|s|p|c]` The flags indicate the type of the list. `m` stands for multiple selection list, `s` stands for simple list, `p` stands for full path list, and `c` stands for category list

source

The source of the list (can be a URL, an object or a function)

closed

Boolean indicating if the list should not allow the user to write values that do not exist in the source

addEmptyOption

Boolean indicating if an empty option should be added to the list

prepareOption

A callback function allowing you to customize the display of the list options

linkId

The id of the master list (if you need to create linked lists; for multi level lists - each list will display it's own level)

Example

```
form_newControl(attrId, 'list', {
    source: {
        values : [
            { key : 'c1', value : 'America', values : [
                { key : 'k1', value : 'New York' },
                { key : 'k2', value : 'Chicago' },
                { key : 'k3', value : 'Boston' }
             ]},
             { key : 'c2', value : 'Europe', values : [
                { key : 'k4', value : 'London' },
                { key : 'k5', value : 'Paris' },
                { key : 'k6', value : 'Vienna' }
             ]}
        ]
    }
});
```

The example above will build a list with 2 levels.

If you add the `flags: 'c'` option the list will transform into a category list in which only the leaf values will be selectable.
By default the flags option is considered `'s'`, in which case all values are selectable. In the case of `'m'` flag you can select multiple values, and in the case of `'p'` flag the selected value will contain the full path (for instance 'Europe/Paris').

getKey

```
getKey()
```

Gets the key of the control.

Parameters

None

Return value

The key of the control.

Example

```
var control = form_getControl(attrId);
var key = control.getKey();
var value = control.getValue();
```

For the example of cities above, if the user selected “Paris (France)”, the key will be `k7` and the value will be `Paris (France)`.

setKey

```
setKey(key)
```

Sets the key of the control.

Parameters

key

The key to set.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setKey('k7');
```

getKeyAndValue

```
getKeyAndValue()
```

Gets an object containing the key and value of the control (the property names will be those configured as displayKey and valueKey).

Parameters

None

Return value

An object containing the key and value of the control.

Example

```
var control = form_getControl(attrId);
var obj = control.getKeyAndValue();
```

For the example of cities above, if the user selected "Paris (France)", the object will be `{ 'key' : 'k7', 'value' : 'Paris (France)' }`.

Equivalent to:

```
getElementKeyAndValue(attrId);
```

setKeyAndValue

```
setKeyAndValue(object)
```

Sets the key and value of the control.

Parameters

object

The key and value to set.

Return value

None

Example

```
var control = form_getControl(attrId);
control.setKeyAndValue({ 'key' : 'k7', 'value' : 'Paris (France)' });
```

Equivalent to:

```
setElementKeyAndValue(attrId, { 'key' : 'k7', 'value' : 'Paris (France)' });
```

Multi-line list

To create a multiline list control:

```
var control = form_newControl(attrId, 'multilinelist', options);
```

This will create a multiline select element.

The options include:

source

The source of the list (an object; see simple list example)

addEmptyOption

Boolean indicating if an empty option should be added to the list

Radio button list

To create a radio button list control:

```
var control = form_newControl(attrId, 'radiolist', options);
```

This will create a list of radio elements.

The options include:

source

The source of the list (an object; see simple list example)

Checkbox list

To create a checkbox list control:

```
var control = form_newControl(attrId, 'checklist', options);
```

This will create a list of checkbox elements.

The options include:

source

The source of the list (an object; see simple list example)

GRID

Definition and properties

A grid is a complex type of attribute. In order to be able to use grids in a GARGANTUA form you must first define a type of grid , then you have to create an attribute of this type which can be used in the form.

A grid presents the data (content) in a table format and it allows the data to be added, viewed, modified in each editable cell. More, the grid can be customized to contain a set of internal features (defined by embedded functions and events handlers) but also to interact externally with other attributes of the form through form scripting. An attribute of grid type can be built assisted by the Grid Designer tool or it is fully scriptable and it can be build in pure Javascript language in a form script.

The grid properties:

`Columns`

defines the grid's structure

Header

defines the grid heading properties and style

Toolbar

provides built-in and user-defined features (actions)

Functions

collection of user-defined functions (Javascript source code)

Events

your own event handlers for the events exposed by the grid (Javascript source code)

Columns

The column properties:

id

the column identification string, it is unique for all columns and must respect the W3C recommandation for HTML ID attribute definition

width

numeric, represent the column width in pixels

type

the column's type **[string|icon|button|link|checkbox|list|date|time|timestamp|timespan|integer|float|currency|email|**

label

the column's name

tooltip

the column's tooltip

defaultValue

a value used for a corresponding cell when a new grid row is created; for now only columns having type **icon** (use any valid URL for an image or local images such as "${baseURL}/images/controls/grid/icons/grid-xxx.gif" where xxx is any of the values specified for toolbar icon) or **checkbox** (use value '1' or 'true' to set a checkbox as checked) are supported;

align

the cell text horizontal alignment **[left|center|right]**

`readonly`
: the cell is not editable, but it can be activated and clickable

`editable`
: the cell is not editable, it can not be activated nor clickable

`onload`
: for list types only - callback function to provide the options values

`resetonload`
: for list types only - specify if `onload` shall be always invoked;

Header

The grid heading has the following properties:

`visible`
: toggles the visibility of the header.

`compact`
: specifies if the header is displayed on a single or multiple lines.

Toolbar

The grid toolbar has the following properties:

`visible`
: toggles the visibility of the toolbar.

`align`
: specifies the alignment of the toolbar. `[top|right|bottom|left]`

The toolbar contains three types of buttons: `[standard|custom|separator]`.

Standard buttons:

`gridButtonAddRow`
: add a new row

`gridButtonRemoveRow`
: remove the selected row

`gridButtonMoveRowUp`
: move the selected row one position up

`gridButtonMoveRowDown`

move the selected row one position down

Button properties:

`type`

`[standard|custom|separator]`

`id`

the button identification string, it is unique for all buttons and must respect the W3C recommandation for HTML ID attribute definition

`visible`

toggle the button visible status (default yes)

`icon`

selected from a limited set of icons (”database”, “delete”, “disk”, “down”, “exclamation”, “eye”, “globe”, “minus”, “pencil”, “plus”, “question”, “star”, “tick-red”, “tick”, “up”)

`onclick`

associate an existing function as click event handler (triggered when the button is clicked)

`title`

tooltip text

`text`

label text

There is a button specific `onclick` event handler and a global `toolbarbuttonclicked` event handler triggered for all toolbar buttons.

The buttons order inside toolbar can be changed with drag and drop in Grid Designer UI.

Functions

The functions can be used to implement an internal feature or to be used as an event handler (ex: `onclick` handler).

A function is callable from another function like this:

```
this.callFunction(functionName, paramsObject);
```

Returned value is optional and shall be specified like this:

```
params.value = '...';
```

Events

Each event handler has two variables already defined: `this` (the grid object) and `params` object (this object is used to provide important parameters depending on event such as `params.row, params.col, params.from, params.to, params.id`).

`loaded`

triggerred when `load` method (grid definition and data) has been finished;

no params defined

`dataloaded`

triggerred when `loadData` method has been finished;

no params defined

`changed`

triggerred when a cell has been modified;

params defined: `params.row` (number - cell's row), `params.col` (number - cell's column)

`focused`

triggerred when a cell has been focused;

params defined: `params.row` (number - cell's row), `params.col` (number - cell's column)

`clicked`

triggerred when a cell has been clicked;

params defined: `params.row` (number - cell's row), `params.col` (number - cell's column)

`rowmoved`

triggerred when a grid row has been moved up or down;

params defined: `params.from` (number - row initial index), `params.to` (number - row final index)

`toolbarbuttonclicked`

triggerred when a toolbar button has been clicked;

params defined: `params.id` (string - button's id)

Example of event handler for `toolbarbuttonclicked`:

```
var buttonId = params.id;

if (buttonId == "cmdShowValue")
{
```

```
        alert( this.getValue());
}
else if (buttonId == "gridButtonAddRow")
{
        var row = this.getRowsCount() - 1;
        var iLink = this.getColumnIndexById("cLink");
        var iIcon = this.getColumnIndexById("cIcon");
        var iButton = this.getColumnIndexById("cButton");

        this.setCellValue(row, iLink, "This is link-"+row);
        this.setCellValue(row, iButton, "button-"+row);
        var base = "${baseURL}/images/controls/grid/icons/grid-";
        var arrIcons = ["question.gif","exclamation.gif"];
        this.setCellValue(row, iIcon, base + arrIcons[row % 2]);
}
else if (buttonId == "cmdAutoSize")
{
        this.autoSize();
}
```

Scripting the grid

Creating a grid object:

```
var grid = form_newControl(attrId, 'grid', options);
```

The options include:

url
: the URL from which to load the grid definition (XML)

definition
: the grid definition (XML)

loaded
: callback function invoked after the grid is initialized

The url and definition options are exclusive.

Load grid data:

```
grid.load(dataURL, true);
```

Event binding:

```
// bind 2 anonymous functions to the same event

grid. bindEvent('loaded', function() { log('loaded listener-1'); });

grid. bindEvent('loaded', function() { log('loaded listener-2'); });
```

```
// bind changed and focused events

grid. bindEvent('changed', function(param) { log('item ('+ param.row + ',' + param.col
+') changed' ); });

grid. bindEvent('focused', function(param) { log('item ('+ param.row + ',' + param.col
+') focused' ); });
```

Functions

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addToolbarButton

```
addToolbarButton(props)
```

Adds a button specified by provided props to the grid toolbar.

Parameters

props

An object with the button properties

Return value

None

Example

```
var props = {
    'id': 'open',
    'type': 'custom',
    'title': 'Open document',
    'text': 'Open'
};
grid.addToolbarButton(props);
```

append

```
append(noEdit)
```

Append a row at the end of the grid. It uses defaultValues to fill up the cells.

Parameters

`noEdit`

boolean that specifies if the append operation should not trigger the edit of the first visible cell in the newly added row

Return value

The newly added `tr` DOM element.

Example

```
grid.append();
```

or

```
var tr = grid.append();
```

appendBefore

```
appendBefore(tr, noEdit)
```

Append a row before the given table row. It uses defaultValues to fill up the cells.

Parameters

`tr`

the `tr` DOM element before which to append the new row

`noEdit`

boolean that specifies if the append operation should not trigger the edit of the first visible cell in the newly added row

Return value

The newly added `tr` DOM element.

Example

```
var tr = grid.getRow(2);
var newTr = grid.appendBefore(tr);
```

bindEvent

```
bindEvent(eventname, callback)
```

Bind a callback function to an event.

Parameters

eventName

`loaded | dataloaded | changed | focused | clicked | rowmoved | toolbarbuttonclicked`

The event to bind

callback

callback function assigned to event

Return value

Returns true for success, false if fails.

Example

```
// bind focus event
grid.bindEvent('focused', function(param) { log('item ('+ param.row + ',' + param.col
+') focused' ); });
```

appendAfter

```
appendAfter(tr, noEdit)
```

Append a row after the given table row. It uses defaultValues to fill up the cells.

Parameters

`tr`

the `tr` DOM element after which to append the new row

`noEdit`

boolean that specifies if the append operation should not trigger the edit of the first visible cell in the newly added row

Return value

The newly added `tr` DOM element.

Example

```
var tr = grid.getRow(2);
var newTr = grid.appendAfter(tr);
```

callFunction

```
callFunction(name, params)
```

Call an internal function (created with the Grid Designer UI) specified by `name` and `param` parameters object.

Parameters

`name`

The name of the function to call

`params`

The parameters for the function to call

Return value

None.

Example

editCell

```
editCell(td, options)
```

Activates a cell editor for the specified `td` DOM element.

Parameters

td

The cell to edit

options

An object with the parameters; for now the only available option is `triggerClick` (boolean - if `true`, force an additional click event)

Return value

None

Example

```
grid.editCell(td, {
    'trigger_click': true
});
```

editCheckbox

```
editCheckbox(td)
```

Activates a checkbox type cell for the specified `td` DOM element.

Parameters

td

The cell to edit

Return value

None

Example

```
grid.editCheckbox(td);
```

forceEditCell

```
forceEdit(td)
```

More than editCell, if the grid is read only, it will force the display of the cell control (in read only mode).

Parameters

td

The cell to activate edit for.

Return value

None

Example

```
grid.forceEditCell(td);
```

getCell

```
getCell(row, column)
```

Returns the cell as `td` DOM element specified by `row` index and `column` index.

Parameters

row

The row

column

The column

Return value

Returns the cell as `td` DOM element specified by `row` index and `column` index.

Example

```
var td = grid.getCell(1, 1);
```

getCellData

```
getCellData(row, column, name)
```

Returns the data having `name` key associated to a cell specified by `row` and `column`.

Parameters

row

The row of the cell

column

The column of the cell

name

The name of the property to retrieve

Return value

Returns the data having `name` key associated to a cell specified by `row` and `column`.

Example

```
var value = grid.getCellData(1, 2, 'test');
```

getCellValue

```
getCellValue(row, column)
```

Returns (string) the cell value specified by `row` index and `column` index.

Parameters

row

The row of the cell

column

The column of the cell

Return value

The cell value specified by `row` index and `column` index.

Example

```
var value = grid.getCellValue(2, 3);
```

getColsCount

```
getColsCount()
```

Return (number) the columns count.

Parameters

None

Return value

The columns count.

Example

```
var columns = grid.getColsCount();
```

getColumnIndexById

```
getColumnIndexById(columnId)
```

Returns the numeric index of a column specified by `columnId`, or -1 if not found.

Parameters

columnId

The column identifier

Return value

The numeric index of a column specified by `columnId`.

Example

```
var column = grid.getColumnIndexById('col1');
```

getColumnProps

```
getColumnProps(column)
```

Get the properties of a grid column specified by `column` index.

Parameters

column

The column index.

Return value

The properties (object) of a grid column.

Example

```
var props = grid.getColumnProps(1);
```

getRow

```
getRow(row)
```

Returns the row as `tr` DOM element specified by `row` index.

Parameters

row

The row index to retrieve

Return value

The row as `tr` DOM element specified by `row` index.

Example

```
var tr = grid.getRow(2);
```

getRowIndex

```
getRowIndex(tr)
```

Returns the numeric row index.

Parameters

tr

The DOM element row to retrieve the index for.

Return value

The numeric row index.

Example

```
var row = grid.getRowIndex(tr);
```

getRowsCount

```
getRowsCount()
```

Return (number) the rows count.

Parameters

None

Return value

The rows count.

Example

```
var rows = grid.getRowsCount();
```

getSelectedRow

```
getSelectedRow()
```

Returns the selected row `tr` DOM element or `null` if not found.

Parameters

None

Return value

Returns the selected row `tr` DOM element or `null` if not found.

Example

```
var tr = grid.getSelectedRow();
```

getToolbarButtonProps

```
getToolbarButtonProps(id)
```

Get the properties of the toolbar button identified by the given `id`.

Parameters

`id`

The button identifier

Return value

The properties (object) of a grid toolbar button.

Example

```
var props = grid.getToolbarButtonProps('open');
```

getValue

```
getValue()
```

Returns the grid value as JSON string.

Parameters

None

Return value

The grid data as JSON string.

Example

```
var data = grid.getValue();
```

hasError

```
hasError()
```

Returns `true` if the grid has at least one row or one cell marked as invalid or if the grid is marked as required but has not rows.

Parameters

None

Return value

`true` if the grid has at least one row or one cell marked as invalid or is empty while being required.

Example

```
if (grid.hasError())
{
    ...
}
```

groupRowsByColumn

```
groupRowsByColumn(column)
```

Scan the cells of the grid column specified by `column` index from top to bottom and merge the cells having the same value.

Parameters

column

The column index to scan

Return value

None

Example

isCellEditable

```
isCellEditable(row, column)
```

Returns true if a specified cell is editable.

Parameters

row

The row that contains the cell

column

The column that contains the cell

Return value

`true` if the specified cell is editable.

Example

```
var isEditable = grid.isCellEditable(5, 4);
```

isCellReadOnly

```
isCellReadOnly(row, column)
```

Returns true if a specified cell is read only.

Parameters

row

The row that contains the cell

column

The column that contains the cell

Return value

`true` if the specified cell is read only.

Example

```
var isReadOnly = grid.isCellReadOnly(5, 4);
```

isFunction

```
isFunction(name)
```

Returns `true` if a function specified by `name` exists (created with the Grid Designer UI).

Parameters

name

The function name to check for

Return value

`true` if the function specified by `name` exists.

Example

```
grid.isFunction('internal_fxn');
```

hasRowErrors

```
hasRowErrors(row)
```

Returns `true` if the specified row has any invalid cell.

Parameters

row

The row to check

Return value

`true` if a specified row has any invalid cell.

Example

isCellInvalid

```
isCellInvalid(row, column)
```

Returns true if the cell specified by `row` and `column` is marked as invalid (has error).

Parameters

row

The row that contains the cell

column

The column that contains the cell

Return value

`true` if the cell specified by `row` and `column` is marked as invalid.

Example

isEmpty

```
isEmpty()
```

Returns `true` if grid is empty.

Parameters

None

Return value

`true` if grid is empty.

Example

isReadOnly

```
isReadOnly()
```

Returns `true` if the grid is read only.

Alias for `isEnabled()`.

Parameters

None

Return value

`true` if the grid is read only.

Example

```
if (!grid.isReadOnly())
{
   ...
}
```

layout

```
layout()
```

Refresh the grid layout, called after a UI option has been changed.

This is called internally after updating the header, toolbar or toolbar buttons.

Parameters

None

Return value

None

Example

```
grid.layout();
```

moveup

```
moveup(tr)
```

Move the row specified by `tr` DOM element one position up. If tr not specified, move the selected row.

Parameters

tr

The row to move

Return value

None

Example

```
var tr = grid.getSelectedRow();
grid.moveup(tr);
```

movedown

```
movedown(tr)
```

Move the row specified by `tr` DOM element one position down. If tr not specified, move the selected row.

Parameters

tr

The row to move

Return value

None

Example

```
var tr = grid.getSelectedRow();
grid.movedown(tr);
```

remove

```
remove(tr)
```

Delete the row specified by `tr`. If tr not specified, delete the selected row.

Parameters

tr

The row to remove

Return value

None

Example

```
var tr = grid.getSelectedRow();
grid.remove(tr);
```

loadData

```
loadData(url, data)
```

Loads the grid content from a specified url. It triggers `dataloaded` when finished.

Parameters

url

The endpoint for data

data

The parameters to pass with the request

Return value

None

Example

removeAll

```
removeAll()
```

Delete all rows.

Parameters

None

Return value

None

Example

```
grid.removeAll();
```

setCellColor

```
setCellColor(row, column, color)
```

Set a cell background color. The color is a CSS string.

Parameters

row

The row that contains the cell

column

The column of the cell

color

The color to set; the color is a CSS string.

Return value

None

Example

```
grid.setCellColor(0, 0, 'red');
grid.setCellColor(0, 0, '#888');
```

selectRow

```
selectRow(tr)
```

Selects the row specified by the `tr` DOM element.

Parameters

tr

The row to select

Return value

None

Example

```
grid.selectRow(grid.getRow(0)); // select first row!
```

setCellData

```
setCellData(row, column, name, value)
```

Set a data `value` of any type, having the `name` key to a cell specified by `row` and `column`.

Parameters

row

The row of the cell

column

The column of the cell

name

The name of the property to set

value

The value to set

Return value

None

Example

```
var pt = { x:100, y:200 };
grid.setCellData( 0, 0, 'point', pt);
grid.setCellData( 0, 0, 'hasPoint', true);
```

setCellEditable

```
setCellEditable(row, column, status)
```

Set/Unset cell editable.

Parameters

row

The row of the cell

column

The column of the cell

status

Boolean to indicate editable status

Return value

None

Example

```
grid.setCellEditable(0, 0, false);
```

setCellInvalid

```
setCellInvalid(row, column, value)
```

Mark a cell as invalid if `value` is true.

Parameters

row

The row of the cell

column

The column of the cell

value

Cell validity indicator

Return value

None

Example

```
grid.setInvalidCell(0, 0, true);  // invalid
grid.setInvalidCell(0, 1, false); // valid
```

setCellReadOnly

```
setCellReadOnly(row, column, status)
```

Set/Unset cell specified by `row` index, `column` index as read only.

Parameters

row

The row of the cell

column

The column of the cell

status

Boolean to indicate read-only status

Return value

None

Example

```
grid.setCellReadOnly(0, 0, true);
```

setCellValue

```
setCellValue(row, column, value)
```

Set the cell value.

Parameters

row

The row of the cell

column

The column of the cell

value

The value to set

Return value

None

Example

```
grid.getCellValue(2, 3, 'John Doe');
```

setColumnReadOnly

```
setColumnReadOnly(column, status)
```

Set/Unset cells in a column specified by column `index` as read only.

Parameters

column

The column of the cell

status

Boolean to indicate read-only status

Return value

None

Example

```
grid.setColumnReadOnly(1, true);
```

setReadOnly

```
setReadOnly(readonly)
```

Set/Unset grid readonly.

Aliases for `disable()`, `enable()`.

Parameters

readonly

The read-only status for the grid

Return value

None

Example

```
grid.setReadOnly(true);
```

setRowInvalid

```
setRowInvalid(row, value)
```

Mark a row as invalid.

Parameters

row

The row to set as invalid.

value

Row validity indicator

Return value

None

Example

```
grid.setRowInvalid(1, true);
```

setRowReadOnly

```
setRowReadOnly(row, status)
```

Set/Unset a grid row read only.

Parameters

row

The row to set.

value

Row read-only indicator

Return value

None

Example

```
grid.setRowReadOnly(1, true);
```

setValue

```
setValue(value)
```

Sets the value of the grid. Value must be a stringified JSON.

Parameters

value

The new grid content

Return value

None

Example

```
grid.setValue('{"col1":[1,2,3],"col2":["John","Maria","Alice"]}');
```

unbindEvent

```
unbindEvent(eventname)
```

Unbind all binded event handlers for an event.

Parameters

eventName

The name of the event to unbind handlers for

Return value

Returns `true` for success, `false` if fails.

Example

```
grid.unbindEvent('focused');
```

updateEditCell

```
updateEditCell(hide)
```

Saves the value from the editor into grid cell; if `hide` boolean is true then close the editing input.

Parameters

hide

Flag to signal that input field should be destroyed, too.

Return value

None

Example

```
grid.updateEditCell(true);
```

updateToolbarButton

```
updateToolbarButton(id, props)
```

Update the properties of the toolbar button identified by the given `id`.

Parameters

`id`

The id of the toolbar button

`props`

The button properties to change

Return value

None

Example

```
grid.updateToolbarButton('btn1', { 'visible': true, 'text': 'New button' });
```

updateToolbar

```
updateToolbar(props)
```

Update the properties of the grid toolbar.

Parameters

props

The toolbar properties to change

Return value

None

Example

```
grid.updateToolbar({ 'visible': true, 'align': 'top' });
```

updateHeader

```
updateHeader(props)
```

Update the properties of the grid header.

Parameters

props

The header properties to change

Return value

None

Example

```
grid.updateHeader({ 'visible': true, 'compact': false });
```

Other input fields

File input

To create a file control:

```
var control = form_newControl(attrId, 'file', options);
```

The options include:

dnd

Boolean indicating if the control should be rendered as a drag-and-drop area or as a simple input file

Phone input

To create a phone control:

```
var control = form_newControl(attrId, 'phone', options);
```

The options include:

initialCountry

The ISO code of the default country to select

placeholderNumberType

The number type to use for the placeholder `[MOBILE|FIXED_LINE|FIXED_LINE_OR_MOBILE]`

autoPlaceholder

Set the input's placeholder to an example number for the selected country, and update it if the country changes `[polite|aggressive]`

FORM CALLBACK FUNCTIONS

FUN_FORM_PRE_LOAD

```
fun_form_pre_load()
```

Callback function called before creating the form controls and filling the form values.

Parameters

None.

Return value

None.

Example

```
function fun_form_pre_load() {
    try {
        // request necesary data
        var result = form_invokePlugin({
            pluginId: 'publicis',
            command: 'NomLists',
            data: {}
        });
        var json = JSON.parse(result);

        // cache data for later use
        // ...
   }
   catch (e) {
       form_error(e);
   }
}
```

FUN_FORM_LOAD

```
fun_form_load()
```

Callback function called after the form controls have been initialized and the form values have been set.

PARAMETERS

None.

RETURN VALUE

None.

EXAMPLE

```
function fun_form_load() {
    // init form values with defaults
    setElementKey(gfxGetAttrIdByLabel('userId'), getElementValue('username'));

    // init custom buttons
    form_displayCustomButtonBar({ ... });
}
```

FUN_FORM_POST_LOAD

```
fun_form_post_load()
```

Callback function called at the very end of the form load sequence.

Parameters

None.

Return value

None.

Example

FUN_FORM_DISPLAYCONTROLS

```
fun_form_displayControls(displayMode)
```

Callback function called after enabling/disabling the controls based on the display mode.

PARAMETERS

displayMode

The current display mode of the form, which can be one of the following values (both symbolic and numeric constants can be used):

```
0 = DISPLAY_NORMAL
1 = DISPLAY_READONLY
2 = DISPLAY_DISABLEDINPUTS
4 = DISPLAY_DISABLEDBUTTONS
```

RETURN VALUE

None.

EXAMPLE

FUN_FORM_PRE_LOADATTRS

```
fun_form_pre_loadattrs(data)
```

Callback function called before setting the attribute values inside the form.

Parameters

data

An object containing two objects: attributes and rights. The attributes object contains the values to set (each object consists of an attributeId-attributeValue pairing). The rights object contains information on weather the user can view and/or modify the values.

Return value

None.

Example

FUN_FORM_POST_LOADATTRS

```
fun_form_post_loadattrs(data)
```

Callback function called after setting the attribute values inside the form.

PARAMETERS

data

An object containing two objects: attributes and rights. The attributes object contains the values to set (each object consists of an attributeId-attributeValue pairing). The rights object contains information on weather the user can view and/or modify the values.

RETURN VALUE

None.

EXAMPLE

FUN_FORM_PENDING_VALIDATE

```
fun_form_pending_validate()
```

Callback function called before actually starting the validation process.

Parameters

None.

Return value

None.

Example

FUN_FORM_PRE_VALIDATEATTRS

```
fun_form_pre_validateattrs(data)
```

Callback function called before starting the validation process.

PARAMETERS

data

An object containing attributeId-attributeValue pairings that will be sent on the server for validation.

RETURN VALUE

None.

EXAMPLE

FUN_FORM_PRE_VALIDATEREPONSE

```
fun_form_pre_validatereponse(failedIds, failedReasons, failedCodes)
```

Callback function called immediately after receiving the validation response but before marking the failed values. Allows you to modify the attributes that will be marked as failed by adding or removing from the `failed*` arrays.

The arrays must be kept in sync, so if you add something to the "failedIds" you have to also add to the other two arrays as well.

Parameters

failedIds

Array of attribute ids that have failed validation.

failedReasons

Array of reasons that describe the failure.

failedCodes

Array of failure codes.

Return value

None.

Example

```
function fun_form_pre_validatereponse(failedIds, failedReasons, failedCodes) {
            var date = gfxVal('date');
            var startDate = gfxVal('startDate');


            if (startDate.trim() !== '' && date.trim() !== '') {
                var sdate = form_parseDate(startDate);
                sdate.setHours(0);
                sdate.setMinutes(0);
                sdate.setSeconds(0);
                sdate.setMilliseconds(0);
```

```
                var adate = form_parseDate(date);
                adate.setHours(0);
                adate.setMinutes(0);
                adate.setSeconds(0);
                adate.setMilliseconds(0);


                if (sdate.getTime() < adate.getTime()) {
                    failedIds.push(gfxGetAttrIdByLabel('startDate'));
                    failedCodes.push('invalid');
                    failedReasons.push('Start date cannot be lower than the date.');
                }
            }
}
```

FUN_FORM_VALIDATE

```
fun_form_validate(result)
```

Callback function called immediately after after marking the failed values.

Parameters

result

An object containing the validation outcome.

The object looks like this:

```
{
    validation : {
        enable : true | false,
        outcome : {
            labeled : true | false,
            anonymous : true | false,
            format : true | false, // true if there are no format failures
            mandatory : true | false, true if there are no mandatory failures
            regex : true | false, // true if there are no regex failures
            formatRegex : true | false, // true if there are no format and regex
failures
            validate : true | false, // true if validation succeeded
            label : true | false, // true if the label is filled
        },
        fulltext : true | false,
    }
}
```

Return value

None.

Example

FUN_FORM_POST_VALIDATEREPONSE

```
fun_form_post_validatereponse(failedIds, failedReasons, failedCodes)
```

Callback function called immediately after after marking the failed values. You might use it to enable/disable the form buttons based on the validation state.

PARAMETERS

failedIds

Array of attribute ids that have failed validation.

failedReasons

Array of reasons that describe the failure.

failedCodes

Array of failure codes.

RETURN VALUE

None.

EXAMPLE

```
function fun_form_post_validatereponse(failedIds, failedReasons, failedCodes) {
      // disable buttons if form not valid
      form_setCustomButtonEnabled('submit', failedIds.length === 0);
}
```

FUN_FORM_GET_VALIDATIONMESSAGE

```
fun_form_get_validationmessage(id, label, reason, code)
```

Callback function that offers the possibility to customize the validation messages that appear as tooltip on invalid controls.

Parameters

id

The invalid attribute's ID.

label

The invalid attribute's label.

reason

The default validation message.

code

The validation code.

Return value

The custom message. If `null`, `empty` or `undefined` is returned then the default validation message will be used.

Example

```
function fun_form_get_validationmessage(id, label, reason, code)
{
  form_info(id + ", " + label);

  switch (code)
  {
    case 'mandatory': // This field is mandatory. You must fill it.
      return 'Custom mandatory message';

    case 'length': // The value entered in this field is too long.
      if (label === 'myattribute')
        return 'The value exceeds the accepted size.';
      else
        return null;
```

```
    // case 'format': // Please check the validity of the data.

    // case 'listsize': // You have selected too many elements from this multiple
selection list.

    // case 'regex': // The value does not match the regular expression specified
for validation.

    // case 'emailsize': // You have selected too many e-mails.

    // case 'mandatorycolumn': // This grid contains mandatory cells. You must fill
them.


    default:

      return null;

  }

}
```

FUN_FORM_PRE_SUBMITFORM

```
fun_form_pre_submitform(formId, parentId, objectId)
```

Callback function called before submitting the form. Can stop the form submit if returned value is `false`.

Parameters

formId

The id of the current form

parentId

The id of the parent object

objectId

The id of the current object

Return value

The return value should be a boolean. If `false` then the form submit is stopped, otherwise the submit is allowed to continue.

Example

```
function fun_form_pre_submitform(formid, parentid, objectid) {
    generateDocumentNumber();
    return true;
}
```

In the example above a function that generates a number is called before allowing the form to submit.

You can also make certain validations and allow or stop the form submit accordingly.

FUN_FORM_CONFIRM_SUBMITFORM

```
fun_form_confirm_submitform(options)
```

Callback function called after `fun_form_pre_submit` and before actually submitting the form. Allows the display of a confirmation message before submitting the form.

PARAMETERS

options

An object containing the command name, the context and subcontext of the call.

RETURN VALUE

An object containing the title, message and buttons to display in the confirmation dialog. You can also specify a callback function to be called when one of the buttons is clicked.

EXAMPLE

```
function fun_form_confirm_submitform(options) {
    switch(options.cmd) {
        case 'save':
        case 'saveClose':
        case 'zap':
            return {
                title : 'Save',
                message : 'Are you sure you want to save this ?',
                callback : function(e, key, btn) {
                    console.log('pressed : ' + btn);

                    switch (btn)
                    {
                        case Button.YES:
                            // do something
                            break;

                        case Button.NO:
```

```
                            // do something
                            break;

                        default:
                            break;
                    }
                }
            };

         default:
            break;
     }
}
```

```
function fun_form_confirm_submitform(options) {
    return {
        title : 'Save',
        message : 'Are you sure you want to save this ?',
        buttons : {
                yes : {
                    label : 'Valider',
                    className : 'btn-default',
                    callback : function(e) {
                        console.log('validate');
                     }
                },
                no : {
                     label : 'Fermer'
                }
            }
    };
}
```

FUN_FORM_POST_SUBMITFORM

```
fun_form_post_submitform(formId, parentId, objectId, success)
```

Callback function called after submitting the form.

PARAMETERS

formId

The id of the current form

parentId

The id of the parent object

objectId

The id of the current object

success

Boolean indicating if the submit was successful or not

RETURN VALUE

None.

EXAMPLE

```
function fun_form_post_submitform(formId, parentId, objectId, success) {
    form_displayAlert({
        type : success ? 'success' : 'error',
        message : success ? 'Success message ...' : 'Error message ...'
    });
}
```

FUN_FORM_PRE_SENDTASK

```
fun_form_pre_sendtask(formId, parentId, objectId)
```

Callback function called before finishing a task. Can stop finish task if returned value is `false`.

This callback is called only for forms displayed in inbox.

Parameters

formId

The id of the current form

parentId

The id of the parent object

objectId

The id of the current object

Return value

The return value should be a boolean. If `false` then the task finish is stopped, otherwise the finish operation is allowed to continue.

Example

FUN_FORM_GET_EDITNATIVE_OPTIONS

```
fun_form_get_editnative_options(id)
```

Callback function called when an edit native operation is requested in order to get the overwrite options for it.

PARAMETERS

id

The id of the document being edited.

RETURN VALUE

Return a JSON object containing the version property with one of the following values: `overwrite | major | minor`.

EXAMPLE

```
function fun_form_get_editnative_options(id) {
    // add check if the doc's parent id is the current process and compute edit native
options
    return {
        version : 'major' // minor or overwrite
    };
}
```

FUN_FORM_REFRESH_DOCS

```
fun_form_refresh_docs()
```

Callback function called when documents were added, edited, deleted, pinned, unpinned, pasted, rolled back or splitted.

Parameters

None.

Return value

None.

Example

```
function fun_form_refresh_docs() {
    // check existence of a certain document using form_hasDocuments
or form_getDocuments methods
}
```

FUN_FORM_PRE_ESCALATETASK

```
fun_form_pre_escalatetask(formId, parentId, objectId)
```

Callback function called before escalating a task. Can stop escalate task if returned value is `false`.

This callback is called only for forms displayed in inbox.

PARAMETERS

formId

The id of the current form

parentId

The id of the parent object

objectId

The id of the current object

RETURN VALUE

The return value should be a boolean. If `false` then the task escalate is stopped, otherwise the escalate operation is allowed to continue.

EXAMPLE

FUN_FORM_GET_OCR_ATTRIBUTES

```
fun_form_get_ocr_attributes()
```

Callback function called in order to retrieve the list of attribute ids that can be used when performing OCR.

Parameters

None.

Return value

The function should return an array of valid attribute ids.

Example

```
var ocrAttrs = ['label', 'alfa', 'text']; // or ids


function fun_form_get_ocr_attributes() {
    var ids = [];
    for (var i = 0; i < ocrAttrs.length; i++)
        ids.push(gfxGetAttrIdByLabel(ocrAttrs[i]));


    return ids;
}
```

FUN_FORM_SET_OCR_ATTRIBUTES

```
fun_form_set_ocr_attributes(attributes)
```

Callback function called after an OCR text to attribute association (after setting the values that were retrieved through OCR).

PARAMETERS

attributes

Array of objects containing attribute id to value pairs.

RETURN VALUE

None.

EXAMPLE

```
function fun_form_set_ocr_attributes(attributes) {
    form_info(attributes);
    // perform necessary actions post OCR value set
}
```

FUN_FORM_POST_SIGN_DOCUMENTS

```
fun_form_post_sign_documents(data)
```

Callback function called after a document sign performed in the sign viewer (according to the task configuration). This allows you to finish the current task after the user has signed all documents.

PARAMETERS

data

An object containing :

- `ids` - array of signed document ids
- `count` - number of documents still left to sign

RETURN VALUE

None.

EXAMPLE

```
function fun_form_post_sign_documents(data) {
    form_info('Signed : ');
    form_info(data.ids);
    form_info('No. documents left to sign : ' + data.count);

    if (data.count == 0)
        form_finish();
}
```

FUN_FORM_RESET

```
fun_form_reset()
```

Callback function called after a form rest was performed.

PARAMETERS

None.

RETURN VALUE

None.

EXAMPLE

Services available through AJAX requests

ajax_syncServerRequestXML

```
function ajax_syncServerRequestXML(strServerUrl)
```

Invokes a service identified by the given URL.

Parameters

strServerUrl

Identifier for the invoked service.

Return value

Response in XML format from the invoked service.

Example

/API/SEARCH/ATTR?REQUEST

Request format

```
<Request> ::= <ConditionIds> & <Criteria>
<ConditionIds> ::= <IdAttr> "=" <valueAttr> | <IdAttr> "=" <valueAttr> "&"
<ConditionIds>
<Criteria> ::= <CriteriaObject> | <CriteriaObject> "&" <Criteria>
<CriteriaObject> ::= "processes=1" | "folders=1" | "documents=1" | "fiches=1" |
"pins=1"
<valueAttr> ::= attribute value
<IdAttr> ::= the ID of the attribute prefixed with DFAT
```

Example:

```
var aReq = "DFAT00010000001000000000000000000000000=siatel";
aReq += "&";
aReq += "folders=1&documents=1";
var xmlResponse = ajax_syncServerRequestXML( '/api/search/attr?' + aReq );
```

Response format

Returns the IDs of the objects (files, documents, processes, etc.)

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<search>
    <results>
       <result>
          <id>DOCF000100000005000000000000000100000000<id>
       </result>
       <result>
          <id> FLDR00010000000500000000000000000000000</id>
       </result>
    </results>
</search>
```

/API/GET?REQUEST

Request format

```
<request> ::= <param1> & <param2>
<param1> ::= "id=" <idObj>
<param2> ::= "mode=" <valueMode>
<valueMode> ::= "simple" | "full"
<idObj> ::= object identifier preceded by an object prefix
```

Example:

```
var aReq = "id=DOCF000100000005000000000000000100000000";
aReq += "&";
aReq += "mode=simple";
var xmlResponse = ajax_syncServerRequestXML( '/api/get?' + aReq );
```

Response format

Returns information about an object.

The full mode is currently not implemented.

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<object>
    <objectid> DOCF000100000005000000000000000100000000
    </objectid>
    <objectlabel> test document </objectlabel>
    <objectdesc></objectdesc>
    <objectattrs>
       <objectattr>
          <attrid>DFAT000100000010000000000000000000000000</attrid>
          <attrvalue>Example</attrvalue>
       </objectattr>
    </objectattrs>
</object>
```

INFORMATION ACCESSIBLE FROM THE FORM

Information accessible from the form :

userid

ID of the currently connected user.

username

User name of the currently connected user.

userrealname

Real name of the currently connected user.

search

Indicates whether the form is used in search mode.

multi

Indicates whether the form is used in multi-mode.

ext_taskName

Task name.

ext_processName

Process name.

ext_processDefName

Process definition name.

Example

```
var taskName = document.getElementById( "ext_taskName" ).value;
```

DEPRECATED APIS

This section lists the deprecated functions for forms; these should be replaced with GARGANTUA 8 APIs.

form_executeURL, form_refreshDocumentList, form_displayDocumentListToggle, form_displayDocumentList, form_displayDocumentListOnReturn, form_processAreaVerticalMaximize, form_processAreaResizeToggle, form_processAreaResize, form_setCustomButtonHidden

Process scripts

What is a process script?

Gargantua processes allow you to automate and standardize most of the processes performed as part of real business processes.

> ***Gargantua processes are executed on the Gargantua server and not on the client machines.***

They can be conceived using **Business Process Modeling Notation** (BPMN) on Gargantua Administration Tool. The Gargantua server contains a workflow engine which executes the tasks and evaluates the conditions to determine the process path (next task).

The Gargantua engine executes predefined actions (send an e-mail, add to EDM, etc.) and custom actions. These custom actions are, amongst others, those which use scripts.

This document **aims for developers skilled** in both JavaScript and Java, and who have previously finished their general training on processes.

Where and why to use scripts in a process

Most of the time, scripts are defined in the body of automatic tasks (custom actions). They are also used in sub-process start tasks and custom events.

In an automatic task

When the workflow engine gets to an automatic task, it executes its script. These tasks' scripts allow you, for example, to **generate values** (sequential identifiers) and documents or to communicate with **software external to Gargantua**.

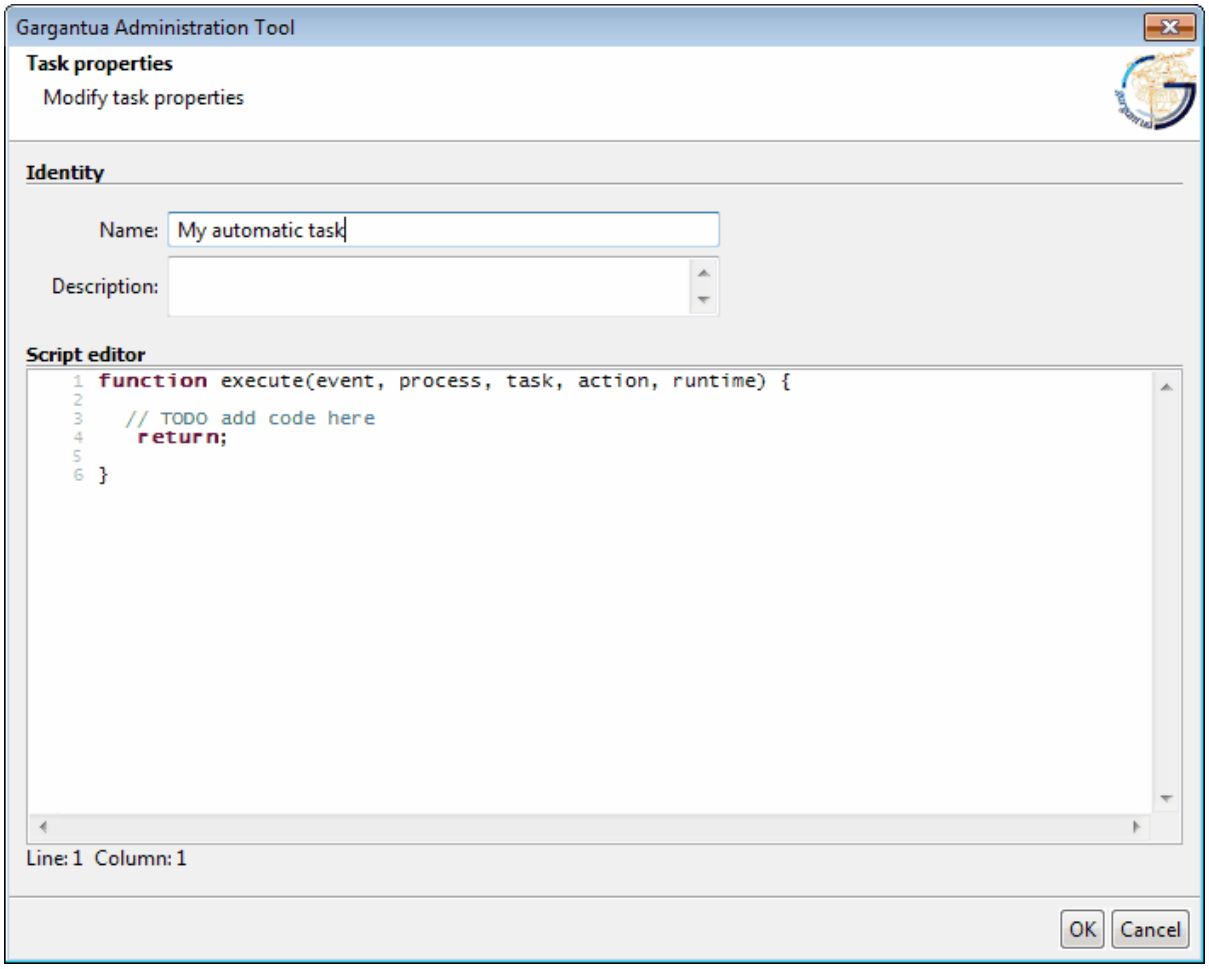
To insert a script in an automatic task, proceed as follows:

1. In the process graph, insert an **Automatic task** from the available workflow elements palette:

2. Double-click the automatic task.

   The task properties window displays. The **Script editor** area allows you to modify the script associated to this task:



IN A SUB-PROCESS START TASK

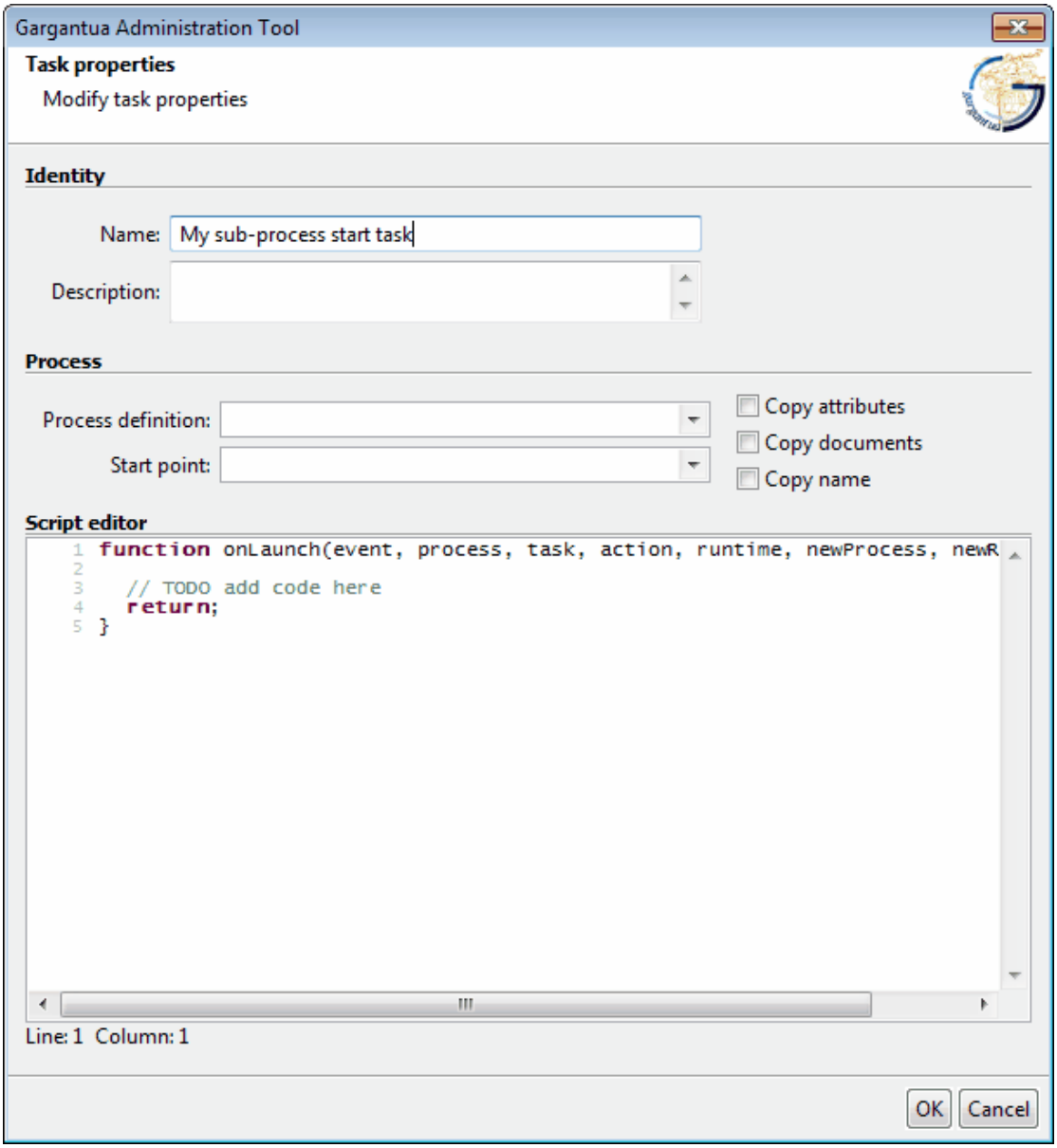
By using scripts, it is also possible to customize the way a process launches another process. This kind of scripts allows you to **accurately control the data transmitted** by the caller process to the called process.

To insert a script in a sub-process start task, proceed as follows:

1. In the process graph, insert a **Start another process** task from the available workflow elements palette:

2. Double-click the sub-process start task.

The task properties window displays. The **Script editor** area allows you to modify the script associated to this task:



In a custom event

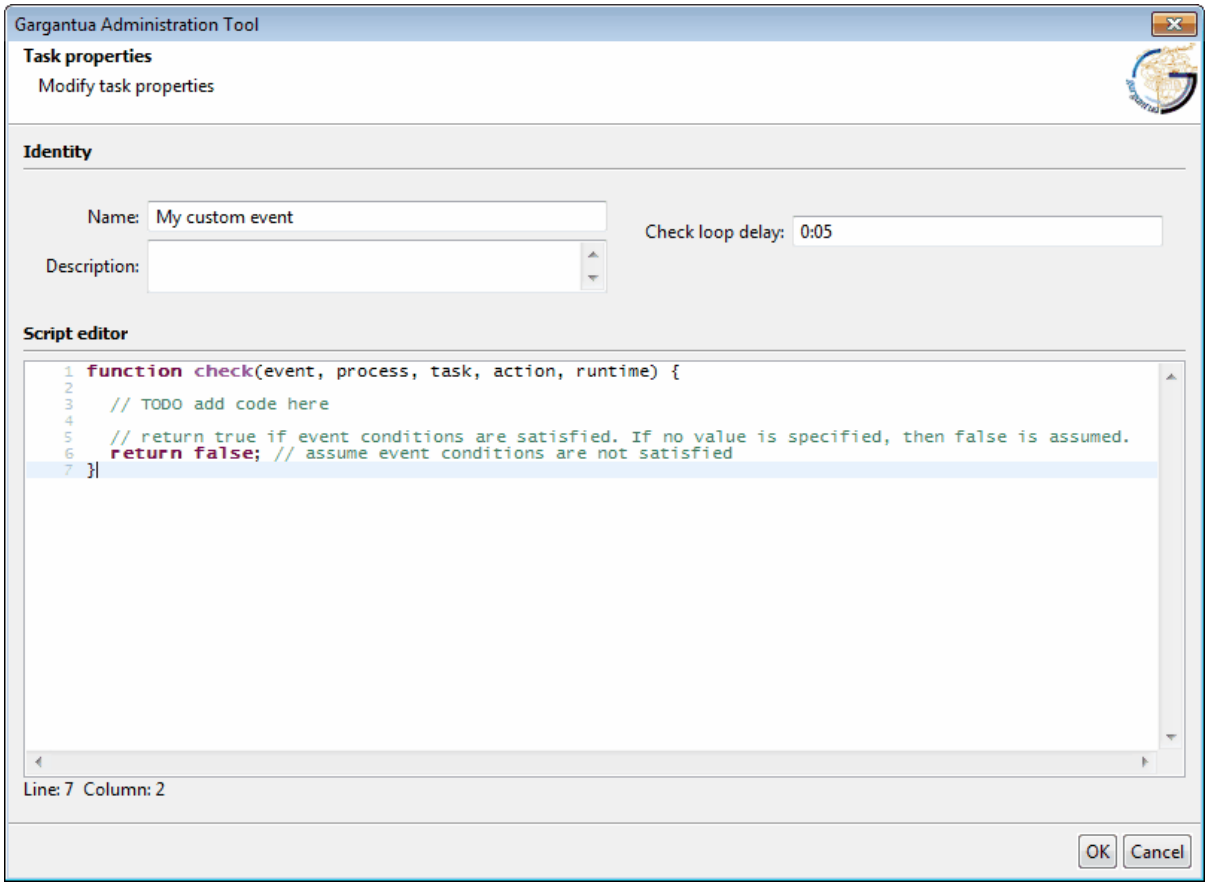
Scripts can be used in custom events, which are points of the process where the workflow engine pauses the workflow execution until an event occurs. For this to be possible, the workflow engine regularly executes a script, which performs the necessary actions to detect if the event occurred, and then returns a boolean to inform the engine about the event having occurred (`true`) or not (`false`).

To insert a script in a custom event, proceed as follows:

1. In the process graph, insert a **Custom event** task from the available workflow elements palette:

2. Double-click the event.

   The event properties window displays. The **Script editor** area allows you to modify the script associated to this event:



SUMMARY TABLE

The following table summarizes the different entry points and the main associated script functions:

Entry points	Roles
 Automatic task	Automate as much of a process as possible, particularly: • Automatic value generation (ex: IDs); • Automatic document generation; • Communication with external software.

 Sub-process start task	Configure accurately the way the sub-process is launched and the information transmitted to it.
 Custom event	Detect whether an event occurs, particularly to know if the execution of a sub-process is finished.

The script editor is always included in the window that displays by double-clicking the corresponding graph element.

Process script syntax and available classes

To keep it simple, the syntax of Gargantua process scripts is that of JavaScript, and the classes and functions library is that of Java, extended by the Gargantua API classes.

JavaScript reminders valid for process scripts

JavaScript language offers a wide range of possibilities that are rarely used. We will start with two examples, and then introduce a more advanced feature of this language that can be handy when creating process scripts.

Declaring associative tables

In JavaScript, you can declare associative tables (equivalent to Java's Map interface). To do so, proceed as follows:

```
// declaration of an associative table
var paul = {
   name: "Paul",
   age: 35
};
```

Associative tables can be nested in each other. In that case, the inner elements can be accessed using the . operator (point):

```
paul.name
```

DECLARING A VARIABLE

To declare a variable, enter the `var` keyword, followed by the variable's name:

```
var MyVariable;
```

DECLARING A FUNCTION

The simplest way to declare a function is by using the `function` keyword, followed by the function's name, its parameters in brackets and the instructions to be executed in a `{ ... }` block:

```
function sayHello(name) {
    return "Hello, " + name;
}
```

In JavaScript, a function is a value just like the rest, and can be stored in a variable. That is, the function above can also be written (with the exact same effects) the following way:

```
var sayHello = function(name) {
    return "Hello, " + name;
};
```

JAVASCRIPT AND JAVA AND GARGANTUA FRAMEWORKS

When developing GARGANTUA process scripts, you can take advantage of the flexibility of the JavaScript language combined with the richness of the Java framework and the GARGANTUA API:

- Java classes are available through **predefined associative tables**. Thus, you can instantiate them using their Java name.

  For example, in process scripts, you can write:

  ```
  var file = new java.io.File("myfile.txt");
  ```

  or:

  ```
  var buffer = new java.lang.StringBuilder();
  buffer.append("Hello");
  System.out.println(buffer.toString());
  ```

- In GARGANTUA processes, the JavaScript code from the scripts also accesses all the **GARGANTUA framework classes** (see the documentation on GARGANTUA API). A preview of the possibilities available with the GARGANTUA framework is explained in the sections that follow and in some practical exercises of this document.

Summary

When writing process scripts, you need to use the JavaScript language syntax, but you can manipulate classes and objects from the Java framework by using their entire name (with their path).

Entry points (functions called by the workflow engine)

There are three types of script-based workflow elements: automatic tasks, sub-process start tasks and custom events. For each type of elements, the associated script must define an entry point in the form of a JavaScript function.

It is never necessary to manually create these functions, as they are generated automatically in the body of scripts corresponding to tasks and custom events.

In sub-process start tasks: "onLaunch" function

The `onLaunch()` function, executed when calling a sub-process, admits two more parameters than the `execute()` function:

```
function onLaunch(event, process, task, action, runtime, newProcess, newRuntime)
```

As the `process` and `runtime` parameters concern the calling process, the `newProcess` and `newRuntime` parameters concern the called process. It is then possible to transmit data from one to the other using only this four variables.

For this function, it is not necessary to define a return value.

In automatic task scripts: "execute" function

When creating an automatic task, a default script is defined. This script already contains the `execute` function, which represents the entry point. Its definition is the following:

```
function execute(event, process, task, action, runtime)
```

Parameters and returned value:

event

"Event" object type, corresponding to the event which has triggered the script execution.

process

"Process" object type, containing information about the ongoing process.

task

Task associated to the script.

action

This parameter is not useful for the moment, but it may in further versions of GARGANTUA.

runtime

Process execution data. It is a "Map" object type used as common context for all the tasks of the same workflow. Thus, you can place data in it if you want to use them in other tasks.

returned value

This function does not return any value.

IN CUSTOM EVENTS: "CHECK" FUNCTION

The `check` function needs to be implemented in the following custom events:

```
function check(event, process, task, action, runtime)
```

Its parameters are the same as those of the `execute()` function when executing an automatic task, but the `check()` function returns a boolean:

- `true` means that the event has occurred. The workflow will get to the following step and the `check()` function will not be called again.
- `false` means that the event has not occurred. After a time, the `check()` function will be called again to check if the event has occurred.

MANAGING PROCESS SCRIPTS

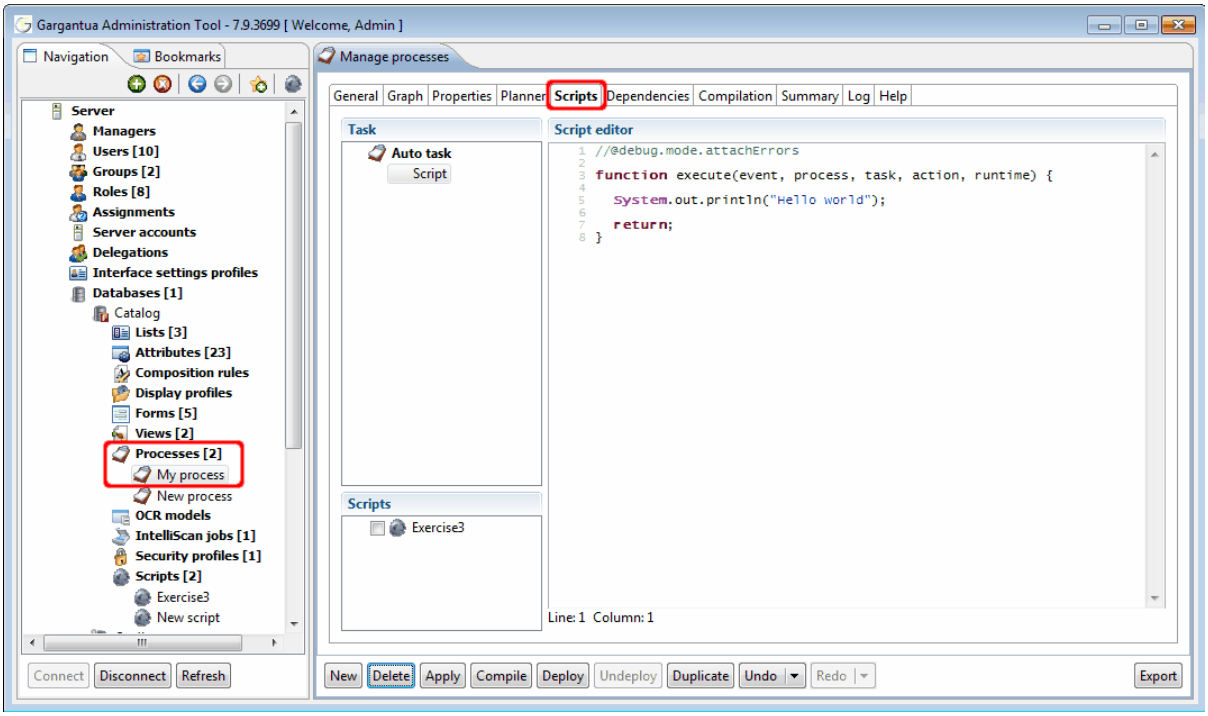
AT GARGANTUA DATABASE LEVEL

It is recommended to define as many scripts at GARGANTUA database level as possible (see Creating a script at GARGANTUA database level).

All the **Workflow** type scripts of the GARGANTUA database are available in the **Scripts** tab of each process management screen. You can check the checkbox corresponding to the script to include it in a process element.

At process level

Instead of double-clicking on each task or event containing a script, you can access all the scripts of a process in the **Scripts** tab of each process management screen:



The **Scripts** frame, in the bottom left corner of the right pane, allows you to include scripts defined at GARGANTUA database level.

Exercise: Writing your first script

Instructions

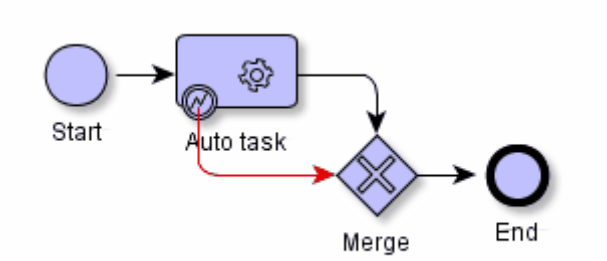
Create the simplest possible workflow, in which a script can be executed using an automatic task. In this script, display the “Hello world” message.

Hint: you need to use `System.out.println`.

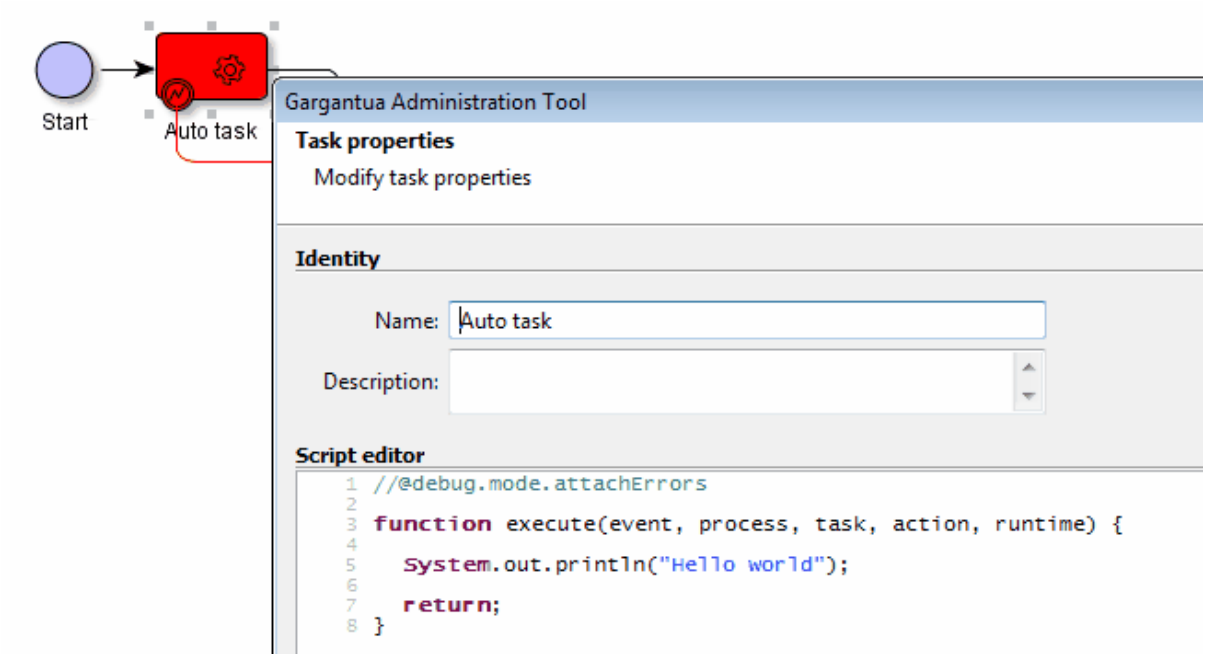
Extra question: where does this message display?

CORRECTION

This is the simplest workflow allowing to execute a script:



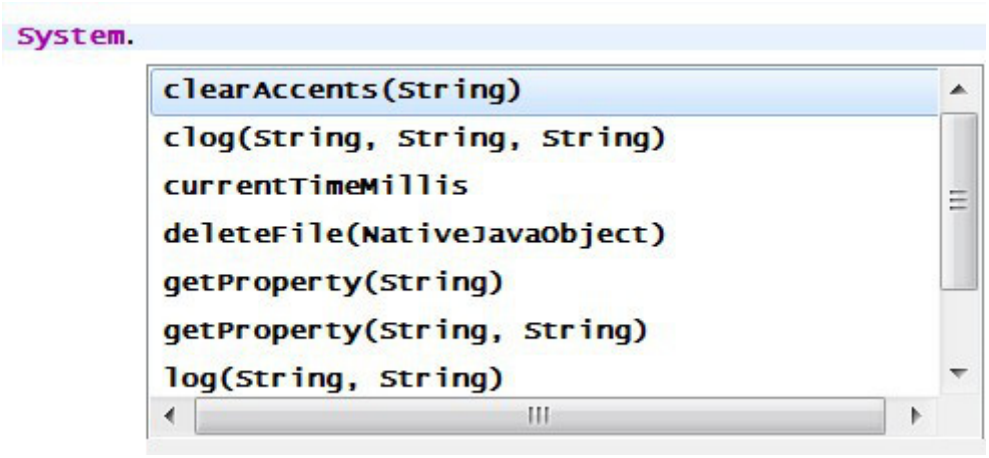
This is the script:



The “Hello world” message will be added to the server logs, which is not really useful when developing a script. There are other techniques to retrieve information about script execution.

Hint: displaying the contextual choices list

When writing a script, a contextual choices list may display:



If this list does not display automatically, you can use the Ctrl + space shortcut to display it.

Debugging

There are two techniques for debugging: error files inclusion and the debugger.

Including error files to the workflow

When adding a particular directive to a script, the exceptions it generates are logged in a file attached to the following task.

To enable this debug mode, you need to add the following directive at the beginning of the script:

```
//@debug.mode.attachErrors
```

It is important not to enter any spaces.

Using the debugger

Using the debugger is the easiest solution, but it requires to work directly on the GARGANTUA server. Once the server is configured, you need to add the directive that suits the best to the scripts you want to debug. A fully featured debug window displays when one of these scripts is about to be executed.

STARTING THE SERVER IN COMMAND MODE

Once you have configured the server, it needs to be started in command mode. To do so, proceed as follows:

1. Stop the **Gargantua 7 Server** service.
2. Access the **apache-tomcat\bin** folder in the installation folder of your GARGANTUA server (by default, **C:\siatel\Gargantua7server**).
3. Execute the **startup.bat** file.

   The debugger is ready to use.

CONFIGURING THE SERVER TO USE THE DEBUGGER

To use the debugger, a particular server configuration is needed. To configure the server, proceed as follows:

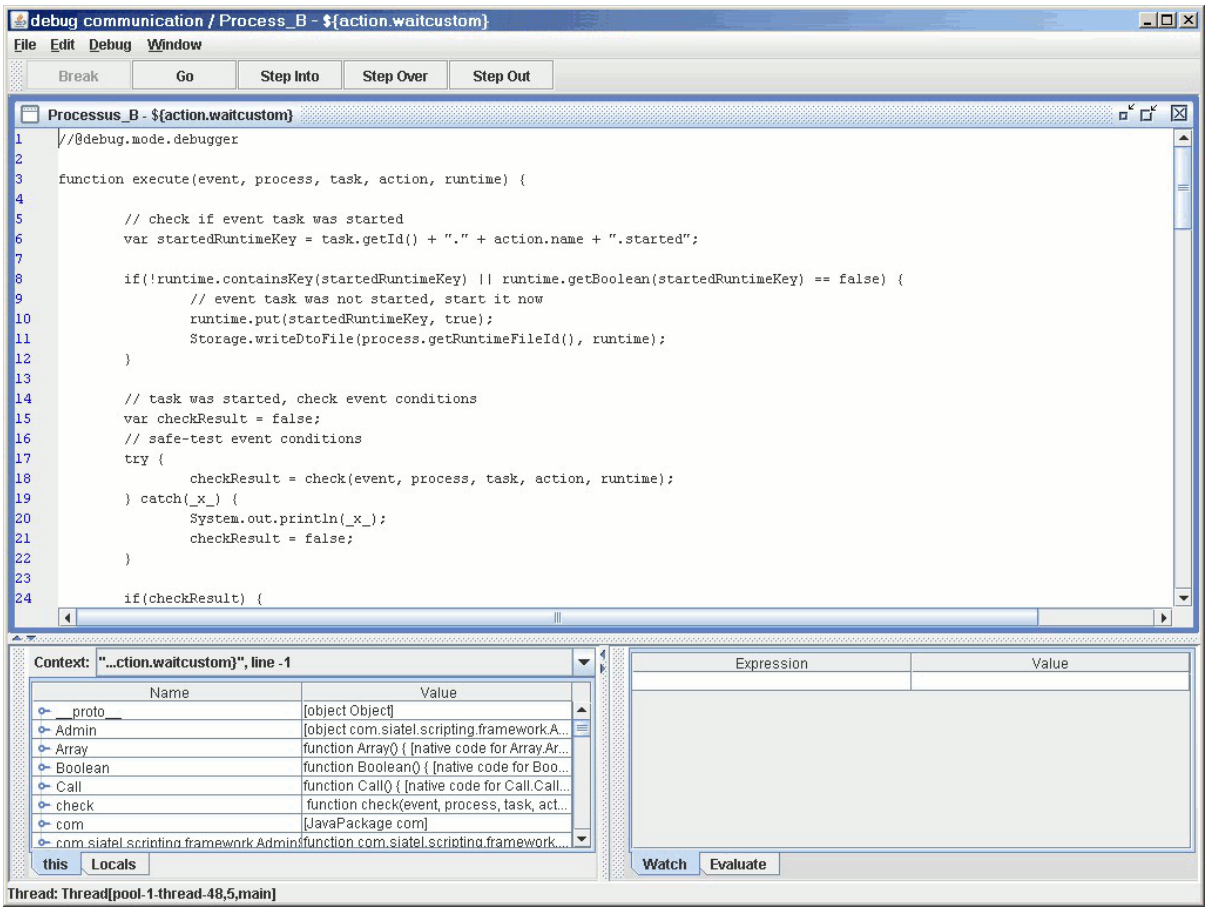
1. Access the **apache-tomcat\bin** folder in the install folder of your GARGANTUA server (by default, **C:\siatel\Gargantua7server**).
2. Open the **catalina.bat** file in a text editor.
3. Search the line that starts by “set JAVA_OPTS”.
4. Before this line, add a new line that contains the following text: set JAVA_OPTS=%JAVA_OPTS% -Djs.debugging=enabled
5. Save the changes.

ADDING A DIRECTIVE BEFORE THE SCRIPTS

The directive to be added before the script is the one that follows:

```
//@debug.mode.debugger
```

The debugger window displays as follows:



Exercise

Test both of the debugging techniques explained before on the script you had created before.

Case study: updating the workflow using a ".properties" file

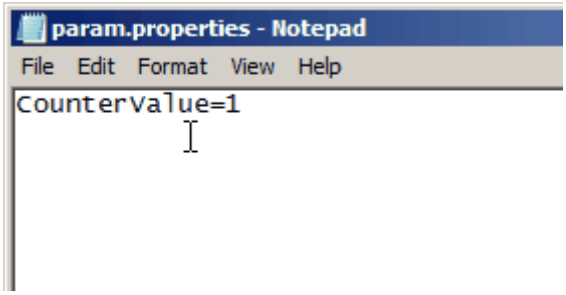
Problem

The objective is to store a numerical value in an external file (**.properties** file) so it can be used as automatic counter. Create a script that:

1. Reads the **.properties** file and retrieves its value.
2. Updates the form field.
3. Increases the value.
4. Updates the **.properties** file.

To do so, you have access to both Java framework and the Gargantua public API.

The **.properties** file is as follows:



CORRESPONDING SCRIPT

The following script solves the problem:

```
//@debug.mode.attachErrors

//Properties file path
var propertiesPath = "C:\\work\\test\\param.properties";
var properties;

/*
  Entry point
*/
function execute(event, process, task, action, runtime) {

  var number = loadNumber();
  majProcess(process, number);
  saveNumber( java.lang.Integer.parseInt(number)

  return;
}

/*
  Loading the number from the properties file
*/
function loadNumber() {
  var file = new java.io.File(propertiesPath);
  var inputStream = java.io.FileInputStream(file);

  properties = new java.util.Properties();
  properties.load(inputStream);

  return properties.getProperty("counterValue","1");
}

/*
  Process update
*/
function majProcess(process, number) {
  //Gargantua database ID recovery
  var databaseId = Idformat.toString(PrefiX.DATABASE, process.getDbId());

  //Attribute ID recovery
  var p1AttrId = Design.idOfAttr("p1", databaseId);

  //Form attribute update
  process.setAttributeValue(p1AttrId, number);

  //Save changes
  Flow.updateProcess(process, new java.lang.Long(Flag.ATTRIBUTES));
}

/*
  Saving the number
*/
function saveNumber(number) {
  properties.setProperty("counterValue", number);
  properties.store(new java.io.FileOutputStream(propertiesPath), new java.util.Date().toString());
}
```

The form attribute we update is called “p1”.

Explanation

The `chargeNumber()` and `saveNumber()` methods only contain standard Java framework functions. Thus, they are not explained here.

However, the `majProcessus()` method calls the GARGANTUA API.

Important note on data types

The script mentioned before contains the following code:

```
new java.lang.Long(Flag.ATTRIBUTES)
```

and the following code in the `execute` function:

```
java.lang.Integer.parseInt(numero) + 1
```

In process scripts, even if the variables accept any value, the operators can return different results depending on the used types.

Example:

- *`"1" + "1"` returns `"11"`.*
- *`"1" + 1` also returns `"11"`.*
- *However, `1 + 1` returns `2`.*

Furthermore, Java types are not processed as language types. For example, `java.lang.Integer` is not a numerical type but an object.

```
new java.lang.Integer(1) + 1 also returns "11".
```

In the background, it is the `toString()` method from `java.lang.Integer` that is called.

Yet, some methods from Java framework or the GARGANTUA API expect a precise Java object type. That is the case of `Flow.updateProcess`, which second parameter needs to be a `java.lang.Long`. Passing directly a JavaScript numerical type does not work, so it is necessary to create a `Long` object type before calling the method.

The types manipulated by the scripts language need to be distinguished from the types of the used API, and conversions from one to another need to be done even if errors can occur or if the obtained results are inconsistent.

USING THE GARGANTUA API TO UPDATE THE PROCESS DATA ("MAJPROCESSUS" METHOD)

The `majProcessus` method allows you to update the data of a process:

- ```
  var databaseId = IdFormat.toString(Prefix.DATABASE, process.getDbId());
  ```

  This source code retrieves the ID of the GARGANTUA database (40 characters). The numerical ID of the database that contains the current process can be retrieved using the `getDbId()` method, but this numerical ID is not the one that is expected by most of the GARGANTUA API framework methods. Thus, you need to use `IdFormat.toString()` to convert it to a standard 40-character GARGANTUA ID. The `Prefix.DATABASE` parameter specifies that the requested ID is a database ID.

- ```
  var p1AttrId = Design.idOfAttr("p1", databaseId);
  ```

  `Design.idOfAttr` returns the GARGANTUA ID of an attribute depending on its name and on the GARGANTUA ID of the database where it belongs.

- ```
  process.setAttributeValue(p1AttrId, numero);
  ```

  At this stage, the `process` variable contains a copy of the information about the process (and not the original process). The `setAttributeValue` method updates the value corresponding to the `p1AttrId` attribute ID of this memory copy.

  The variable needs two parameters: the attribute ID and the new value.

- ```
  Flow.updateProcess(process, new java.lang.Long(Flag.ATTRIBUTES));
  ```

  `Flow.updateProcess` updates the data of the real GARGANTUA process, from the copy of the process stored in memory (first parameter). The second parameter is used to choose what to synchronize.

When working on the GARGANTUA API, do not forget that there are two types of ID for the same objects: numerical ID and GARGANTUA ID, each one consisting of 40 alphanumerical characters. You can pass from one to the other using "IdFormat".

USING JAVA LIBRARIES

Using a Java library allows you to expand the potential of scripts, or simply to write less code lines. It is particularly useful when GARGANTUA needs to communicate with an external business application with a public Java API.

Installing a Java library

To install a Java library, proceed as follows:

1. Access the **apache-tomcat\lib** folder in the installation folder of your Gargantua server (by default, **C:\siatel\Gargantua7server**).
2. Paste the .jar file corresponding to the Java library.
3. Restart the **Gargantua 7 Server** service.

   The Java library is now available in the process scripts.

Exercise: Connecting to an external system using a Java API

In this exercise, you will apply everything you have learned until now on a frequently useful case: connecting to a database from a script.

MySQL is a RDBMS available under a GPL license, often used for web development. To install MySQL on Windows, you can use a package like EasyPHP or WAMP.

MySQL will be the external system with which we will communicate. A Java API is provided by its JDBC connector.

Instructions

Create a process that allows you to obtain the following result:

1. A receptionist enters the name of an enquirer and validates the form.
2. The “enquirer” table is updated using a script.
3. The form is returned to the receptionist with the ID generated by MySQL.

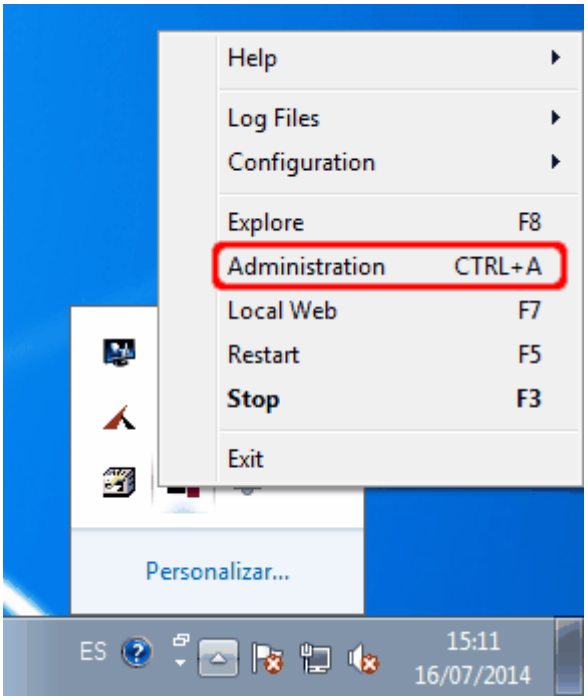
Preparation

To carry out this exercise, you need to take the following preparation steps:

1. Download and install EasyPHP.

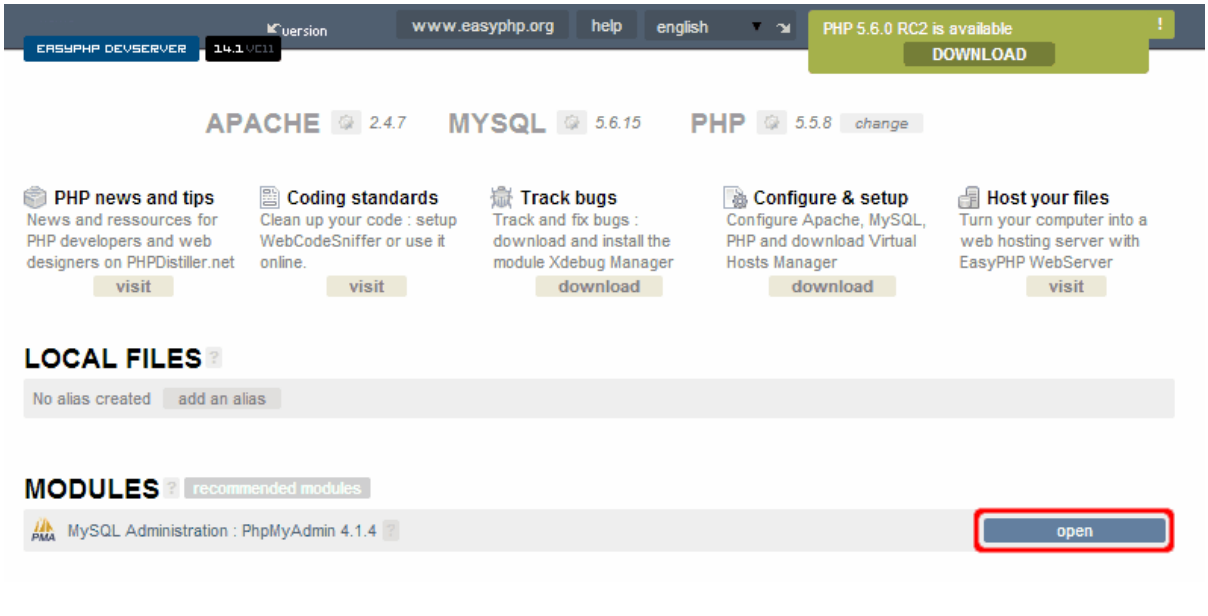
   A default MySQL user with all the rights is created automatically. Its name is “root” and its password is an empty text chain “”.

2. Right click the EasyPHP icon available in the notification area (right side of the task bar) , then select **Administration**:



The EasyPHP manager screen opens in the default web browser.

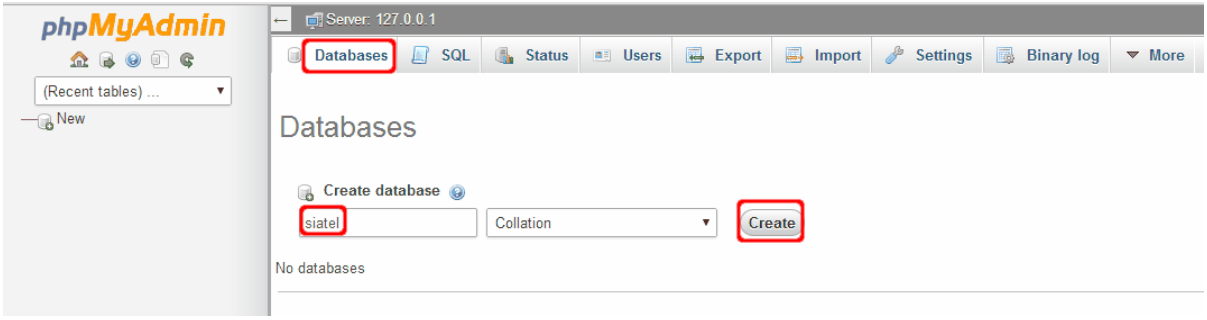
3. In the **Modules** section, click the **open** button:



phpMyAdmin, the database managing tool provided with the EasyPHP, opens in the default web browser.

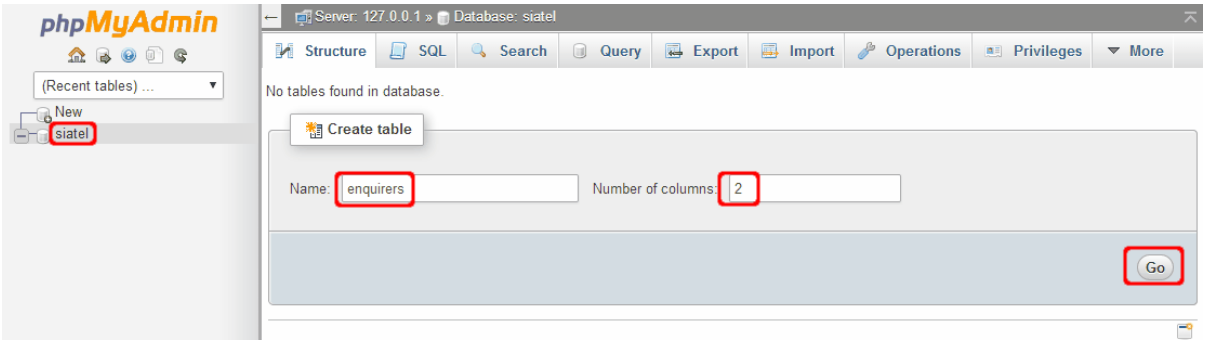
4. Click the **Databases** tab.
5. In **Create a database**, enter “siatel”.

6. Click the **Create** button:



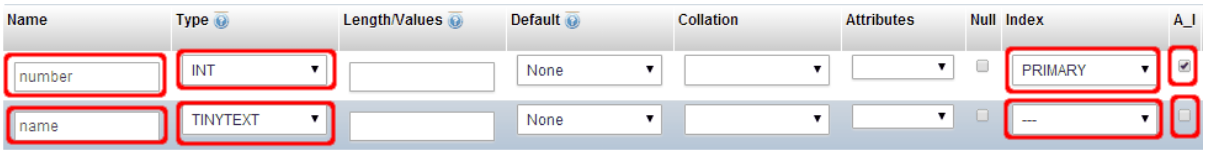
A message indicating that the “siatel” database has been created displays.

7. In the folder tree on the left side of the screen, click the **siatel** link.
8. In **Create table**, in the **Name** field, enter “enquirers”.
9. In the **Number of columns** field, enter “2”.
10. Click the **Go** button:



The “enquirers” table has been created.

11. To finish the definition of the table, enter the characteristics of both columns as shown below:



12. Click the **Save** button.

    The external system is now ready to use: the aim of this exercise is to insert data in the “enquirers” table and to recover the numbers created automatically by MySQL.

13. Download the **.jar** file of the MySQL connector (mysql-connector-java).

HELP: EXAMPLE OF A CONNECTION THROUGH JDBC TO MYSQL IN JAVA

The aim of the following information is to spare you, if you want, from searching the syntax of the connection to MySQL through JDBC.

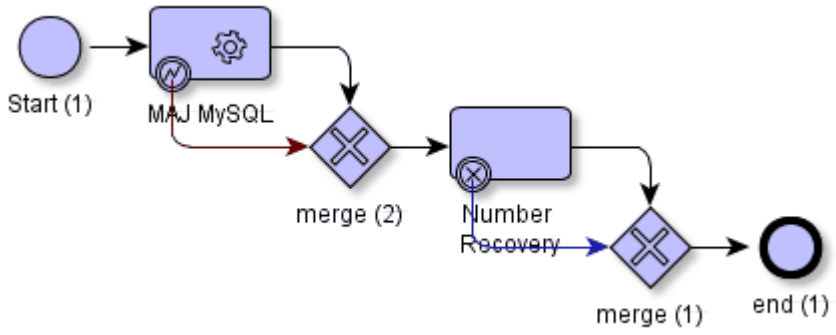
The following Java source code inserts a name into the table and recovers the generated ID:

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
                                    ...
final String url = "jdbc:mysql://localhost/siatel";
final String user = "root";
final String psswd = "";

Class.forName("com.mysql.jdbc.Driver");

Connection connection = DriverManager.getConnection(
        url,
        user,
        psswd
);

final Statement statement = connection.createStatement();
statement.executeUpdate(
        "INSERT INTO siatel.enquirers (`name`) VALUES ('Patrick')"
);

ResultSet generatedKeys = statement.getGeneratedKeys();
if (generatedKeys.next()) {
    System.out.println(generatedKeys.getLong(1));
}

statement.close();
connection.close();
```

CORRECTION

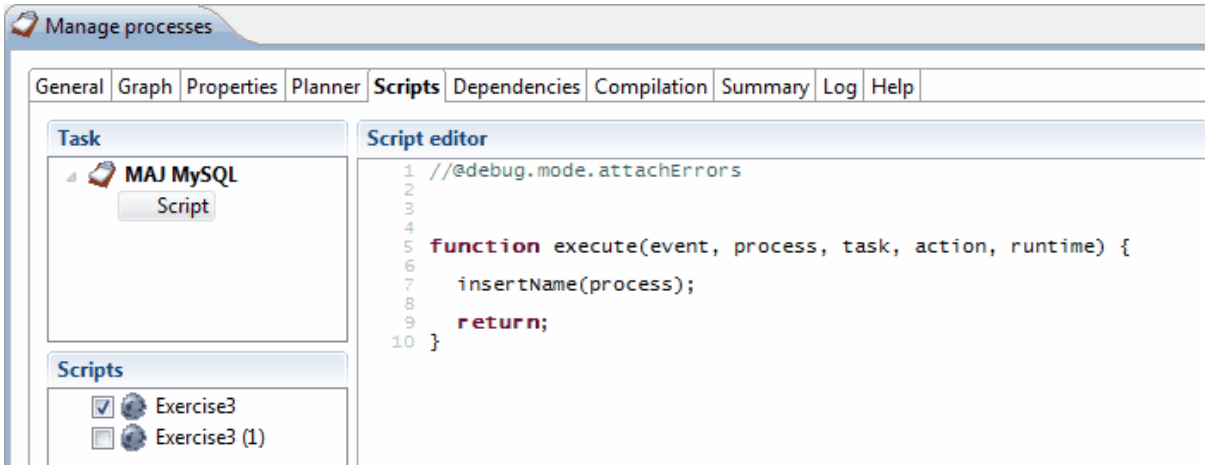
The following procedure allows you to solve the exercise:

1. Create two attributes:
    - A “MySQL number” integer number attribute;
    - A “Name & surname” alphanumerical attribute.
2. Create a form, named for example “Exercise 3”, that contains both attributes.
3. Design the following process:



- At the beginning, the receptionist enters the name and the surname.
- The “MAJ MySQL” automatic task contains the script.
- Its two exit paths merge.
- The resulting path redirects to the “Number recovery” user task, which allows the receptionist to view the number generated by MySQL in the form. During the script development, this task also allows you to see the exceptions.
- Whatever is the exit of this step (cancellation or validation), the workflow stops (merge and end).

4. Access the **apache-tomcat\lib** file in the install folder of your GARGANTUA server (by default, **C:\siatel\Gargantua7server**).
5. In this folder, paste the MySQL connector file, **mysql-connector-java-*.*.*-bin.jar**, downloaded during the preparation for the exercise.
6. Restart the **Gargantua 7 Server** service.
7. Add a **Workflow** type script named, for example, “Exercise 3” to the level of the GARGANTUA database, then insert it in the script of the “MAJ MySQL” automatic task:



Here, the "insertName" method defined in the "Exercice3" script of the GARGANTUA database is called.

8. Using the Java script provided before, create the following script in the **Scripts** branch of the GARGANTUA database:

```
var url = "jdbc:mysql://localhost/siatel";
var user = "root";
var psswd = "";

function insertName(process) {

    var dbId = IdFormat.toString(Prefix.DATABASE, process.getDbId());

    var nom = process.getAttributeValue(Design.idOfAttr("Name & surname", dbId));


    java.lang.Class.forName("com.mysql.jdbc.Driver");

    var connection = java.sql.DriverManager.getConnection(
            url,
            user,
            psswd
            );

    var statement = connection.createStatement();
    statement.executeUpdate("INSERT INTO siatel.enquirers (`name`) VALUES ('"
            + name + "')");

    var generatedKeys = statement.getGeneratedKeys();
    if (generatedKeys.next()) {
        process.setAttributeValue(Design.idOfAttr("mysql number", dbId),
                new java.lang.Integer(generatedKeys.getLong(1)));
        Flow.updateProcess(process, new java.lang.Long(Flag.PROCESS_ALL));
    }

    statement.close();
    connection.close();

}
```

This script accesses the process form attributes the same way it does in the case study example.

Scripting API

PACKAGE COM.SIATEL.SCRIPTING.FRAMEWORK

CLASSES

ADMINSCRIPTING

```
public class AdminScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning administration functionalities. The methods in this class are bound to the JavaScript `Admin` namespace object's methods and properties.

Author :

victorc

Constructor summary :

Constructor
AdminScripting ()
AdminScripting (Scriptable scope, Scriptable prototype)

Method summary :

Modifier and type	Method
public final void	readDatabase (NativeJavaObject object, NativeJavaObject flags)
public final void	enumerateDatabases (NativeJavaObject list, NativeJavaObject flags)
public final void	enumerateDatabasesOffsetCount (NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public final java.lang.String	idOfDatabase (java.lang.String label)
public final void	createUser (NativeJavaObject object)
public final void	readUser (NativeJavaObject object, NativeJavaObject flags)
public final void	updateUser (NativeJavaObject object, NativeJavaObject flags)
public final void	deleteUser (java.lang.String userId)

public final void	enumerateUsers (java.lang.Object criteria, NativeJavaObject list, NativeJavaObject flags)
public final void	enumerateUsersOffsetCount (java.lang.Object criteria, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public final java.lang.String	idOfUser (java.lang.String label)
public final void	createGroup (NativeJavaObject object)
public final void	readGroup (NativeJavaObject object, NativeJavaObject flags)
public final void	updateGroup (NativeJavaObject object, NativeJavaObject flags)
public final void	deleteGroup (java.lang.String groupId)
public final void	enumerateGroups (java.lang.Object criteria, NativeJavaObject list, NativeJavaObject flags)
public final void	enumerateGroupsOffsetCount (java.lang.Object criteria, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public final java.lang.String	idOfGroup (java.lang.String label)
public final void	createRole (NativeJavaObject object)
public final void	readRole (NativeJavaObject object, NativeJavaObject flags)
public final void	updateRole (NativeJavaObject object, NativeJavaObject flags)
public final void	deleteRole (java.lang.String roleId)
public final void	enumerateRoles (java.lang.Object criteria, NativeJavaObject list, NativeJavaObject flags)
public final void	enumerateRolesOffsetCount (java.lang.Object criteria, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public final java.lang.String	idOfRole (java.lang.String label)
public final void	enumeratePortal (NativeJavaObject criteria, NativeJavaObject list, NativeJavaObject flags)
public final void	enumeratePortalOffsetCount (NativeJavaObject criteria, NativeJavaObject list, NativeJavaObject

	flags, int offset, int count)
public final java.lang.String	idOfPortal (java.lang.String label)
public final void	readPlanner (NativeJavaObject object, NativeJavaObject flags)
public final void	enumeratePlanner (NativeJavaObject criteria, NativeJavaObject list, NativeJavaObject flags, Integer offset, Integer count)
public final void	enumerateStorages (NativeJavaObject criteria, NativeJavaObject list, NativeJavaObject flags)
public final void	enumerateStoragesOffsetCount (NativeJavaObject criteria, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public final java.lang.String	idOfStorage (java.lang.String label)
public final void	createResource (NativeJavaObject object)
public final void	readResource (NativeJavaObject object, NativeJavaObject flags)
public final void	updateResource (NativeJavaObject object, NativeJavaObject flags)
public final void	deleteResource (java.lang.String resourceId)
public final void	enumerateResource (NativeJavaObject criteria, NativeJavaObject list, NativeJavaObject flags)
public final void	enumerateResourceOffsetCount (NativeJavaObject criteria, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public final java.lang.String	idOfResource (java.lang.String label)
public final java.lang.String	idOfCertificate (java.lang.String label)
public final void	setObjectExtendedProperty (java.lang.String objectId, java.lang.String propertyKey, NativeJavaObject propertyType, NativeJavaObject propertyValue)
public final java.lang.Object	getObjectExtendedProperty (java.lang.String objectId, java.lang.String propertyKey)
public final void	enumerateExtendedPropertyForObject

	(NativeJavaObject map, java.lang.String objectId)
public final RegistryEntry	setRegistryEntry (java.lang.String userId, java.lang.String resourceId, java.lang.String key, java.lang.String value)
public final java.lang.String	getRegistryEntry (java.lang.String userId, java.lang.String resourceId, java.lang.String key)
public final void	deleteRegistryEntry (java.lang.String userId, java.lang.String resourceId, java.lang.String key)
public final void	enumerateRegistryEntries (NativeJavaObject list, java.lang.String userId, java.lang.String resourceId)
public final void	enumeratePositions (java.lang.Object criteria, NativeJavaObject list, NativeJavaObject flags, java.lang.Integer offset, java.lang.Integer count)
public final void	createPosition (NativeJavaObject object)
public final void	readPosition (NativeJavaObject object, NativeJavaObject flags)
public final void	updatePosition (NativeJavaObject object, NativeJavaObject flags)
public final void	deletePosition (java.lang.String objectId)
public final void	getSuperiors (java.lang.Object objectOrId, NativeJavaObject list)

Constructor detail :

```
public AdminScripting ()
```

Default constructor.

```
public AdminScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

scope -

prototype -

Method detail :

```
public final void readDatabase (NativeJavaObject object,
NativeJavaObject flags)
```

Database lazy-load.

JS call:

```
Admin.readDatabase(object, flags)
```

Parameters :

`object` - wrapped Database object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Database fields to initialize

Throws :

`SiatelException` - wrapped error

```
public final void enumerateDatabases (NativeJavaObject list,
NativeJavaObject flags)
```

Database enumeration. All databases are returned.

JS call:

```
Admin.enumerateDatabases(list, flags)
```

Parameters :

`list` - wrapped List to fill by this method with database objects

`flags` - wrapped Long value specifying Database fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public final void enumerateDatabasesOffsetCount (NativeJavaObject list,
NativeJavaObject flags, int offset, int count)
```

Database enumeration with offset and count.

Parameters :

`list` - wrapped List to fill by this method with database objects

`flags` - wrapped Long value specifying Database fields to initialize for enumeration results

`offset` - enumeration offset

`count` - enumeration count

Throws :

`SiatelException` - wrapped error

```
public final String idOfDatabase (String label)
```

Get the id of a database based on it's name.

JS call:

```
Admin.idOfDatabase(name)
```

Parameters :

`label` - the name of the database

Returns :

the id of the database. `null` if database of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public final void createUser (NativeJavaObject object)
```

Create a User object.

JS call:

```
Admin.createUser(object)
```

Parameters :

`object` - wrapped User object to create

Throws :

`SiatelException` - wrapped error

```
public final void readUser (NativeJavaObject object, NativeJavaObject
flags)
```

User lazy-load.

JS call:

```
Admin.readUser(object, flags)
```

Parameters :

`object` - wrapped User object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying User fields to initialize

Throws :

`SiatelException` - wrapped error

```
public final void updateUser (NativeJavaObject object, NativeJavaObject
flags)
```

User update.

JS call:

```
Admin.updateUser(object, flags)
```

Parameters :

`object` - wrapped User object to update. Must have the `id` set

`flags` - wrapped Long value specifying User fields to persist

Throws :

`SiatelException` - wrapped error

```
public final void deleteUser (String userId)
```

User delete.

JS call:

```
Admin.deleteUser(userId)
```

Parameters :

`userId` - id of user to delete

Throws :

`SiatelException` - wrapped error

```
public final void enumerateUsers (Object criteria, NativeJavaObject
list, NativeJavaObject flags)
```

User enumeration using criteria.

JS call:

```
Admin.enumerateUsers(criteria, list, flags)
```

Parameters :

`criteria` - a String role or group id, a Role object, a Group object or a wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_USERS for supported 'list' criteria format.

`list` - wrapped List to fill by this method with User objects

`flags` - wrapped Long value specifying User fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public final void enumerateUsersOffsetCount (Object criteria,
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

User enumeration using criteria, offset and count.

JS call:

```
Admin.enumerateUsersOffsetCount(criteria, list, flags, offset, count)
```

Parameters :

`criteria` - a String role or group id, a Role object, a Group object or a wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_USERS for supported 'list' criteria format.

`list` - wrapped List to fill by this method with User objects

`flags` - wrapped Long value specifying User fields to initialize for enumeration results

`offset` - enumeration start offset

`count` - maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public final String idOfUser (String label)
```

Get the id of a user based on it's name.

JS call:

```
Admin.idOfUser(label)
```

Parameters :

`label` - the user (account) name

Returns :

the id of the user

Throws :

`SiatelException` - wrapped error

```
public final void createGroup (NativeJavaObject object)
```

Create a Group object.

JS call:

```
Admin.createGroup(object)
```

Parameters :

`object` - wrapped Group object to create

Throws :

`SiatelException` - wrapped error

```
public final void readGroup (NativeJavaObject object, NativeJavaObject
flags)
```

Group lazy-load.

JS call:

```
Admin.readGroup(object, flags)
```

Parameters :

`object` - wrapped Group object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Group fields to initialize

Throws :

`SiatelException` - wrapped error

```
public final void updateGroup (NativeJavaObject object, NativeJavaObject
flags)
```

Group update.

JS call:

```
Admin.updateGroup(object, flags)
```

Parameters :

`object` - wrapped Group object to persist. Must have the `id` set

`flags` - wrapped Long value specifying Group fields to persist

Throws :

`SiatelException` - wrapped error

```
public final void deleteGroup (String groupId)
```

Group delete.

JS call:

```
Admin.deleteGroup(groupId)
```

Parameters :

`groupId` - id of Group to delete

Throws :

`SiatelException` - wrapped error

```
public final void enumerateGroups (Object criteria, NativeJavaObject
list, NativeJavaObject flags)
```

Group enumeration using criteria.

JS call:

```
Admin.enumerateGroups(criteria, list, flags)
```

Parameters :

`criteria` - a String role or user id, a Role object, a User object or a wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_GROUPS for supported 'list' criteria format.

`list` - wrapped List to fill by this method with Group objects

`flags` - wrapped Long value specifying Group fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public final void enumerateGroupsOffsetCount (Object criteria,
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

Group enumeration using criteria, offset and count.

JS call:

```
Admin.enumerateGroupsOffsetCount(criteria, list, flags, offset, count)
```

Parameters :

`criteria` - a String role or user id, a Role object, a User object or a wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_GROUPS for supported 'list' criteria format.

`list` - wrapped List to fill by this method with Group objects

`flags` - wrapped Long value specifying Group fields to initialize for enumeration results

`offset` - enumeration start offset

`count` - maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public final String idOfGroup (String label)
```

Get the id of a group based on it'a name.

JS call:

```
Admin.idOfGroup(label)
```

Parameters :

`label` - the group name

Returns :

the id of the group

Throws :

`SiatelException` - wrapped error

```
public final void createRole (NativeJavaObject object)
```

Create a Role object.

JS call:

```
Admin.createRole(object)
```

Parameters :

`object` - wrapped Role object to create

Throws :

`SiatelException` - wrapped error

```
public final void readRole (NativeJavaObject object, NativeJavaObject
flags)
```

Role lazy-load.

JS call:

```
Admin.readRole(object, flags)
```

Parameters :

`object` - wrapped Role object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Role fields to initialize

Throws :

`SiatelException` - wrapped error

```
public final void updateRole (NativeJavaObject object, NativeJavaObject
flags)
```

Role update.

JS call:

```
Admin.updateRole(object, flags)
```

Parameters :

`object` - wrapped Role object to persist. Must have the `id` set

`flags` - wrapped Long value specifying Role fields to persist

Throws :

`SiatelException` - wrapped error

```
public final void deleteRole (String roleId)
```

Role delete.

JS call:

```
Admin.deleteRole(roleId)
```

Parameters :

`roleId` - id of Role to delete

Throws :

`SiatelException` - wrapped error

```
public final void enumerateRoles (Object criteria, NativeJavaObject
list, NativeJavaObject flags)
```

Role enumeration using criteria.

JS call:

```
Admin.enumerateRoles(criteria, list, flags)
```

Parameters :

`criteria` - a String group or user id, a Group object, a User object or a wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_ROLES for supported 'list' criteria format.

`list` - wrapped List to fill by this method with Role objects

`flags` - wrapped Long value specifying Role fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public final void enumerateRolesOffsetCount (Object criteria,
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

Role enumeration using criteria, offset and count.

JS call:

```
Admin.enumerateRolesOffsetCount(criteria, list, flags, offset, count)
```

Parameters :

`criteria` - a String group or user id, a Group object, a User object or a wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_ROLES for supported 'list' criteria format.

`list` - wrapped List to fill by this method with Role objects

`flags` - wrapped Long value specifying Role fields to initialize for enumeration results

`offset` - enumeration start offset

`count` - maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public final String idOfRole (String label)
```

Get the id of a role based on it'a name.

JS call:

```
Admin.idOfRole(label)
```

Parameters :

`label` - the role name

Returns :

the id of the role

Throws :

`SiatelException` - wrapped error

```
public final void enumeratePortal (NativeJavaObject criteria,
NativeJavaObject list, NativeJavaObject flags)
```

Portal enumeration using criteria.

JS call:

```
Admin.enumeratePortal(criteria, list, flags)
```

Parameters :

`criteria` - wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_PORTALS for supported criteria.

`list` - wrapped List to fill by this method with Portal objects

`flags` - wrapped Long value specifying Portal fields to initialize for enumeration results.

Throws :

`SiatelException` - wrapped error

```
public final void enumeratePortalOffsetCount (NativeJavaObject criteria,
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

Portal enumeration using criteria, offset and count.

JS call:

```
Admin.enumeratePortalOffsetCount(criteria, list, flags, offset, count)
```

Parameters :

`criteria` - wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_PORTALS for supported criteria.

`list` - wrapped List to fill by this method with Portal objects

`flags` - wrapped Long value specifying Portal fields to initialize for enumeration results.

`offset` - enumeration start offset

`count` - maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public final String idOfPortal (String label)
```

Get the id of a portal based on it'a name

JS call:

```
Admin.idOfPortal(label)
```

Parameters :

`label` - the portal name

Returns :

the id of the portal

Throws :

`SiatelException` - wrapped error

```
public final void readPlanner (NativeJavaObject object, NativeJavaObject
```

```
flags)
```

Planner lazy-load.

JS call:

```
Admin.readPlanner(object, flags)
```

Parameters :

`object` - wrapped Planner object to initialize. Must have the `id` or `label` set

`flags` - wrapped Long value specifying Planner fields to initialize

Throws :

`SiatelException` - wrapped error

```
public final void enumeratePlanner (NativeJavaObject criteria,
NativeJavaObject list, NativeJavaObject flags, Integer offset, Integer
count)
```

Planner enumeration using criteria.

JS call:

```
Admin.enumeratePlanner(criteria, list, flags[, offset, count])
```

Parameters :

`criteria` - wrapped `List<String>` containing enumeration criteria.

`list` - wrapped List to fill by this method with Planner objects

`flags` - wrapped Long value specifying Planner fields to initialize for enumeration results.

`offset` - enumeration start offset

`count` - maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public final void enumerateStorages (NativeJavaObject criteria,
NativeJavaObject list, NativeJavaObject flags)
```

Storage enumeration using criteria.

JS call:

```
Admin.enumerateStorage(criteria, list, flags)
```

Parameters :

`criteria` - wrapped `List<String>` containing enumeration criteria. See

AdministrationLibrary#CMD_ENUMERATE_STORAGES for supported criteria.

`list` - wrapped List to fill by this method with Storage objects

`flags` - wrapped Long value specifying Storage fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public final void enumerateStoragesOffsetCount (NativeJavaObject
criteria, NativeJavaObject list, NativeJavaObject flags, int offset, int
count)
```

Storage enumeration using criteria, offset and count.

JS call:

```
Admin.enumerateStorage(criteria, list, flags, offset, count)
```

Parameters :

`criteria` - wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_STORAGES for supported criteria.

`list` - wrapped List to fill by this method with Storage objects

`flags` - wrapped Long value specifying Storage fields to initialize for enumeration results

`offset` - an enumeration start offset

`count` - a maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public final String idOfStorage (String label)
```

Get the id of a storage area based on it's name

JS call:

```
Admin.idOfStorage(label)
```

Parameters :

`label` - the storage area name

Returns :

the id of the storage area

Throws :

`SiatelException` - wrapped error

```
public final void createResource (NativeJavaObject object)
```

Create a Resource object

JS call:

```
Admin.createResource(object)
```

Parameters :

`object` - wrapped Resource object to create

Throws :

`SiatelException` - wrapped error

```
public final void readResource (NativeJavaObject object,
NativeJavaObject flags)
```

Resource lazy-load.

JS call:

```
Admin.readResource(object, flags)
```

Parameters :

`object` - wrapped Resource object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Resource fields to initialize

Throws :

`SiatelException` - wrapped error

```
public final void updateResource (NativeJavaObject object,
NativeJavaObject flags)
```

Resource update.

JS call:

```
Admin.updateResource(object, flags)
```

Parameters :

`object` - wrapped Resource object to persist. Must have the `id` set

`flags` - wrapped Long value specifying Resource fields to persist

Throws :

`SiatelException` - wrapped error

```
public final void deleteResource (String resourceId)
```

Resource delete.

JS call:

```
Admin.deleteResource(resourceId)
```

Parameters :

`resourceId` - id of Resource to delete

Throws :

`SiatelException` - wrapped error

```
public final void enumerateResource (NativeJavaObject criteria,
NativeJavaObject list, NativeJavaObject flags)
```

Resource enumeration using criteria.

JS call:

```
Admin.enumerateResource(criteria, list, flags)
```

Parameters :

`criteria` - wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_RESOURCES for supported criteria.

`list` - wrapped List to fill by this method with Resource objects

`flags` - wrapped Long value specifying Resource fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public final void enumerateResourceOffsetCount (NativeJavaObject
criteria, NativeJavaObject list, NativeJavaObject flags, int offset, int
count)
```

Resource enumeration using criteria, offset and count

JS call:

```
Admin.enumerateResource(criteria, list, flags, offset, count)
```

Parameters :

`criteria` - wrapped `List<String>` containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_RESOURCES for supported criteria.

`list` - wrapped List to fill by this method with Resource objects

`flags` - wrapped Long value specifying Resource fields to initialize for enumeration results

`offset` - an enumeration start offset

`count` - a maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public final String idOfResource (String label)
```

Get the id of a resource based on it's name

JS call:

```
Admin.idOfResource(label)
```

Parameters :

`label` - the resource name

Returns :

the id of the resource

Throws :

`SiatelException` - wrapped error

```
public final String idOfCertificate (String label)
```

Get the id of a certificate based on it's name

JS call:

```
Admin.idOfCertificate(label)
```

Parameters :

`label` - the certificate name

Returns :

the id of the certificate

Throws :

`SiatelException` - wrapped error

```
public final void setObjectExtendedProperty (String objectId, String
propertyKey, NativeJavaObject propertyType, NativeJavaObject
propertyValue)
```

Set an extended property

JS call:

```
Admin.setObjectExtendedProperty(objectId, propertyKey, propertyType,
propertyValue)
```

Parameters :

`objectId` - the id of the object for which the property is added

`propertyKey` - the key of the property

`propertyType` - the type of the property value. See Types for details.

`propertyValue` - the value of the property

Throws :

`SiatelException` - wrapped error

```
public final Object getObjectExtendedProperty (String objectId, String propertyKey)
```

Get an extended property value

JS call:

```
var propVal = Admin.getObjectExtendedProperty(objectId, propertyKey)
```

Parameters :

`objectId` - the id of the object

`propertyKey` - the key of the property

Returns :

wrapped Object containing the value of the property

Throws :

`SiatelException` - wrapped error

```
public final void enumerateExtendedPropertyForObject (NativeJavaObject map, String objectId)
```

Extended property enumeration for a certain object.

JS call:

```
Admin.enumerateExtendedPropertyForObject(map, objectId)
```

Parameters :

`map` - wrapped Map to fill by this method with (key, value) pairs

`objectId` - wrapped String specifying the id of the object whose properties are enumerated

Throws :

`SiatelException` - wrapped error

```
public final RegistryEntry setRegistryEntry (String userId, String
resourceId, String key, String value)
```

Assign some alphanumeric (string) data to a resource (database, folder, document etc) in the context of a specific user.

Parameters :

`userId` - the id of the user to whom the registry key belongs. Use Constants#SYSTEM for entries that are not to be used in the context of a user session

`resourceId` - the resource targeted by the registry key

`key` - the registry key (a string identifying the persistent data)

`value` - the data to store (string)

Returns :

a RegistryEntry object containing all the data received on input

Throws :

`SiatelException` -

```
public final String getRegistryEntry (String userId, String resourceId,
String key)
```

Read some previously assigned resource data, based on user context, resource id and key.

Parameters :

`userId` - the id of the user whose context is to be used for reading the data

`resourceId` - the targeted resource

`key` - the registry entry identification key

Returns :

the string data previously assigned, `null` if not existing

Throws :

`SiatelException` - wrapped error

```
public final void deleteRegistryEntry (String userId, String resourceId,
String key)
```

Delete some previously assigned resource data, based on user context, resource id and key.

Parameters :

`userId` - the id of the user whose context is to be used for deleting the data

`resourceId` - the targeted resource

`key` - the registry entry identification key

Throws :

`SiatelException` - wrapped error

```
public final void enumerateRegistryEntries (NativeJavaObject list,
String userId, String resourceId)
```

List the registry entries for a given resource, based on user context.

Parameters :

`list` - the java.util.List to fill with RegistryEntry objects

`userId` - the id of the user whose context is to be used for listing the data

`resourceId` - the targeted resource

Throws :

`SiatelException` - wrapped error

```
public final void enumeratePositions (Object criteria, NativeJavaObject
list, NativeJavaObject flags, Integer offset, Integer count)
```

Orgchart Position enumeration using criteria, offset and count.

JS call:

```
Admin.enumeratePositions(criteria, list, flags, offset, count)
```

Parameters :

`criteria` - a String user, group or role id, a User object, a Role object, a Group object or a wrapped {@code List<String>} containing enumeration criteria. See AdministrationLibrary#CMD_ENUMERATE_POSITIONS for supported 'list' criteria format

`list` - wrapped List to fill by this method with Position objects

`flags` - wrapped Long value specifying Position fields to initialize for enumeration results; optional, defaults to Flag#POSITION_ALL

`offset` - enumeration start offset; optional, defaults to 0

`count` - maximum number of objects to return; optional, defaults to 0 (which mean all objects)

Throws :

`SiatelException` - wrapped error

```
public final void createPosition (NativeJavaObject object)
```

Create an orgchart Position object.

JS call:

```
Admin.createPosition(object)
```

Parameters :

`object` - wrapped Position object to create

Throws :

`SiatelException` - wrapped error

```
public final void readPosition (NativeJavaObject object,
NativeJavaObject flags)
```

Orgchart Position lazy-load.

JS call:

```
Admin.readPosition(object, flags)
```

Parameters :

`object` - wrapped Position object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Position fields to initialize; optional, defaults to Flag#POSITION_ALL

Throws :

`SiatelException` - wrapped error

```
public final void updatePosition (NativeJavaObject object,
NativeJavaObject flags)
```

Orgchart Position update.

JS call:

```
Admin.updatePosition(object, flags)
```

Parameters :

`object` - wrapped Position object to update; must have the `id` set

`flags` - wrapped Long value specifying Position fields to persist

Throws :

`SiatelException` - wrapped error

```
public final void deletePosition (String objectId)
```

Orgchart Position delete.

JS call:

```
Admin.deletePosition(objectId)
```

Parameters :

`objectId` - id of position to delete

Throws :

`SiatelException` - wrapped error

```
public final void getSuperiors (Object objectOrId, NativeJavaObject
list)
```

Orgchart Position get superior list.

JS call:

```
Admin.getSuperiors(objectOrId, list)
```

Parameters :

`objectOrId` - id of position to query for superior or position itself

`list` - list object to fill with positions that match criteria

Throws :

`SiatelException` - wrapped error

AUDITSCRIPTING

```
public class AuditScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning audit functionalities. The methods in this class are bound to the JavaScript `Audit` namespace object's methods and properties.

Constructor summary :

Constructor
`AuditScripting ()`
`AuditScripting (Scriptable scope, Scriptable prototype)`

Method summary :

Modifier and type	Method
`public AuditEntry`	`newEntry ()`
`public void`	`logEntry (NativeJavaObject entry)`
`public void`	`log (java.lang.String resourceId, java.lang.String resourceName, int operation, int result, int duration)`

Constructor detail :

```
public AuditScripting ()
```

Default constructor.

```
public AuditScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

`scope` -

`prototype` -

Method detail :

```
public AuditEntry newEntry ()
```

Create an audit entry with 'system'-user identification.

JS call:

`Audit.newEntry()`

Returns :

an AuditEntry object having timestamp set to current time and user identification set to 'system'. This object is not yet persisted and is to be used with `Audit.logEntry(entry)`.

```
public void logEntry (NativeJavaObject entry)
```

Persist an AuditEntry object obtained using `Audit.newEntry()` and populated with targeted object and operation data (id, label, operation code, result and duration).

JS call:

`Audit.logEntry(entry)`

Parameters :

`entry` - the AuditEntry object to persist

Throws :

`SiatelException` - wrapped error

```
public void log (String resourceId, String resourceName, int operation,
int result, int duration)
```

Create an audit entry for given resource specifying operation code, result and duration.

JS call:

`Audit.log(resourceId, resourceName, operation, result, duration)`

Parameters :

`resourceId` - the id of the resource to be audited

`resourceName` - the name of the resource to be audited

`operation` - the code of the operation performed

`result` - the result of the operation

`duration` - the duration (in milliseconds) of the operation audited

Throws :

`SiatelException` - wrapped error

DESIGNSCRIPTING

```
public class DesignScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning design functionalities. The methods in this class are bound to the JavaScript `design` namespace object's methods and properties.

Author :

victorc

Constructor summary :

Constructor
DesignScripting ()
DesignScripting (Scriptable scope, Scriptable prototype)

Method summary :

Modifier and type	Method
public void	readAttribute (NativeJavaObject object, NativeJavaObject flags)
public void	readAttributeByLabel (NativeJavaObject object, NativeJavaObject flags, java.lang.String databaseId)
public void	enumerateAttributes (java.lang.String databaseId, NativeJavaObject list, NativeJavaObject flags)
public java.lang.String	idOfAttr (java.lang.String label, java.lang.String databaseId)
public java.lang.String	typeIdOfAttr (java.lang.String attrId)
public void	readForm (NativeJavaObject object, NativeJavaObject flags)
public void	readFormByLabel (NativeJavaObject object, NativeJavaObject flags, java.lang.String databaseId)
public void	enumerateForms (java.lang.String databaseId, NativeJavaObject list, NativeJavaObject flags)
public java.lang.String	idOfForm (java.lang.String label, java.lang.String databaseId)

public java.lang.String	idOfProcDef (java.lang.String label, java.lang.String databaseId)
public void	readType (NativeJavaObject object, NativeJavaObject flags)
public void	readTypeByLabel (NativeJavaObject object, NativeJavaObject flags, java.lang.String databaseId)
public void	enumerateTypes (java.lang.String databaseId, NativeJavaObject list, NativeJavaObject flags)
public java.lang.String	idOfType (java.lang.String label, java.lang.String databaseId)
public void	readGrid (NativeJavaObject object, NativeJavaObject flags)
public void	readGridByLabel (NativeJavaObject object, NativeJavaObject flags, java.lang.String databaseId)
public java.lang.String	idOfGrid (java.lang.String label, java.lang.String databaseId)
public void	readEntity (NativeJavaObject object, NativeJavaObject flags)
public void	readEntityByLabel (NativeJavaObject object, NativeJavaObject flags, java.lang.String databaseId)
public java.lang.String	idOfEntity (java.lang.String label, java.lang.String databaseId)
public java.lang.String	idOfSecurityProfile (java.lang.String label, java.lang.String databaseId)
public void	readSecurityProfile (NativeJavaObject object, NativeJavaObject flags)
public void	enumerateSecurityProfiles (java.lang.String databaseId, NativeJavaObject list, NativeJavaObject flags)
public java.lang.Object	getMailTemplateContent (java.lang.String mailTemplateId, java.lang.String language)
public java.lang.Object	getMailTemplateContentEx (java.lang.String mailTemplateId, java.lang.String language, java.lang.Boolean allowUnlocalizedContent)
public void	readDocumentTemplate (NativeJavaObject object,

	NativeJavaObject flags)
public void	readDocumentTemplateByLabel (NativeJavaObject object, NativeJavaObject flags, java.lang.String databaseId)
public void	enumerateDocumentTemplates (java.lang.String databaseId, NativeJavaObject list, NativeJavaObject flags)
public java.lang.String	idOfDocumentTemplate (java.lang.String label, java.lang.String databaseId)
public void	readMailTemplate (NativeJavaObject object, NativeJavaObject flags, java.lang.String language)
public void	readMailTemplateByLabel (NativeJavaObject object, NativeJavaObject flags, java.lang.String databaseId)
public long	fetchCounter (java.lang.String objectId, java.lang.String attributeId)
public java.lang.Object	getLocalizedLabel (java.lang.String objectId, java.lang.String language)
public void	read (NativeJavaObject object, NativeJavaObject flags)

Constructor detail :

```
public DesignScripting ()
```

Default constructor.

```
public DesignScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

scope -

prototype -

Method detail :

```
public void readAttribute (NativeJavaObject object, NativeJavaObject flags)
```

Attribute lazy-load.

JS call:

```
Design.readAttribute(object, flags)
```

Parameters :

`object` - wrapped Attribute object to initialize with data; the id field must be set

`flags` - wrapped Long value specifying the object fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public void readAttributeByLabel (NativeJavaObject object,
NativeJavaObject flags, String databaseId)
```

Attribute lazy-load using the object name as key.

JS call:

```
Design.readAttributeByLabel(object, flags, databaseId)
```

Parameters :

`object` - Attribute object to initialize; the label field must be set

`flags` - Long value specifying the object fields to initialize with data

`databaseId` - String value specifying the id of the database containing the attribute

Throws :

`SiatelException` - wrapped error

```
public void enumerateAttributes (String databaseId, NativeJavaObject
list, NativeJavaObject flags)
```

List Attribute objects contained in a database.

JS call:

```
Design.enumerateAttributes(databaseId, list, flags)
```

Parameters :

`databaseId` - String the id of database from which attributes are listed

`list` - List the list to fill with attributes

`flags` - Long value specifying the listed objects' fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public String idOfAttr (String label, String databaseId)
```

Get the id of an attribute based on it's name and containing database.

JS call:

```
Design.idOfAttr(name, databaseId)
```

Parameters :

`label` - String the name of the attribute

`databaseId` - String the id of the database

Returns :

the id of the attribute. `null` if attribute of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public String typeIdOfAttr (String attrId)
```

Get the type id of an attribute based on it's id.

JS call:

```
Design.typeIdOfAttr(attrId)
```

Parameters :

`attrId` - the id of the attribute

Returns :

the type id of the attribute. `null` if specified attribute does not exist

Throws :

`SiatelException` - wrapped error

```
public void readForm (NativeJavaObject object, NativeJavaObject flags)
```

Form lazy-load.

JS call:

```
Design.readForm(object, flags)
```

Parameters :

`object` - wrapped Form object to initialize with data; the id field must be set

`flags` - wrapped Long value specifying the object fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public void readFormByLabel (NativeJavaObject object, NativeJavaObject
flags, String databaseId)
```

Form lazy-load using the object name as key.

JS call:

```
Design.readFormByLabel(object, flags, databaseId)
```

Parameters :

`object` - Form object to initialize; the label field must be set

`flags` - Long value specifying the object fields to initialize with data

`databaseId` - String value specifying the id of the database containing the form

Throws :

`SiatelException` - wrapped error

```
public void enumerateForms (String databaseId, NativeJavaObject list,
NativeJavaObject flags)
```

List Form objects contained in a database.

JS call:

```
Design.enumerateForms(databaseId, list, flags)
```

Parameters :

`databaseId` - String the id of database from which forms are listed

`list` - List the list to fill with forms

`flags` - Long value specifying the listed objects' fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public String idOfForm (String label, String databaseId)
```

Get the id of a form based on it's name and containing database.

JS call:

```
Design.idOfForm(name, databaseId)
```

Parameters :

`label` - the name of the form

`databaseId` - the id of the database

Returns :

the id of the form. `null` if form of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public String idOfProcDef (String label, String databaseId)
```

Get the id of a process definition based on it's name and containing database.

JS call:

```
Design.idOfProcDef(name, databaseId)
```

Parameters :

`label` - the name of the process definition

`databaseId` - the id of the database

Returns :

the id of the process definition. `null` if process definition of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public void readType (NativeJavaObject object, NativeJavaObject flags)
```

Type lazy-load.

JS call:

```
Design.readType(object, flags)
```

Parameters :

`object` - wrapped Type object to initialize with data; the id field must be set

`flags` - wrapped Long value specifying the object fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public void readTypeByLabel (NativeJavaObject object, NativeJavaObject
flags, String databaseId)
```

Type lazy-load using the object name as key.

JS call:

```
Design.readTypeByLabel(object, flags, databaseId)
```

Parameters :

`object` - Type object to initialize; the label field must be set

`flags` - Long value specifying the object fields to initialize with data

`databaseId` - String value specifying the id of the database containing the type

Throws :

`SiatelException` - wrapped error

```
public void enumerateTypes (String databaseId, NativeJavaObject list,
NativeJavaObject flags)
```

List Type objects contained in a database.

JS call:

`Design.enumerateTypes(databaseId, list, flags)`

Parameters :

`databaseId` - String the id of database from which types are listed

`list` - List the list to fill with types

`flags` - Long value specifying the listed objects' fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public String idOfType (String label, String databaseId)
```

Get the id of a type based on it's name and containing database.

JS call:

`Design.idOfType(name, databaseId)`

Parameters :

`label` - the name of the type

`databaseId` - the id of the database

Throws :

`SiatelException` - wrapped error

```
public void readGrid (NativeJavaObject object, NativeJavaObject flags)
```

Grid lazy-load.

JS call:

```
Design.readGrid(object, flags)
```

Parameters :

`object` - wrapped Grid object to initialize with data; the id field must be set

`flags` - wrapped Long value specifying the object fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public void readGridByLabel (NativeJavaObject object, NativeJavaObject
flags, String databaseId)
```

Grid lazy-load using the object name as key.

JS call:

```
Design.readGridByLabel(object, flags, databaseId)
```

Parameters :

`object` - Grid object to initialize; the label field must be set

`flags` - Long value specifying the object fields to initialize with data

`databaseId` - String value specifying the id of the database containing the grid

Throws :

`SiatelException` - wrapped error

```
public String idOfGrid (String label, String databaseId)
```

Get the id of a grid based on it's name and containing database.

JS call:

```
Design.idOfGrid(name, databaseId)
```

Parameters :

`label` - the name of the grid

`databaseId` - the id of the database

Returns :

the id of the grid. `null` if grid of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public void readEntity (NativeJavaObject object, NativeJavaObject flags)
```

Entity lazy-load.

JS call:

```
Design.readEntity(object, flags)
```

Parameters :

`object` - wrapped Entity object to initialize with data; the id field must be set

`flags` - wrapped Long value specifying the object fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public void readEntityByLabel (NativeJavaObject object, NativeJavaObject
flags, String databaseId)
```

Entity lazy-load using the object name as key.

JS call:

```
Design.readEntityByLabel(object, flags, databaseId)
```

Parameters :

`object` - Entity object to initialize; the label field must be set

`flags` - Long value specifying the object fields to initialize with data

`databaseId` - String value specifying the id of the database containing the entity

Throws :

`SiatelException` - wrapped error

```
public String idOfEntity (String label, String databaseId)
```

Get the id of an entity based on it's name and containing database.

JS call:

```
Design.idOfEntity(label, databaseId)
```

Parameters :

`label` - the name of the entity

`databaseId` - the id of the database

Returns :

the id of the entity. `null` if entity of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public String idOfSecurityProfile (String label, String databaseId)
```

Get the id of a security profile based on it's name and containing database.

JS call:

```
Design.idOfSecurityProfile(name, databaseId)
```

Parameters :

`label` - the name of the security profile

`databaseId` - the id of the database

Returns :

the id of the security profile. `null` if security profile of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public void readSecurityProfile (NativeJavaObject object,
NativeJavaObject flags)
```

SecurityProfile lazy-load.

JS call:

```
Design.readSecurityProfile(object, flags)
```

Parameters :

`object` - wrapped SecurityProfile object to initialize with data; the id field must be set

`flags` - wrapped Long value specifying the object fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public void enumerateSecurityProfiles (String databaseId,
NativeJavaObject list, NativeJavaObject flags)
```

List SecurityProfile objects contained in a database.

JS call:

```
Design.enumerateSecurityProfiles(databaseId, list, flags)
```

Parameters :

`databaseId` - String the id of database from which security profiles are listed

`list` - List the list to fill with security profiles

`flags` - Long value specifying the listed objects' fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public Object getMailTemplateContent (String mailTemplateId, String
language)
```

MailTemplateContent load from MailTemplate id.

JS call:

`Design.getMailTemplateContent(mailTemplateId, language)`

Parameters :

`mailTemplateId` - id of MailTemplate object whose content is requested

`language` - the language in which to retrieve the mail template

Returns :

The mail template content

Throws :

`SiatelException` - wrapped error

```
public Object getMailTemplateContentEx (String mailTemplateId, String
language, Boolean allowUnlocalizedContent)
```

MailTemplateContent load from MailTemplate id.

JS call:

`Design.getMailTemplateContentEx(mailTemplateId, language, allowUnlocalizedContent)`

Parameters :

`mailTemplateId` - id of MailTemplate object whose content is requested

`language` - the language in which to retrieve the mail template

`allowUnlocalizedContent` - in case of unlocalized mail templates return same content for all languages. If not specified it is assumed `true`

Returns :

The mail template content

Throws :

`SiatelException` - wrapped error

```
public void readDocumentTemplate (NativeJavaObject object,
NativeJavaObject flags)
```

DocumentTemplate lazy-load.

JS call:

```
Design.readDocumentTemplate(object, flags)
```

Parameters :

`object` - wrapped DocumentTemplate object to initialize with data; the id field must be set

`flags` - wrapped Long value specifying the object fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public void readDocumentTemplateByLabel (NativeJavaObject object,
NativeJavaObject flags, String databaseId)
```

DocumentTemplate lazy-load using the object name as key.

JS call:

```
Design.readDocumentTemplateByLabel(object, flags, databaseId)
```

Parameters :

`object` - DocumentTemplate object to initialize; the label field must be set

`flags` - Long value specifying the object fields to initialize with data

`databaseId` - String value specifying the id of the database containing the document template

Throws :

`SiatelException` - wrapped error

```
public void enumerateDocumentTemplates (String databaseId,
NativeJavaObject list, NativeJavaObject flags)
```

List DocumentTemplate objects contained in a database.

JS call:

```
Design.enumerateDocumentTemplates(databaseId, list, flags)
```

Parameters :

`databaseId` - String the id of database from which document templates are listed

`list` - List the list to fill with document templates

`flags` - Long value specifying the listed objects' fields to initialize with data

Throws :

`SiatelException` - wrapped error

```
public String idOfDocumentTemplate (String label, String databaseId)
```

Get the id of a document template based on it's name and containing database.

JS call:

```
Design.idOfDocumentTemplate(name, databaseId)
```

Parameters :

`label` - String the name of the document template

`databaseId` - String the id of the database

Returns :

the id of the document template. `null` if document template of specified name does not exist

Throws :

`SiatelException` - wrapped error

```
public void readMailTemplate (NativeJavaObject object, NativeJavaObject
flags, String language)
```

MailTemplate lazy-load using the object id as key.

JS call:

```
Design.readMailTemplate(object, flags, language)
```

Parameters :

`object` - MailTemplate object to initialize; the id field must be set

`flags` - Long value specifying the object fields to initialize with data

`language` - String The language to retrieve

Throws :

`SiatelException` - wrapped error

```
public void readMailTemplateByLabel (NativeJavaObject object,
NativeJavaObject flags, String databaseId)
```

MailTemplate lazy-load using the object name as key.

JS call:

```
Design.readMailTemplateByLabel(object, flags, databaseId)
```

Parameters :

`object` - MailTemplate object to initialize; the label field must be set

`flags` - Long value specifying the object fields to initialize with data

`databaseId` - String value specifying the id of the database containing the document template

Throws :

`SiatelException` - wrapped error

```
public long fetchCounter (String objectId, String attributeId)
```

Get the next counter value for an object

JS call:

```
Design.fetchCounter(objectId, attributeId)
```

Parameters :

`objectId` - String The id of the object to fetch a new value from the counter

`attributeId` - String the id of the attribute to fetch the value from

Returns :

The fetched value from the counter. `-1` if an exception has occurred

```
public Object getLocalizedLabel (String objectId, String language)
```

Get the localized label for a specific object

JS call:

```
Design.getLocalizedLabel(objectId, language)
```

Parameters :

`objectId` - String The id of the object that has localized labels

`language` - String The language to retrieve

Returns :

The localized label

Throws :

`SiatelException` - wrapped error

```
public void read (NativeJavaObject object, NativeJavaObject flags)
```

Generic object lazy-load.

JS call:

```
Design.read(object, flags)
```

Parameters :

`object` - wrapped Generic object to initialize. Must have the `id` member set

`flags` - wrapped Long value specifying fields to initialize

Throws :

`SiatelException` - wrapped exception object as thrown by core layer

FLOWSCRIPTING

```
public class FlowScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning workflow engine functionalities. The methods in this class are bound to the JavaScript `Flow` namespace object's methods and properties.

Author :

victorc

Constructor summary :

Constructor
FlowScripting ()
FlowScripting (Scriptable scope, Scriptable prototype)

Method summary :

Modifier and type	Method
public void	createProcess (NativeJavaObject object)
public void	readProcess (NativeJavaObject object, NativeJavaObject flags)
public void	updateProcess (NativeJavaObject object, NativeJavaObject flags)
public void	deleteProcess (java.lang.String processId)
public void	enumerateProcesses (NativeJavaObject list, NativeJavaObject flags, java.lang.Object criteria)
public void	enumerateProcessesOffsetCount (NativeJavaObject list, NativeJavaObject flags, java.lang.Object criteria, int offset, int count)
public com.siatel.dto.DTO	createProcessRuntime (NativeJavaObject object)
public ProcessContent	readProcessContent (NativeJavaObject object)
public void	readTask (NativeJavaObject object, NativeJavaObject flags)
public void	readTaskLocalized (NativeJavaObject object, NativeJavaObject flags, NativeJavaObject locale)
public void	enumerateTasks (NativeJavaObject list, NativeJavaObject flags)

public void	enumerateTasksOffsetCount (NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public java.lang.String	idOfActiveTask (java.lang.String label, NativeJavaObject processRuntime)
public void	enumerateProcessRunningTasks (java.lang.String processId, NativeJavaObject list, NativeJavaObject flags)
public void	enumerateProcessRunningTasksOffsetCount (java.lang.String processId, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public void	enumerateProcessTasks (java.lang.String processId, NativeJavaObject list, NativeJavaObject flags)
public void	enumerateProcessTasksOffsetCount (java.lang.String processId, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public Event	newEvent (int code, java.lang.String processId, java.lang.String processName, java.lang.String taskId, java.lang.String taskName)
public void	postQueueEvent (NativeJavaObject event)
public java.lang.Long	synchronize (java.lang.String objectId, NativeJavaObject access)
public void	release (NativeJavaObject token)
public long	renderDelay (java.lang.String procDefId, NativeJavaObject delay)

Constructor detail :

```
public FlowScripting ()
```

Default constructor.

```
public FlowScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

scope -

prototype -

Method detail :

```
public void createProcess (NativeJavaObject object)
```

Create a Process object. The process created will be a 'draft' and not immediately active.

JS call:

```
Flow.createProcess(object)
```

Parameters :

`object` - wrapped Process object to create

Throws :

`SiatelException` - wrapped error

```
public void readProcess (NativeJavaObject object, NativeJavaObject
flags)
```

Process lazy-load.

JS call:

```
Flow.readProcess(object, flags)
```

Parameters :

`object` - wrapped Process object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Process fields to initialize

Throws :

`SiatelException` - wrapped error

```
public void updateProcess (NativeJavaObject object, NativeJavaObject
flags)
```

Process update.

JS call:

```
Flow.updateProcess(object, flags)
```

Parameters :

`object` - wrapped Process object to persist. Must have the `id` set

`flags` - wrapped Long value specifying Process fields to persist

Throws :

`SiatelException` - wrapped error

```
public void deleteProcess (String processId)
```

Process delete.

JS call:

`Flow.deleteProcess(processId)`

Parameters :

`processId` - id of process to delete

Throws :

`SiatelException` - wrapped error

```
public void enumerateProcesses (NativeJavaObject list, NativeJavaObject
flags, Object criteria)
```

Process enumeration.

JS call:

`Flow.enumerateProcesses(list, flags, criteria)`

Parameters :

`list` - wrapped List to fill by this method with criteria-matching processes

`flags` - wrapped Long value specifying Process fields to initialize for enumeration results

`criteria` - direct or wrapped List containing enumeration criteria. See ProcessPersistence#enumerate for supported criteria combinations

Throws :

`SiatelException` - wrapped error

```
public void enumerateProcessesOffsetCount (NativeJavaObject list,
NativeJavaObject flags, Object criteria, int offset, int count)
```

Process enumeration with offset and count.

JS call:

`Flow.enumerateProcessesOffsetCount(list, flags, criteria, offset, count)`

Parameters :

`list` - wrapped List to fill by this method with criteria-matching processes

`flags` - wrapped Long value specifying Process fields to initialize for enumeration results

`criteria` - direct or wrapped List containing enumeration criteria. See FlowLibrary#CMD_ENUMERATE_PROCESSES for supported criteria combinations

`offset` - enumeration start offset

`count` - maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public DTO createProcessRuntime (NativeJavaObject object)
```

Create a process' runtime data. This will instantiate a DTO.

JS call:

```
Flow.createProcessRuntime(object)
```

Parameters :

`object` - wrapped Process object

Returns :

the runtime data dto. This object is string-ized and stored as 'process runtime file'

Throws :

`SiatelException` - wrapped error

```
public ProcessContent readProcessContent (NativeJavaObject object)
```

ProcessContent lazy-load from file.

JS call:

```
Flow.readProcessContent(object)
```

Parameters :

`object` - wrapped Process object whose content is requested by this method (must have the `fileId` set) or String specifying the id of the stored file from which to load the content

Returns :

the initialized {@link ProcessContent} corresponding to the specified object of file

Throws :

`SiatelException` - wrapped error

```
public void readTask (NativeJavaObject object, NativeJavaObject flags)
```

Task lazy-load.

JS call:

```
Flow.readTask(object, flags)
```

Parameters :

`object` - wrapped Task object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Task fields to initialize

Throws :

`SiatelException` - wrapped error

```
public void readTaskLocalized (NativeJavaObject object, NativeJavaObject
flags, NativeJavaObject locale)
```

Task lazy-load.

JS call:

```
Flow.readTaskLocalized(object, flags, locale)
```

Parameters :

`object` - wrapped Task object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Task fields to initialize

`locale` - string value specifying language to retrieve

Throws :

`SiatelException` - wrapped error

```
public void enumerateTasks (NativeJavaObject list, NativeJavaObject
flags)
```

Task enumeration. All tasks in the system regardless of status are returned.

JS call:

```
Flow.enumerateTasks(list, flags)
```

Parameters :

`list` - wrapped List to fill by this method with task

`flags` - wrapped Long value specifying Task fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public void enumerateTasksOffsetCount (NativeJavaObject list,
NativeJavaObject flags, int offset, int count)
```

Task enumeration with offset and count. All tasks in the system regardless of status are returned.

JS call:

```
Flow.enumerateTasksOffsetCount(list, flags, offset, count)
```

Parameters :

`list` - wrapped List to fill by this method with task

`flags` - wrapped Long value specifying Task fields to initialize for enumeration results

`offset` - enumeration start offset

`count` - maximum number of objects to return

Throws :

`SiatelException` - wrapped error

```
public String idOfActiveTask (String label, NativeJavaObject
processRuntime)
```

Locate an active task by name in a given process' runtime data.

JS call:

`Flow.idOfActiveTask(label, runtime)`

Parameters :

`label` - name of the task whose id is requested

`processRuntime` - the runtime DTO-map of the process to be used for looking up active tasks

Returns :

the id of the task with specified name if any such active tasks exists in the process, `null` otherwise

Throws :

`SiatelException` - wrapped error

```
public void enumerateProcessRunningTasks (String processId,
NativeJavaObject list, NativeJavaObject flags)
```

Running Task objects enumeration. All active tasks for a specified process are returned.

JS call:

`Flow.enumerateProcessRunningTasks(processId, list, flags)`

Parameters :

`processId` - id of the process whose active tasks are requested

`list` - wrapped List to fill by this method with criteria-matching tasks

`flags` - wrapped Long value specifying Task fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public void enumerateProcessRunningTasksOffsetCount (String processId,
```

```
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

Running Task objects paged enumeration. Active tasks for a specified process are returned given a starting index offset and a count items to return.

JS call:

```
Flow.enumerateProcessRunningTasksOffsetCount(processId, list, flags,
offset, count)
```

Parameters :

`processId` - id of the process whose active tasks are requested

`list` - wrapped List to fill by this method with criteria-matching tasks

`flags` - wrapped Long value specifying Task fields to initialize for enumeration results

`offset` - the index offset

`count` - the count of items to return

Throws :

`SiatelException` - wrapped error

```
public void enumerateProcessTasks (String processId, NativeJavaObject
list, NativeJavaObject flags)
```

Task objects enumeration. All tasks for a specified process are returned, regardless of their status, sorted ascending by start timestamp (oldest first)

JS call:

```
Flow.enumerateProcessTasks(processId, list, flags)
```

Parameters :

`processId` - id of the process whose active tasks are requested

`list` - wrapped List to fill by this method with criteria-matching tasks

`flags` - wrapped Long value specifying Task fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public void enumerateProcessTasksOffsetCount (String processId,
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

Task objects paged enumeration. Tasks for a specified process are returned, regardless of their status, given a starting index offset and a count o items to return. Sort order is ascending by task start

timestamp (oldest first).

JS call:

```
Flow.enumerateProcessTasksOffsetCount(processId, list, flags, offset, count)
```

Parameters :

`processId` - id of the process whose active tasks are requested

`list` - wrapped List to fill by this method with criteria-matching tasks

`flags` - wrapped Long value specifying Task fields to initialize for enumeration results

`offset` - the index offset

`count` - the count of items to return

Throws :

`SiatelException` - wrapped error

```
public Event newEvent (int code, String processId, String processName, String taskId, String taskName)
```

Create a flow queue Event object. The created object is not posted in the flow event queue. The execution time (`timestamp`) for the event is the current system time so if posted the event will be executed immediately. The execution time can be modified by in the calling script before posting the event to the flow queue.

JS call:

```
Flow.newEvent(code, processId, processName, taskId, taskName)
```

Parameters :

`code` - the EventCode

`processId` - id of process

`processName` - name of process (optional)

`taskId` - id of task (depending on event type, can be `null`)

`taskName` - name of task (optional)

Returns :

the Event object

```
public void postQueueEvent (NativeJavaObject event)
```

Post an Event object to the flow queue.

JS call:

```
Flow.postQueueEvent(event)
```

Parameters :

`event` - the Event object

Throws :

`SiatelException` - wrapped error

```
public Long synchronize (String objectId, NativeJavaObject access)
```

Synchronize access to a flow object (process /and task). When synchronizing task access, access to containing process is also synchronized. 'READ' access is shared, 'WRITE' access is exclusive.

JS call:

```
var token = Flow.synchronize(objectId, access)
```

Parameters :

`objectId` - id of process or task

`access` - wrapped Integer specifying the requested access. One of Synch#READ or Synch#WRITE

Returns :

a lock token to be used when calling the pair- #release

Throws :

`SiatelException` - wrapped error

```
public void release (NativeJavaObject token)
```

Release a locked process /and task.

JS call:

```
Flow.release(token)
```

Parameters :

`token` - the lock token obtained from the pair- #synchronize

Throws :

`SiatelException` - wrapped error

```
public long renderDelay (String procDefId, NativeJavaObject delay)
```

Render a delay in milliseconds according to a process definition's planner.

JS call:

```
Flow.renderDelay(procDefId, delay)
```

Parameters :

`procDefId` - id of process definition that specifies the planner to use

`delay` - the Long millisecond-specified duration to be rendered according to a planner

Returns :

the rendered delay (in milliseconds)

Throws :

`SiatelException` - wrapped error

GedScripting

```
public class GedScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning document management functionalities. The methods in this class are bound to the JavaScript `Ged` namespace object's methods and properties.

Author :

victorc

Constructor summary :

Constructor
GedScripting ()
GedScripting (Scriptable scope, Scriptable prototype)

Method summary :

Modifier and type	Method
public void	createFolder (NativeJavaObject object)
public void	readFolder (NativeJavaObject object, NativeJavaObject flags)
public void	updateFolder (NativeJavaObject object, NativeJavaObject flags)
public void	deleteFolder (java.lang.String objectId)
public void	enumerateFolders (java.lang.String parentId, NativeJavaObject list, NativeJavaObject flags)
public void	enumerateFoldersOffsetCount (java.lang.String parentId, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public void	moveFolder (java.lang.String objectId, java.lang.String targetId)
public void	createDocument (NativeJavaObject object, java.lang.String parentId)
public void	readDocument (NativeJavaObject object, NativeJavaObject flags)
public void	updateDocument (NativeJavaObject object, NativeJavaObject flags)
public void	deleteDocument (java.lang.String objectId)

public void	enumerateDocuments (java.lang.String parentId, NativeJavaObject list, NativeJavaObject flags)
public void	enumerateDocumentsOffsetCount (java.lang.String parentId, NativeJavaObject list, NativeJavaObject flags, int offset, int count)
public void	createDocumentFulltext (java.lang.String objectId)
public void	recreateDocumentFulltext (java.lang.String objectId)
public void	deleteDocumentFulltext (java.lang.String objectId)
public void	addAttributeTextAreaFulltext (java.lang.String objectId, java.lang.String formId, java.lang.String attributeId, java.lang.String fileId)
public void	deleteAttributeTextAreaFulltext (java.lang.String objectId, java.lang.String formId, java.lang.String attributeId, java.lang.String fileId)
public java.lang.String	createNote (java.lang.String docId, java.lang.String text, java.lang.String ownerId)
public void	createNoteFulltext (java.lang.String objectId)
public void	recreateNoteFulltext (java.lang.String objectId)
public void	deleteNoteFulltext (java.lang.String objectId)
public void	addVersion (java.lang.String verType, java.lang.String documentId, java.lang.String newLabel, java.lang.String newFileId, java.lang.String verOwnerId)
public void	addVersionWithComment (java.lang.String verType, java.lang.String documentId, java.lang.String newLabel, java.lang.String newFileId, java.lang.String comment, java.lang.String verOwnerId)
public void	synchronizeOriginal (java.lang.String verType, NativeJavaObject idOrDoc, java.lang.String comment)
public void	unpinDocuments (NativeJavaObject newIds, NativeJavaObject oldIds, java.lang.String oldPinId)
public void	pinDocuments (NativeJavaObject object, java.lang.String parentId, NativeJavaObject list)
public java.lang.String	copyDocument (java.lang.String objectId, java.lang.String targetId, java.lnag.Boolean asCopy)

public void	replaceDocument (NativeJavaObject sourceObj, NativeJavaObject targetObj)
public void	synchronizeAttributes (java.lang.String synchType, NativeJavaObject srcProc, NativeJavaObject targetProc)
public void	makeFiche (NativeJavaObject object, java.lang.String pinId)
public void	copyCustomSecurity (java.lang.String sourceObjectId, java.lang.String targetObjectId)
public void	copyNotes (java.lang.String sourceObjectId, java.lang.String targetObjectId)
public void	copyDocumentComments (java.lang.String sourceObjectId, java.lang.String targetObjectId)
public void	copyAttributes (NativeJavaObject target, NativeJavaObject source)
public void	synchronizePin (NativeJavaObject target, NativeJavaObject source)
public java.lang.Object	renderAttribute (java.lang.String attributeId, java.lang.String formId, java.lang.String formPage, NativeJavaObject attributes)
public void	renderAttributes (NativeJavaObject map, java.lang.String formId, java.lang.String formPage, NativeJavaObject attributes)
public java.lang.String	getCurrencyAttributeValue (java.lang.String attributeId, NativeJavaObject attributeValue, java.lang.Integer modifier)
public java.lang.String	getTextareaAttributeValue (NativeJavaObject attributeValue)
public java.lang.String	replaceTextareaAttributeValue (java.lang.String textareaFileId, NativeJavaObject attributeValue)
public java.lang.String	getFloatAttributeValue (java.lang.String attributeValue)
public java.lang.String	getTypeLocalizedValue (java.lang.String typeId, NativeJavaObject value, java.lang.String language)
public void	updatePinSize (java.lang.String pinId)
public void	setAttributeValue (NativeJavaObject object, java.lang.String attributeId, NativeJavaObject attributeValue)

public void	createLink (java.lang.String strDest, java.lang.String strSrc, boolean bidirectional, java.lang.String label)
public void	autoPositionFolderInParent (java.lang.String folderId, java.lang.String ascDesc, java.lang.String attributeId)
public void	setSecurity (java.lang.String objectId, NativeJavaObject juListAccessMasks)
public void	repositionFolder (java.lang.String folderId, java.lang.String afterId)
public void	repositionDocument (java.lang.String documentId, java.lang.String afterId)
public void	repositionObject (java.lang.String objectId, java.lang.String afterId)
public void	enumerateNotes (java.lang.String objectId, NativeJavaObject list)
public void	deleteLink (NativeJavaObject ntvDTOCriteria)
public void	updateLink (NativeJavaObject ntvLink, NativeJavaObject ntvLongFlags)
public void	enumerateLinks (NativeJavaObject ntvDTOCriteria, NativeJavaObject ntvList)
public void	broadcastDocuments (java.lang.Object ntStartDate, java.lang.Object nDueTime, java.lang.Object nExpireTime, java.lang.Object sActorIdOrName, java.lang.Object ntIdsOrObjects, java.lang.Boolean bNotify)
public java.lang.String	createWorkspace (java.lang.Object workspaceFolder, java.lang.Object expireDate, java.lang.Object managerAccessIds, java.lang.Object writeAccessIds, java.lang.Object readAccessIds)
public java.lang.Boolean	lock (java.lang.String docId, java.lang.Integer lockType, java.lang.String userId, java.lang.Object validity, java.lang.Boolean steal)
public void	unlock (java.lang.String docId)
public void	moveToTrash (java.lang.String objectId)
public void	archiveFolder (java.lang.Object maybeFolder)
public void	unarchiveFolder (java.lang.Object maybeFolder, java.lang.Boolean recurse)

public java.lang.String	copyPin (java.lang.String objectId, java.lang.String targetId, java.lang.Object maybeAsCopy)
public java.lang.Boolean	getFTMode (java.lang.String objectId)
public void	evaluateFormSecurity (java.lang.String objectId)
public NativeArray	getDocRefs (java.lang.Object docOrId)
public void	readSigningCycle (NativeJavaObject object, NativeJavaObject flags)
public java.lang.Object	findDocByLabel (java.lang.String parentId, java.lang.String label, NativeJavaObject flags)
public java.lang.Object	findFolderByLabel (java.lang.String parentId, java.lang.String label, NativeJavaObject flags)

Constructor detail :

```
public GedScripting ()
```

Default constructor.

```
public GedScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

-

prototype -

Method detail :

```
public void createFolder (NativeJavaObject object)
```

Create a Folder object. Object must have at least the `label` and `parentId` property initialized.

JS call:

```
Ged.createFolder(object)
```

Parameters :

`object` - wrapped Folder object to create

Throws :

`SiatelException` - wrapped error

```
public void readFolder (NativeJavaObject object, NativeJavaObject flags)
```

Folder lazy-load.

JS call:

```
Ged.readFolder(object, flags)
```

Parameters :

`object` - wrapped Folder object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Folder fields to initialize

Throws :

`SiatelException` - wrapped error

```
public void updateFolder (NativeJavaObject object, NativeJavaObject
flags)
```

Folder update.

JS call:

```
Ged.updateFolder(object, flags)
```

Parameters :

`object` - wrapped Folder object to update. Must have the `id` set

`flags` - wrapped Long value specifying Folder fields to persist

Throws :

`SiatelException` - wrapped error

```
public void deleteFolder (String objectId)
```

Delete Folder by id.

JS call:

```
Ged.deleteFolder(objectId)
```

Parameters :

`objectId` - id of folder to delete

Throws :

`SiatelException` - wrapped error

```
public void enumerateFolders (String parentId, NativeJavaObject list,
NativeJavaObject flags)
```

Folder enumeration. All direct children of a folder are returned.

JS call:

```
Ged.enumerateFolders(parentId, list, flags)
```

Parameters :

`parentId` - id of parent folder

`list` - wrapped List to fill by this method with database objects

`flags` - wrapped Long value specifying Folder fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public void enumerateFoldersOffsetCount (String parentId,
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

Folder enumeration with offset and count.

JS call:

```
Ged.enumerateFoldersOffsetCount(parentId, list, flags, offset, count)
```

Parameters :

`parentId` - id of parent folder

`list` - wrapped List to fill by this method with database objects

`flags` - wrapped Long value specifying Folder fields to initialize for enumeration results

`offset` - enumeration offset

`count` - enumeration count

Throws :

`SiatelException` - wrapped error

```
public void moveFolder (String objectId, String targetId)
```

Move a folder from current location (parent) to another location (parent) in same database.

JS call:

```
Ged.moveFolder(objectId, targetId)
```

Parameters :

`objectId` - id of folder to move

`targetId` - id of new parent folder, where the folder will be moved

Throws :

`SiatelException` - wrapped error

```
public void createDocument (NativeJavaObject object, String parentId)
```

Create a Document object. Object must have at least the `label` property initialized

JS call:

```
Ged.createDocument(object, parentId)
```

Parameters :

`object` - wrapped Document object to create

`parentId` - id of container where the document will be created. A folder, process or pin-document id

Throws :

`SiatelException` - wrapped error

```
public void readDocument (NativeJavaObject object, NativeJavaObject flags)
```

Document lazy-load. Applies also to 'pin' documents.

JS call:

```
Ged.readDocument(object, flags)
```

Parameters :

`object` - wrapped Document object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying Document fields to initialize

Throws :

`SiatelException` - wrapped error

```
public void updateDocument (NativeJavaObject object, NativeJavaObject flags)
```

Document update.

JS call:

```
Ged.updateDocument(object, flags)
```

Parameters :

`object` - wrapped Document object to update. Must have the `id` set

`flags` - wrapped Long value specifying Document fields to persist

Throws :

`SiatelException` - wrapped error

```
public void deleteDocument (String objectId)
```

Delete Document by id. Applies to pin documents as well.

JS call:

```
Ged.deleteDocument(objectId)
```

Parameters :

`objectId` - id of document or pin to delete

Throws :

`SiatelException` - wrapped error

```
public void enumerateDocuments (String parentId, NativeJavaObject list,
NativeJavaObject flags)
```

Document enumeration. All documents in a folder, process or pin are listed.

JS call:

```
Ged.enumerateDocuments(parentId, list, flags)
```

Parameters :

`parentId` - id of parent folder, process or pin

`list` - wrapped List to fill by this method with database objects

`flags` - wrapped Long value specifying Document fields to initialize for enumeration results

Throws :

`SiatelException` - wrapped error

```
public void enumerateDocumentsOffsetCount (String parentId,
NativeJavaObject list, NativeJavaObject flags, int offset, int count)
```

Document enumeration with offset and count.

JS call:

```
Ged.enumerateDocumentsOffsetCount(parentId, list, flags, offset,
count)
```

Parameters :

`parentId` - id of parent folder, process or pin

`list` - wrapped List to fill by this method with database objects

`flags` - wrapped Long value specifying Document fields to initialize for enumeration results

`offset` - enumeration offset

`count` - enumeration count

Throws :

`SiatelException` - wrapped error

```
public void createDocumentFulltext (String objectId)
```

Index a text document's content.

JS call:

```
Ged.createDocumentFulltext(objectId)
```

Parameters :

`objectId` - id of document whose content to index

Throws :

`SiatelException` - wrapped error

```
public void recreateDocumentFulltext (String objectId)
```

Re-index a text document's content. The content index is first removed and then created again.

JS call:

```
Ged.recreateDocumentFulltext(objectId)
```

Parameters :

`objectId` - id of document whose content to reindex

Throws :

`SiatelException` - wrapped error

```
public void deleteDocumentFulltext (String objectId)
```

Delete the content index a text document.

JS call:

```
Ged.deleteDocumentFulltext(objectId)
```

Parameters :

`objectId` - id of document whose content index to delete

Throws :

`SiatelException` - wrapped error

```
public void addAttributeTextAreaFulltext (String objectId, String
formId, String attributeId, String fileId)
```

Index a text area attribute's content for a given object.

JS call:

```
Ged.addAttributeTextAreaFulltext(objectId, formId, attributeId,
fileId)
```

Parameters :

`objectId` - id of document/pin/folder/process having the text area attribute

`formId` - id of form that specifies the text area attribute containment for object

`attributeId` - id of the text area attribute whose content to index

`fileId` - id of file with the content of the text area attribute

Throws :

`SiatelException` - wrapped error

```
public void deleteAttributeTextAreaFulltext (String objectId, String
formId, String attributeId, String fileId)
```

Delete the content index of a text area attribute for a given object.

JS call:

```
Ged.deleteAttributeTextAreaFulltext(objectId, formId, attributeId,
fileId)
```

Parameters :

`objectId` - id of document/pin/folder/process having the text area attribute

`formId` - id of form that specifies the text area attribute containment for object

`attributeId` - id of the text area attribute whose content index to delete

`fileId` - id of file with the content of the text area attribute

Throws :

`SiatelException` - wrapped error

```
public String createNote (String docId, String text, String ownerId)
```

Create a note on a document.

JS call:

```
Ged.createNote(docId, text, ownerId)
```

Parameters :

`docId` - id of document on which to create post-it note

`text` - text of note

`ownerId` - id of user who will own the note. This parameter is optional and will fallback to the owner of the parent document

Returns :

id of newly created note

Throws :

`SiatelException` - wrapped error

```
public void createNoteFulltext (String objectId)
```

Index a note's content.

JS call:

```
Ged.createNoteFulltext(objectId)
```

Parameters :

`objectId` - id of note object

Throws :

`SiatelException` - wrapped error

```
public void recreateNoteFulltext (String objectId)
```

Re-index a note's content. The content index is first removed and then created again.

JS call:

```
Ged.recreateNoteFulltext(objectId)
```

Parameters :

`objectId` - id of note object

Throws :

`SiatelException` - wrapped error

```
public void deleteNoteFulltext (String objectId)
```

Delete the content index of a note.

JS call:

```
Ged.deleteNoteFulltext(objectId)
```

Parameters :

`objectId` - id of note object

Throws :

`SiatelException` - wrapped error

```
public void addVersion (String verType, String documentId, String
newLabel, String newFileId, String verOwnerId)
```

Create a version for a document.

JS call:

```
Ged.addVersion(verType, documentId, newLabel, newFileId[, verOwnerId])
```

Parameters :

`verType` - the version type. One of `"major"`, `"minor"` or `"replace"` strings. Usign `"replace"` will not actually create a new version but the document content and label are replaced and the document size is refreshed

`documentId` - id of document to which a version is created

`newLabel` - the new label of the document

`newFileId` - id of the file that will replace the current content of the document

`verOwnerId` - id of the user who takes ownership of the created version

Throws :

`SiatelException` - wrapped error

```
public void addVersionWithComment (String verType, String documentId,
String newLabel, String newFileId, String comment, String verOwnerId)
```

Create a version for a document adding a comment in the document version history.

JS call:

```
Ged.addVersionWithComment(verType, documentId, newLabel, newFileId,
comment[, verOwnerId])
```

Parameters :

`verType` - the version type. One of `"major"`, `"minor"` or `"replace"` strings. Usign `"replace"` will not actually create a new version but the document content and label are replaced and the document size is refreshed

`documentId` - id of document to which a version is created

`newLabel` - the new label of the document

`newFileId` - id of the file that will replace the current content of the document

`comment` - the comment to add to document version history

`verOwnerId` - id of the user who takes ownership of the created version

Throws :

`SiatelException` - wrapped error

```
public void synchronizeOriginal (String verType, NativeJavaObject
idOrDoc, String comment)
```

Apply the changes of a "copy" document to the original, with optional versioning and comment in version history.

JS call:

```
Ged.synchronizeOriginal(verType, idOrDoc, comment)
```

Parameters :

`verType` - the version type. One of `"major"`, `"minor"` or `"replace"` strings. Usign `"replace"` will not actually create a new version but the original document's content is replaced and the document size is refreshed

`idOrDoc` - id of "copy" document or the "copy"- Document itself used as content synchornization source

`comment` - the comment to be used if versioning

Throws :

`SiatelException` - wrapped error

```
public void unpinDocuments (NativeJavaObject newIds, NativeJavaObject
oldIds, String oldPinId)
```

Extract documents from a pin document.

JS call:

```
Ged.unpinDocuments(newIds, oldIds, oldPinId)
```

Parameters :

`newIds` - a List to be filled with the new document ids, including the id of the resulting index card (fiche) if this is the case

`oldIds` - a List containing the ids of documents to be extracted from the pin. If all components are specified the `newIds` list will contain the id of the resulting index card (fiche) upon the end of this call

`oldPinId` - the id of the pin from which documents are extracted

Throws :

`SiatelException` - wrapped error

```
public void pinDocuments (NativeJavaObject object, String parentId,
NativeJavaObject list)
```

Pin a group of documents from the same database.

JS call:

```
Ged.pinDocuments(object, parentId, list)
```

Parameters :

`object` - the pin Document object to create; must have `label` and `formId` properties set

`parentId` - the id of the folder or process where the new pin will be created

`list` - a List or native javascript array containing the Document objects to be pinned or their ids

Throws :

`SiatelException` - wrapped error

```
public String copyDocument (String objectId, String targetId, Boolean
asCopy)
```

Copy a document, keeping a reference to the source document.

JS call:

```
Ged.copyDocument(objectId, targetId[, asCopy])
```

Parameters :

`objectId` - the id of the source document to copy

`targetId` - the id of the folder or process where the copy-document will be created

`asCopy` - indicates if the new created copy is linked to the source document or not

Returns :

the id of the new document

Throws :

`SiatelException` - wrapped error

```
public void replaceDocument (NativeJavaObject sourceObj,
NativeJavaObject targetObj)
```

Replace an existing document with another existing document.

JS call:

```
Ged.replaceDocument(sourceObj, targetObj)
```

Parameters :

`sourceObj` - the source Document of the replacement operation

`targetObj` - the destination Document of the replacement operation

Throws :

`SiatelException` - wrapped error

```
public void synchronizeAttributes (String synchType, NativeJavaObject
srcProc, NativeJavaObject targetProc)
```

Copy the attributes from one process to another. Useful when a process starts and waits for another process and they share a set of attributes.

JS call:

```
Ged.synchronizeAttributes(synchType, srcProc, targetProc)
```

Parameters :

`synchType` - no longer used

`srcProc` - the source Process of the attribute copy operation

`targetProc` - the destination Process of the attribute copy opration

Throws :

`SiatelException` - wrapped error

```
public void makeFiche (NativeJavaObject object, String pinId)
```

Convert a pin-document to an index card (fiche). Useful when programatically extracting all components of the pin leaving it empty and it needs to be converted.

JS call:

```
Ged.makeFiche(object, pinId)
```

Parameters :

`object` - the Document index card object to be initialized with the resulting id

`pinId` - the id of the pin to convert

Throws :

`SiatelException` - wrapped error

```
public void copyCustomSecurity (String sourceObjectId, String
targetObjectId)
```

Apply the security descriptor of an object (folder/process/pin/document) to another object (folder/process/pin/document).

JS call:

```
Ged.copyCustomSecurity(sourceObjectId, targetObjectId)
```

Parameters :

`sourceObjectId` - the object to use as source for the security-copy operation

`targetObjectId` - the object used as destination for the security-copy operation

Throws :

`SiatelException` - wrapped error

```
public void copyNotes (String sourceObjectId, String targetObjectId)
```

Copy the notes of a document to another document.

JS call:

```
Ged.copyNotes(sourceObjectId, targetObjectId)
```

Parameters :

`sourceObjectId` - the object to use as source for the note-copy operation

`targetObjectId` - the object used as destination for the note-copy operation

Throws :

`SiatelException` - wrapped error

```
public void copyDocumentComments (String sourceObjectId, String
targetObjectId)
```

Copy the comments of a document to another document.

JS call:

```
Ged.copyDocumentComments(sourceObjectId, targetObjectId)
```

Parameters :

`sourceObjectId` - the object to use as source for the comments-copy operation

`targetObjectId` - the object used as destination for the comments-copy operation

Throws :

`SiatelException` - wrapped error

```
public void copyAttributes (NativeJavaObject target, NativeJavaObject
source)
```

Fill the attributes' list of an object using another object as source. This will not store the destination object's attributes.

JS call:

```
Ged.copyAttributes(target, source)
```

Parameters :

`target` - the copy-destination object

`source` - the copy-source object

Throws :

`SiatelException` - wrapped error

```
public void synchronizePin (NativeJavaObject target, NativeJavaObject source)
```

Copy the attributes and security of one pin document to another pin document.

JS call:

```
Ged.synchronizePin(target, source)
```

Parameters :

`target` - the copy-destination object

`source` - the copy-source object

Throws :

`SiatelException` - wrapped error

```
public Object renderAttribute (String attributeId, String formId, String formPage, NativeJavaObject attributes)
```

Retrieve the value of the given attribute for a specified form page and set of attributes by evaluating its associated rule if any and all rules that must be evaluated before it.

JS call:

```
Ged.renderAttribute(attributeId, formId, formPage, attributes)
```

Parameters :

`attributeId` - the id of the attribute whose value must be rendered

`formId` - the id of the form

`formPage` - the current form page

`attributes` - the current list of attributes

Returns :

the new value of the attribute after rule evaluation or null if could not be determined

Throws :

`SiatelException` - wrapped error

```
public void renderAttributes (NativeJavaObject map, String formId,
String formPage, NativeJavaObject attributes)
```

Evaluate rules for given form page and return a map containing all modified attributes and their corresponding values.

JS call:

```
Ged.renderAttributes(attributeId, formId, formPage, attributes)
```

Parameters :

`map` - wrapped Map to fill by this method with (attributeId, newValue) pairs

`formId` - the id of the form

`formPage` - the current page form

`attributes` - the current list of attributes

Throws :

`SiatelException` - wrapped error

```
public String getCurrencyAttributeValue (String attributeId,
NativeJavaObject attributeValue, Integer modifier)
```

Render the value of a currency attribute as string.

JS call:

```
Ged.getCurrencyAttributeValue(attributeId, attributeValue, modifier)
```

Parameters :

`attributeId` - id of attribute to render. Will be used to extract the currency name from attribute definition

`attributeValue` - the value of the attribute

`modifier` - `Modifier.CURRENCY_VALUE` for just the numeric value, `Modifier.CURRENCY_NAME` for just the currency name, `null` for the whole string with name and numeric

Returns :

the rendered currency string

Throws :

`SiatelException` - wrapped error

```
public String getTextareaAttributeValue (NativeJavaObject
attributeValue)
```

Retreieve the text in a text-area attribute's file id value.

JS call:

`Ged.getTextareaAttributeValue(attributeValue)`

Parameters :

`attributeValue` - the file id value of the attribute

Returns :

the string represented text

Throws :

`SiatelException` - wrapped error

```
public String replaceTextareaAttributeValue (String textareaFileId,
NativeJavaObject attributeValue)
```

Replace the content of a text-area attribute's file with the specified string

JS call:

`Ged.replaceTextareaAttributeValue(textareaFileId, attributeValue)`

Parameters :

`textareaFileId` - the file id value of the attribute

`attributeValue` - the text content to replace with

Returns :

a new file id if storage has changed

Throws :

`SiatelException` - wrapped error

```
public String getFloatAttributeValue (String attributeValue)
```

Re-format a string-represented floating point numeric to form-valid string for a numeric decimal attribute.

JS call:

`Ged.getFloatAttributeValue(attributeValue)`

Parameters :

`attributeValue` - the string to re-format

Returns :

the formated string

Throws :

`SiatelException` - wrapped error

```
public String getTypeLocalizedValue (String typeId, NativeJavaObject
value, String language)
```

Retrieves the translation of the given value from the localized type content.

JS call:

```
var localizedValue = Ged.getTypeLocalizedValue(typeId, value, lang)
```

Parameters :

`typeId` - The id of the type

`value` - The value to translate

`language` - The language in which to translate the value

Returns :

The translation of the value if one exists otherwise the value itself

Throws :

`SiatelException` - wrapped error

```
public void updatePinSize (String pinId)
```

Recomputes the size of a pin.

JS call:

```
Ged.updatePinSize(pinId)
```

Parameters :

`pinId` - the id of the pin

```
public void setAttributeValue (NativeJavaObject object, String
attributeId, NativeJavaObject attributeValue)
```

Set the value of an attribute for a Viewable object enforcing conversion to attribute native type.

JS call:

```
Ged.setAttributeValue(object, attributeId, attributeValue)
```

Parameters :

`object` - the scripting-wrapped Viewable implementing object (Folder, Process or Document)

`attributeId` - `String` representing the id of the attribute whose value to set

`attributeValue` - the scripting-wrapped value for the attribute. Most of the time a `java.lang.String`.

```
public void createLink (String strDest, String strSrc, boolean
bidirectional, String label)
```

Create a link between two documents.

JS call:

```
Ged.createLink(strDest, strSrc, bidirectional[, label])
```

Parameters :

`strDest` - id of link destination

`strSrc` - id of link source

`bidirectional` - `true` if link should be bidirectional, as in both objects are source and destination for each other

`label` - a name for the link. Optional name for the link

```
public void autoPositionFolderInParent (String folderId, String ascDesc,
String attributeId)
```

Analyze and update a folder's position within it's parent based on a sort order and a sort attribute to be used as criteria. Useful when generating classification structures.

JS call:

```
Ged.autoPositionFolderInParent(folderId, ascDesc, attributeId)
```

Parameters :

`folderId` - id of the folder to re-position.

`ascDesc` - `"asc"` or `"desc"`. Any other value will be ignored and will have no effect.

`attributeId` - the id of the attribute to use as criteria. System scalar attributes (label, dates) can also be used.

```
public void setSecurity (String objectId, NativeJavaObject
juListAccessMasks)
```

Set the security access rights for a given object. The new security will overwrite any existing security,

making the newly specified security the only one applicable onwards.

JS call:

```
Ged.setSecurity(objectId, juListAccessMasks)
```

Parameters :

`objectId` - id of object whose security access rights to change.

`juListAccessMasks` - scripting-wraped `java.util.List` of com.siatel.domain.security.AccessMask to apply to the object.

```
public void repositionFolder (String folderId, String afterId)
```

Reposition a folder within its parent; does not move the folder to another parent

JS call:

```
Ged.repositionFolder(folderId, afterId)
```

Parameters :

`folderId` - id of the folder to re-position.

`afterId` - the id of the folder that this folder is to be inserted after

```
public void repositionDocument (String documentId, String afterId)
```

Reposition a document within its parent; does not move the document to another parent

JS call:

```
Ged.repositionDocument(documentId, afterId)
```

Parameters :

`documentId` - id of the document to re-position.

`afterId` - the id of the document that this document is to be inserted after

```
public void repositionObject (String objectId, String afterId)
```

Reposition a folder or a document within its parent; does not move the document or folder to another parent

JS call:

```
Ged.repositionObject(objectId, afterId)
```

Parameters :

`objectId` - id of the folder or document to re-position.

`afterId` - the id of the folder or document that this folder or document is to be inserted after

```
public void enumerateNotes (String objectId, NativeJavaObject list)
```

Enumerate the notes of an object.

JS call:

```
Ged.enumerateNotes(objectId)
```

Parameters :

`objectId` - the object to use as source for the note-listing operation

`list` - the list to fill with Note objects

Throws :

`SiatelException` - wrapped error

```
public void deleteLink (NativeJavaObject ntvDTOCriteria)
```

Delete object links based on criteria.

JS call:

```
Ged.deleteLink(ntvDTOCriteria)
```

Parameters :

`ntvDTOCriteria` - the criteria to use for delete. See com.siatel.framework.service.ged.GedLibrary#CMD_DELETE_OBJECT_LINK for parameters

Throws :

`SiatelException` - wrapped error

```
public void updateLink (NativeJavaObject ntvLink, NativeJavaObject
ntvLongFlags)
```

Update on object link's properties

JS call:

```
Ged.updateLink(strNodeId)
```

Parameters :

`ntvLink` - the ObjectLink instance to update

`ntvLongFlags` - the Long bit encoded flags for the link properties to update. If `null` or not set then `Flag.OBJECT_LINK_ALL` is assumed

Throws :

`SiatelException` - wrapped error

```
public void enumerateLinks (NativeJavaObject ntvDTOCriteria,
```

```
NativeJavaObject ntvList)
```

Enumerate object links.

JS call:

```
Ged.enumerateLinks(ntvDTOCriteria, ntvList)
```

Parameters :

`ntvDTOCriteria` - the enumeration criteria. See com.siatel.framework.service.ged.GedLibrary#CMD_ENUMERATE_OBJECT_LINKS for parameters

`ntvList` - the list to fill with results

Throws :

`SiatelException` - wrapped error

```
public void broadcastDocuments (Object ntStartDate, Object nDueTime,
Object nExpireTime, Object sActorIdOrName, Object ntIdsOrObjects,
Boolean bNotify)
```

Broadcast a document or set of documents to one or more users.

JS call:

```
Ged.broadcastDocuments(ntStartDate, nDueTime, nExpireTime,
sActorIdOrName, ntIdsOrObjects, bNotify)
```

Parameters :

`ntStartDate` - the start date for the broadcast; must be a date object or an integer; if an integer is provided, the start date will be calculated as current time plus the number of minutes provided

`nDueTime` - expected 'read' time for the document; an absolute date can be provided or a number of minutes

`nExpireTime` - expire time for the read operation; an absolute date can be provided or a number of minutes

`sActorIdOrName` - an actor or list of actors to broadcast to; an actor can be either a group, a role or a specific user; the function call accepts either a single entity or a java.util.List of actors

`ntIdsOrObjects` - a document or list of documents to broadcast; the function call accepts either a single document or a java.util.List of documents

`bNotify` - boolean indicating if actors are to be notified

Throws :

`SiatelException` - wrapped error

```
public String createWorkspace (Object ntStartDate, Object nDueTime,
Object nExpireTime, Object sActorIdOrName, Object ntIdsOrObjects,
Boolean bNotify)
```

Create a workspace.

JS call:

```
Ged.createWorkspace(workspaceFolder, expireDate, managerAccessIds,
writeAccessIds, readAccessIds)
```

Parameters :

`workspaceFolder` - the workspace folder. Must have `parentId` set to an object that indicates the repo that it will contain it; internally this id will be coerced to the workspace root id

`expireDate` - the expiration `java.util.Date`. Can also be a string of one of the following formats: dd/MM/yyyy, dd.MM.yyyy, dd-MM-yyyy, yyyy/MM/dd, yyyy.MM.dd, yyyy-MM-dd, ISO8601

`managerAccessIds` - the list of user account ids assigned with 'manager' access to the workspace

`writeAccessIds` - the list of user account ids assigned with 'write' access to the workspace

`readAccessIds` - the list of user account ids assigned with 'read' access to the workspace

Returns :

the id of the workspace

Throws :

`SiatelException` - wrapped error

```
public Boolean lock (String docId, Integer lockType, String userId,
Object validity, Boolean steal)
```

Lock a document.

JS call:

```
var success = Ged.lock(docId, lockType, userId, validity, steal)
```

Parameters :

`docId` - the document id

`lockType` - the lock type (0 = LOCK, 1 = CHECK_OUT)

`userId` - the user under whose identity the lock is set

`validity` - the validity of the lock, can be a `java.util.Date` in which case is absolute, a

number in which case is the duration in millis or `null` in which case the lock is permanent

`steal` - `true` if any existing lock should be overridden

Returns :

`true` if lock set successfully, `false` otherwise (e.g. document is already locked and 'steal' was not set)

Throws :

`SiatelException` - wrapped error

```
public void unlock (String docId)
```

Unlock a document.

JS call:

```
Ged.unlock(docId)
```

Parameters :

`docId` - the document id

Throws :

`SiatelException` - wrapped error

```
public void moveToTrash (String objectId)
```

Move a 'trash'-able object (folder, document, version) to trash.

JS call:

```
Ged.moveToTrash(objectId)
```

Parameters :

`objectId` - the id of object to move to trash

Throws :

`SiatelException` - wrapped error

```
public void archiveFolder (Object maybeFolder)
```

Move folder from active to archive database.

JS call:

```
Ged.archiveFolder(folderId)
```

Parameters :

`maybeFolder` - the folder to archive or its id

Throws :

`SiatelException` - wrapped error

```
public void unarchiveFolder (Object maybeFolder, Boolean recurse)
```

Move folder from archive to active database.

JS call:

```
Ged.unarchiveFolder(folderId, true)
```

Parameters :

`maybeFolder` - the folder to retrieve from archive or its id

`recurse` - indicates if the operation should be recursive or not

Throws :

`SiatelException` - wrapped error

```
public String copyPin (String objectId, String targetId, Object
maybeAsCopy)
```

Copy a pin document and all it's components, optionally keeping a reference to the source document/s.

JS call:

```
Ged.copyPin(objectId, targetId[, asCopy])
```

Parameters :

`objectId` - the id of the source document to copy

`targetId` - the id of the folder or process where the copy-document will be created

`maybeAsCopy` - optional parameter that can evaluate to a `boolean` specifying weather references to the source should be kept

Returns :

the id of the new pin

Throws :

`SiatelException` - wrapped error

```
public Boolean getFTMode (String objectId)
```

Get the full-text mode of a database given the id of a database-contained object.

JS call:

```
Ged.getFTMode(objectId)
```

Parameters :

`objectId` - id of a database-contained object

Returns :

`true` if database containing object specified by id if full-text enabled, `false` otherwise or in case of error

Throws :

`SiatelException` - wrapped error

```
public void evaluateFormSecurity (String objectId)
```

Force a form-based security re-evaluation.

JS call:

```
Ged.evaluateFormSecurity(objectId)
```

Parameters :

`objectId` - the id of object to re-evaluate

Throws :

`SiatelException` - wrapped error

```
public NativeArray getDocRefs (Object docOrId)
```

Get the references of a document.

JS call:

```
var references = Ged.getDocRefs(documentId)
```

Parameters :

`docOrId` - the document or its id

Returns :

an array of Document objects

Throws :

`SiatelException` - wrapped error

```
public void readSigningCycle (NativeJavaObject object, NativeJavaObject
flags)
```

SigningCycle lazy-load.

JS call:

```
Ged.readSigningCycle(object, flags)
```

Parameters :

`object` - wrapped SigningCycle object to initialize. Must have the `id` set

`flags` - wrapped Long value specifying SigningCycle fields to initialize

Throws :

`SiatelException` - wrapped error

```
public Object findDocByLabel (String parentId, String label,
NativeJavaObject flags)
```

Searches for a document by its label.

JS call:

```
var result = Ged.findDocByLabel(parentId, docLabel, flags)
```

Parameters :

`parentId` - the id of the parent into which to search

`label` - the label of the searched document

`flags` - wrapped Long value specifying Document fields to initialize

Returns :

a list of Document objects, or a single Document, or `null` if nothing was found

Throws :

`SiatelException` - wrapped error

```
public Object findFolderByLabel (String parentId, String label,
NativeJavaObject flags)
```

Searches for a folder by its label.

JS call:

```
var result = Ged.findFolderByLabel(parentId, folderLabel, flags)
```

Parameters :

`parentId` - the id of the parent into which to search

`label` - the label of the searched folder

`flags` - wrapped Long value specifying Folder fields to initialize

Returns :

a list of Folder objects, or a single Folder, or `null` if nothing was found

Throws :

`SiatelException` - wrapped error

MAILSCRIPTING

```
public final class MailScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning e-mail functionalities. The methods in this class are bound to the JavaScript `Mail` namespace object's methods and properties.

Constructor summary :

Constructor
`MailScripting ()`
`MailScripting (Scriptable scope, Scriptable prototype)`

Method summary :

Modifier and type	Method
`public MailMessage`	`create ()`
`public void`	`send (NativeJavaObject message, java.lang.Object async)`
`public void`	`open (NativeJavaObject mailCtx, NativeJavaObject mailboxData)`
`public void`	`close (NativeJavaObject mailCtx)`
`public void`	`enumerate (NativeJavaObject mailCtx, java.lang.String whichMessages, NativeJavaObject list)`
`public MailMessage`	`readMessage (NativeJavaObject mailCtx, NativeJavaObject message, java.lang.Boolean markAsRead, java.lang.Boolean downloadAttachments)`
`public void`	`unreadMessage (NativeJavaObject mailCtx, NativeJavaObject message)`
`public boolean`	`check (NativeJavaObject mailCtx)`
`public MailboxData`	`createMailboxData ()`
`public MailContext`	`createMailContext ()`

Constructor detail :

```
public MailScripting ()
```

'Serializable' required constructor. Required for all implementations of ScriptingWrapper

```
public MailScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

`scope` -

`prototype` -

Method detail :

```
public MailMessage create ()
```

Create a mail message object in order to send it later. Alternate call to using a MailMessage constructor.

JS call:

```
Mail.create()
```

Returns :

a MailMessage object.

```
public void send (NativeJavaObject message, Object async)
```

Send a mail message.

JS call:

```
Mail.send(message[, async])
```

Parameters :

`message` - the MailMessage object to send

`async` - optional `boolean` to indicate if the operation should be synchronous or not

Throws :

`Exception` - in case mail send failed. Examine the exception message and stack-trace for details since it heavily depends on the underlying mail-system support

```
public void open (NativeJavaObject mailCtx, NativeJavaObject
mailboxData)
```

Open a session on a specified mailbox.

JS call:

```
Mail.open(mailCtx, mailboxData)
```

Parameters :

`mailCtx` - a MailContext object describing the session. To be initialized by this call

`mailboxData` - the MailboxData to be used for context initialization (server, account, user, pass etc.)

Throws :

`Exception` - as thrown by the underlying system mail support. Examine the exception message and stack-trace for details since it heavily depends on the underlying mail-system support

```
public void close (NativeJavaObject mailCtx)
```

Close the mailbox session.

JS call:

```
Mail.close(mailCtx)
```

Parameters :

`mailCtx` - the MailContext describing the session

Throws :

`Exception` - as thrown by the underlying system mail support. Examine the exception message and stack-trace for details since it heavily depends on the underlying mail-system support

```
public void enumerate (NativeJavaObject mailCtx, String whichMessages,
NativeJavaObject list)
```

List the messages in on open mailbox.

JS call:

```
Mail.enumerate(mailCtx, whichMessages, list)
```

Parameters :

`mailCtx` - the MailContext describing the session

`whichMessages` - String value establishing enumeration criteria. Can be one of `all`, `read`, `unread`

`list` - the java.util.List to fill with MailMessage objects, results of the enumeration

Throws :

`Exception` - as thrown by the underlying system mail support. Examine the exception message and stack-trace for details since it heavily depends on the system underlying mail support

```
public MailMessage readMessage (NativeJavaObject mailCtx,
NativeJavaObject message, Boolean markAsRead, Boolean
downloadAttachments)
```

Read a mail message.

JS call:

```
Mail.readMessage(mailCtx, message, markAsRead, downloadAttachments)
```

Parameters :

`mailCtx` - the MailContext describing the session

`message` - the MailMessage object to mark as read

`markAsRead` - `true` is message should be marked as read

`downloadAttachments` - `true` if message attachments should be downloaded and made available via the same `message` object received on input

Returns :

the same MailMessage object

Throws :

`SiatelException` - wrapped as thrown by the underlying system mail support. Examine the exception message and stack-trace for details since it heavily depends on the underlying mail-system support

```
public void unreadMessage (NativeJavaObject mailCtx, NativeJavaObject message)
```

Mark a mail message as unread.

JS call:

```
Mail.unreadMessage(mailCtx, message)
```

Parameters :

`mailCtx` - the MailContext describing the session

`message` - the MailMessage object to mark as unread

Throws :

`Exception` - as thrown by the underlying system mail support. Examine the exception message and stack-trace for details since it heavily depends on the system underlying mail support

```
public boolean check (NativeJavaObject mailCtx)
```

Check the open mailbox for new (unread) messages.

JS call:

```
Mail.check(mailCtx)
```

Parameters :

`mailCtx` - the MailContext describing the session and mailbox

Returns :

`true` if there are unread messages in the mailbox

Throws :

`Exception` - as thrown by the underlying system mail support. Examine the exception message and stack-trace for details since it heavily depends on the underlying mail-system support

```
public MailboxData createMailboxData ()
```

Instantiate a MailboxData object to be used for opening a session. Alternate call to using the MailboxData constructor.

JS call:

```
Mail.createMailboxData()
```

Returns :

a blank MailboxData object

```
public MailContext createMailContext ()
```

Instantiate a MailContext object to be initialized with a mail session description.

JS call:

```
Mail.createMailContext()
```

Returns :

a blank MailContext object

MiscScripting

```
public class MiscScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning miscellaneous functionalities. The methods in this class are bound to the JavaScript `Misc` namespace object's methods and properties.

Constructor summary :

Constructor
MiscScripting ()
MiscScripting (Scriptable scope, Scriptable prototype)

Method summary :

Modifier and type	Method
public void	signDocument (java.lang.String objectId, java.lang.String signatureType, java.lang.String certificateId, java.lang.Object convertToPdf)
public boolean	file2pdf (java.lang.String srcFilePath, java.lang.String pdfFilePath)
public java.lang.String	getLocalizedLabel (java.lang.String objectId, java.lang.String language)
public void	generateDocument (NativeJavaObject newDoc, java.lang.String parentId, java.lang.String templateId, NativeJavaObject attrSource, java.lang.String language)
public java.lang.String	generateFile (java.lang.String templateFile, NativeJavaObject attrSource, NativeJavaObject mappings)
public java.lang.String	form2pdf (NativeJavaObject object)
public void	zipFolder (java.lang.String srcFolder, java.lang.String zipFile)
public java.util.List	getAudit (java.lang.String objectId, java.lang.Boolean content, java.lang.Boolean recurse)
public java.lang.Object	attrValue (NativeJavaObject object, java.lang.String attrIdOrLabel, NativeJavaObject

	locale, NativeJavaObject defaultValue)
public java.lang.String	getExtensionInfo (java.lang.Object object, java.lang.String extensionId, java.lang.String instanceId, java.lang.String varName)
public java.util.List	getExtensionInstanceIds (java.lang.Object object, java.lang.String extensionId)
public java.lang.String	getCounterValue (java.lang.String counterId, java.lang.Object maybeIncrement)
public java.lang.String	getCounterValueEx (java.lang.String counterId, NativeJavaObject instanceSource)
public java.lang.String	lookupBarcode (java.lang.String idOrFilePath)
public java.lang.String	httpPostRequest (java.lang.String url, java.lang.Object _headers, java.lang.Object _parameters, java.lang.Object _body, java.lang.Object _contentType)
public java.lang.String	httpGetRequest (java.lang.String url, NativeJavaObject _headers)
public java.lang.String	httpHeadRequest (java.lang.String url, NativeJavaObject _headers)
public java.lang.String	httpDownloadRequest (java.lang.String url, NativeJavaObject _headers)
public java.lang.String	rebuildId (java.lang.String partialId)
public java.lang.String	broadcastDocument (java.lang.String objectId, NativeJavaObject _sDate, java.lang.Object _duration, java.lang.Object _maxDuration, NativeJavaObject _actorIds, java.lang.String ownerId, java.lang.String managerId, NativeJavaObject _bNotify)
public java.lang.Object	signRequest (java.lang.String objectId, java.lang.Object params)
public void	createObject (NativeJavaObject _object, java.lang.String parentId)
public void	readObject (java.lang.Object _object, java.lang.Object _flags, java.lang.Object _attributes)
public void	updateObject (NativeJavaObject _object,

	NativeJavaObject _flags)
public void	setAttributeValue (NativeJavaObject _attribute, java.lang.String value)
public java.lang.Object	getAttributeValue (NativeJavaObject _attribute, java.lang.String value)
public boolean	checkFlags (NativeJavaObject _ntLongFlags1, NativeJavaObject _ntLongFlags2)
public java.lang.Object	renderDelay (java.lang.String plannerId, java.lang.Object reference, java.lang.Object delay, java.lang.Object maxDelay)
public java.lang.Object	renderElapsed (java.lang.String plannerId, java.lang.Object dateS, java.lang.Object dateE)
public java.lang.Object	renderElapsedDays (java.lang.String plannerId, java.lang.Object dateS, java.lang.Object dateE)
public java.lang.Object	renderTextTemplate (java.lang.String template, java.lang.Object attrSource, java.lang.Object mappings)

Constructor detail :

```
public MiscScripting ()
```

Default constructor.

```
public MiscScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

scope -

prototype -

Method detail :

```
public void signDocument (String objectId, String signatureType, String certificateId, NativeJavaObject convertToPdf)
```

Sign a document using the given method and certificate. Requires a GARGANTUA Document on input. The signature type has to be one of the following:

- Constants.SIG_TYPE_CERTIFIED : the document is certified
- Constants.SIG_TYPE_ENCRYPTION : the document is encrypted

JS call:

```
Misc.signDocument(objectId, signatureType, certificateId,
convertToPdf)
```

Example :

```
function execute(params) {
        var oDocument = SiatelFactory.newObject('DOCF0076000000010000
         Ged.readDocument(oDocument, _l(Flag.DOCUMENT_ALL));

         System.log('--- before sign call', 'INFO');
         System.log(' document ID : ' + oDocument.getId(), 'INFO');
         System.log('     file ID : ' + oDocument.getFileId(), 'INFO'
         System.log('     form ID : ' + oDocument.getFormId(), 'INFO'

         var sCertificateId = Admin.idOfCertificate('SIATEL - GARGANTUA
         System.log('certicate ID : ' + sCertificateId, 'INFO');

         Misc.signDocument(
                oDocument.getId(),
                Constants.SIG_TYPE_CERTIFIED,
                 sCertificateId,
                 true);
         Ged.readDocument(oDocument, _l(Flag.DOCUMENT_ALL));

        System.log('--- after sign call', 'INFO');
        System.log(' document ID : ' + oDocument.getId(), 'INFO');
        System.log('     file ID : ' + oDocument.getFileId(), 'INFO')
        System.log('     form ID : ' + oDocument.getFormId(), 'INFO')
 }
```

Output of the example script :

```
[...] ScriptingSystem  - --- before sign call
[...] ScriptingSystem  -  document ID : DOCF00760000001000000000000
[...] ScriptingSystem  -      file ID : FILE00760000000000000010000
[...] ScriptingSystem  -      form ID : DFFR00760000001000000000000
[...] ScriptingSystem  - certicate ID : CERT00000000001F000000000000
[...] ScriptingSystem  - --- after sign call
[...] ScriptingSystem  -  document ID : DOCF00760000001000000000000
[...] ScriptingSystem  -      file ID : FILE00760000000000000010000
```

```
[...] ScriptingSystem  -      form ID : DFFR00760000001000000000000
```

Parameters :

`objectId` - the id of the document to sign

`signatureType` - one of Constants `.SIG_TYPE_CERTIFIED`, Constants `.SIG_TYPE_ENCRYPTION`, Constants `SIG_TYPE_ETOKEN` constants.

`certificateId` - the id of the certificate to use for signing

`convertToPdf` - boolean value specifying weather the original document should be replaced by the resulting signed pdf document. `true` to do replacement

Throws :

`SiatelException` - wrapped error

```
public boolean file2pdf (String srcFilePath, String pdfFilePath)
```

Convert a file to pdf.

JS call:

```
Misc.file2pdf(filePath, pdfPath)
```

Parameters :

`srcFilePath` - the path to the file to convert

`pdfFilePath` - the path of the resulting pdf file

Returns :

the result of the pdf conversion operation. `true` if the conversion was completed successfully.

Throws :

`SiatelException` - wrapped error

```
public String getLocalizedLabel (String objectId, String language)
```

Read the localized label of a design object (form, attribute etc.)

JS call:

```
Misc.getLocalizedLabel(objectId, lang)
```

Parameters :

`objectId` - id of object whose label is retrieved

`language` - localization language

Returns :

the localized object label in the specified language

Throws :

`SiatelException` - wrapped error

```
public void generateDocument (NativeJavaObject newDoc, String parentId,
String templateId, NativeJavaObject attrSource, String language)
```

Generate a document from a document template using an object as attribute data source.

JS call:

`Misc.generateDocument(newDoc, parentId, templateId, sourceAttrList, lang)`

Parameters :

`newDoc` - wrapped Document object to create

`parentId` - id of container where the document will be created. A folder, process or pin-document id

`templateId` - the id of the DocumentTemplate used to generate the document content

`attrSource` - the Viewable object (folder, process, document) used as data source for the document template markers

`language` - the language of the document template

Throws :

`SiatelException` - wrapped error

```
public String generateFile (String templateFile, NativeJavaObject
attrSource, NativeJavaObject mappings)
```

Generate a file from a document template using an object as attribute data source. It is the caller's responsibility to handle the file afterwards. The template provided is a document file, not a GARGANTUA document template

JS call:

`Misc.generateFile(templateFile, attrSource, mappings)`

Parameters :

`templateFile` - full file path of the template used to generate the document content

`attrSource` - the Viewable object (folder, process, document) used as data source for the document template markers

`mappings` - a map of markers to values used for replacing text in the template

Throws :

`SiatelException` - wrapped error

```
public String form2pdf (NativeJavaObject object)
```

Render a PDF file using an object's form and it's attributes to fill the form fields.

JS call:

```
var pdfFilePath = Misc.form2pdf(object)
```

Parameters :

`object` - the object whose form and attributes are used to render the pdf file

Returns :

the path to the rendered PDF file

Throws :

`SiatelException` - wrapped error

```
public void zipFolder (String srcFolder, String zipFile)
```

Create a `.zip` or `.tgz` file from file-system folder (recursively). The extension of the specified output file will determine the output format.

JS call:

```
Misc.zipFolder(srcFolder, zipFile)
```

Parameters :

`srcFolder` - the file-system source folder

`zipFile` - the output file

Throws :

`SiatelException` - wrapped error

```
public List getAudit (String objectId, Boolean content, Boolean recurse)
```

Get the audit of a given object, optionally recursing to content and children.

JS call:

```
Misc.getAudit(objectId, content, recurse)
```

Parameters :

`objectId` - the object whose audit is requested

`content` - `true` if immediate children documents' audit should also be queried (applies to

folder/process/pin objects)

`recurse` - `true` if children folders' audit should be queried (applies to folder objects)

Returns :

the List of AuditEntry objects

Throws :

`SiatelException` - wrapped error

```
public Object attrValue (NativeJavaObject object, String attrIdOrLabel,
NativeJavaObject locale, NativeJavaObject defaultValue)
```

Get the value of an attribute for a given object, based on either attribute id or label, a localization language and optional fallback default value.

JS call:

```
var value = Misc.attrValue(object, attrIdOrLabel, locale,
defaultValue)
```

Parameters :

`object` - a Viewable implementing object (folder, process, pin or document)

`attrIdOrLabel` - the id or the label of the attribute whose values is requested

`locale` - a Locale language specifier

`defaultValue` - the fallback value, depending on attribute type. Can be omitted or `null`.

Returns :

the requested attribute's value (or fallback value if not found or `null` and a fallbcak value is specified)

Throws :

`SiatelException` - wrapped error

```
public String getExtensionInfo (Object object, String extensionId,
String instanceId, String varName)
```

Get an extension's object-assigned-instance variable value.

JS call:

```
var value = Misc.getExtensionInfo(object, extensionId, instanceId,
varName)
```

Parameters :

`object` - the object whose extensions to look up

`extensionId` - the extension id

`instanceId` - the extension instance id

`varName` - the variable name

Returns :

the value of the extension instance variable

Throws :

`SiatelException` - wrapped error

```
public List getExtensionInstanceIds (Object object, String extensionId)
```

List an extension's instances' ids for a given object

JS call:

```
var instanceIds = Misc.getExtensionInstanceIds(object, extensionId)
```

Parameters :

`object` - the object whose extension to lookup

`extensionId` - the extension id

Returns :

the List of extension's instances' ids

Throws :

`SiatelException` - wrapped error

```
public String getCounterValue (String counterId, Object maybeIncrement)
```

Fetch a new value from a counter attribute. This applies to counter attributes that **DO NOT** have counter instance criteria (like for example folders in which case the attribute would render a counter instance for every folder having said counter attribute in it's form)

JS call:

```
var counterValue = Misc.getCounterValue(counterId, maybeIncrement)
```

Parameters :

`counterId` - the counter attribute id

`maybeIncrement` - increment the counter after value retrieved. default `true`

Returns :

the new counter value

Throws :

`SiatelException` - wrapped error

```
public String getCounterValueEx (String counterId, NativeJavaObject
instanceSource)
```

Fetch a new value from a counter attribute. This applies to counter attributes that **DO** have counter instance criteria (like for example folders in which case the attribute will render a counter instance for every folder having said counter attribute in it's form)

JS call:

```
var counterValue = Misc.getCounterValue(counterId, instanceSource)
```

Parameters :

`counterId` - the counter attribute id

`instanceSource` - the object that instantiates the counter attribute

Returns :

the new counter value

Throws :

`SiatelException` - wrapped error

```
public String lookupBarcode (String idOrFilePath)
```

Lookup barcodes in an image file obtained either from a document id or a file id or a file path.

JS call:

```
var barcodesValue = Misc.lookupBarcode(idOrFilePath)
```

Parameters :

`idOrFilePath` - can be a document id or a file id or a file path

Returns :

the found barcodes' content

Throws :

`SiatelException` - wrapped error

`IOException` - in case supplied argument is not an id and attempts to render it as file-system path fail

```
public String httpPostRequest (String url, Object _headers, Object
_parameters, Object _body, Object _contentType)
```

Send a HTTP POST request and receive result as string.

JS call:

```
var responseText = Misc.httpPostRequest(url, headers, parameters,
body, contentType)
```

Parameters :

`url` - the URL to send the request to

`_headers` - Map of headers for the request

`_parameters` - Map of POST parameters for the request

`_body` - the request body as string

`_contentType` - the payload Content-Type of the HTTP request

Returns :

the response body as text

Throws :

`SiatelException` - wrapped error

`HttpException` - HTTP subsystem deferred error

`IOException` - HTTP subsystem deferred error

```
public String httpGetRequest (String url, NativeJavaObject _headers)
```

Send a HTTP GET request and receive result as string.

JS call:

```
var responseText = Misc.httpGetRequest(url, headers)
```

Parameters :

`url` - the URL to send the request to

`_headers` - Map of headers for the request

Returns :

the response body as text

Throws :

`SiatelException` - wrapped error

`HttpException` - HTTP subsystem deferred error

`IOException` - HTTP subsystem deferred error

```
public String httpHeadRequest (String url, NativeJavaObject _headers)
```

Send a HTTP HEAD request and receive result as string.

JS call:

```
var responseText = Misc.httpHeadRequest(url, headers)
```

Parameters :

`url` - the URL to send the request to

`_headers` - Map of headers for the request

Returns :

the response body as text

Throws :

`SiatelException` - wrapped error

`HttpException` - HTTP subsystem deferred error

`IOException` - HTTP subsystem deferred error

```
public String httpDownloadRequest (String url, NativeJavaObject
_headers)
```

Send a HTTP GET request and receive result as file on disk.

JS call:

```
var fileName = Misc.httpDownloadRequest(url, headers)
```

Parameters :

`url` - the URL to send the request to

`_headers` - Map of headers for the request

Returns :

the response body as text

Throws :

`SiatelException` - wrapped error

`HttpException` - HTTP subsystem deferred error

`IOException` - HTTP subsystem deferred error

```
public String rebuildId (String partialId)
```

Rebuilds a full ID based on a partial ID.

JS call:

```
var fullId = Misc.rebuildId(partialId)
```

Parameters :

`partialId` - The partial ID to analyze

Returns :

the real and full ID of the requested object

Throws :

`SiatelException` - wrapped error

```
public String broadcastDocument (String objectId, NativeJavaObject
_sDate, Object _duration, Object _maxDuration, NativeJavaObject
_actorIds, String ownerId, String managerId, NativeJavaObject _bNotify)
```

Broadcast one or more documents to one or more specific targets.

JS call:

```
var validationCycleId = Misc.broadcastDocument(objectId, startDate,
readDuration, readMaxDuration, actorIds, ownerId, managerId, notify)
```

Parameters :

`objectId` - document to be broadcasted

`_sDate` - start date for broadcast vcycle

`_duration` - duration for 'read' stage

`_maxDuration` - maximum duration for 'read' stage

`_actorIds` - who to broadcast to; list of user ids

`ownerId` - owner of the broadcast operation; defaults to the owner of the document

`managerId` - manager of the broadcast operation; defaults to the owner of the document

`_bNotify` - true if recipients are to be notified

Returns :

the broadcast validation cycle ID

Throws :

`SiatelException` - wrapped error

```
public Object signRequest (String objectId, Object params)
```

Start a sign request for a specific document

Please refer to the section Misc.signRequest for more detailed information.

JS call:

```
var signRequestId = Misc.signRequest(objectId, signRequestData)
```

Parameters :

`objectId` - the id of the document for which you can start a sign request circuit

`params` - the sign request data, as JSON object

Returns :

the sign request circuit ID

Throws :

`SiatelException` - wrapped error

```
public void createObject (NativeJavaObject _object, String parentId)
```

Create a new document/folder/process. This unifies `Ged.createDocument(...)`, `Ged.createFolder(...)`, `Flow.createProcess(...)`.

JS call:

```
Misc.createObject(object, parentId)
```

Parameters :

`_object` - the folder/document/process object

`parentId` - the id of the parent where to create the folder or document. It does not apply to process targets

Throws :

`SiatelException` - wrapped error

```
public void readObject (Object _object, Object _flags, Object
_attributes)
```

Lazy-load an object (folder/pin/document/process). This method unifies `Ged.readFolder(...)`, `Ged.readPin(...)`, `Ged.readDocument(...)`, `Flow.readProcess(...)`.

JS call:

```
Misc.readObject(object, flags[, attrIds])
```

Parameters :

`_object` - the object to fill in with data loaded from persistence

`_flags` - which fields of the object to retrieve from persistence

`_attributes` - which attributes of the object to retrieve from persistence, packed as javascript array or java native List ; optional, can also be null, empty javascript array or empty java native List

Throws :

`SiatelException` - wrapped error

```
public void updateObject (NativeJavaObject _object, NativeJavaObject
_flags)
```

Update an object (folder/pin/document/process). This method unifies `Ged.updateFolder(...)`, `Ged.updatePin(...)`, `Ged.updateDocument(...)`, `Flow.updateProcess(...)`.

JS call:

```
Misc.updateObject(object, flags)
```

Parameters :

`_object` - the object to update

`_flags` - which fields of the object to save to persistence

Throws :

`SiatelException` - wrapped error

```
public void setAttributeValue (NativeJavaObject _attribute, String
value)
```

Set an attribute's value using a String representation of it's native internal value. This will internally convert the string value to the accepted native value of the attribute

JS call:

```
Misc.setAttributeValue(attribute, value)
```

Parameters :

`_attribute` - the attribute whose value to set. Must have the `id` and `typeId` properties set

`value` - the attribute value specified as string

Throws :

`SiatelException` - wrapped error

```
public Object getAttributeValue (NativeJavaObject _attribute, String
value)
```

Convert a string value to the accepted native value of an attribute.

JS call:

```
Misc.getAttributeValue(attribute, value)
```

Parameters :

`_attribute` - the attribute whose type to use for conversion. Must have the `id` and `typeId` properties set

`value` - the string value to convert

Throws :

`SiatelException` - wrapped error

```
public boolean checkFlags (NativeJavaObject _ntLongFlags1,
NativeJavaObject _ntLongFlags2)
```

Test two sets of flags against each other and check if the first set contains the second set of flags. Used to rigorously test Long flag values outside of JavaScript which converts them to JavaScript numbers.

Parameters :

`_ntLongFlags1` -

`_ntLongFlags2` -

Returns :

`true` if applying bitwise AND between the two values equals the second method argument

Throws :

`SiatelException` - wrapped error

```
public Object renderDelay (String plannerId, Object reference, Object
delay, Object maxDelay)
```

Render a delay in milliseconds according to a planner and a reference point.

JS call:

`Misc.renderDelay(plannerId, reference, delay, maxDelay)`

Parameters :

`plannerId` - id of planner that describes working time

`reference` - the Long millisecond-specified reference point in time

`delay` - the Long millisecond-specified duration to be rendered according to a planner. Can be negative in which case the delay is rendered backwards (as in 'before the given reference')

`maxDelay` - the Long millisecond-specified maximum duration allowed; this parameter is optional; if not specified, Long.MAX_VALUE will be used

Returns :

the rendered timestamp (in milliseconds)

Throws :

`SiatelException` - wrapped error

```
public Object renderElapsed (String plannerId, Object dateS, Object
dateE)
```

Render the activity elapsed duration in milliseconds of a time interval according to a planner.

JS call:

```
Misc.renderElapsed(plannerId, dateS, dateE)
```

Parameters :

`plannerId` - id of planner that describes working time

`dateS` - the Long millisecond-specified or Date of time interval start

`dateE` - the Long millisecond-specified or Date of time interval end

Returns :

the rendered elapsed duration (in milliseconds)

Throws :

`SiatelException` - wrapped error

```
public Object renderElapsedDays (String plannerId, Object dateS, Object
dateE)
```

Render the activity elapsed duration in days of a time interval according to a planner.

JS call:

```
Misc.renderElapsedDays(plannerId, dateS, dateE)
```

Parameters :

`plannerId` - id of planner that describes working time

`dateS` - the Long millisecond-specified or Date of time interval start

`dateE` - the Long millisecond-specified or Date of time interval end

Returns :

the rendered elapsed duration (in days)

Throws :

`SiatelException` - wrapped error

```
public Object renderTextTemplate (String template, Object attrSource,
Object mappings)
```

Render the supplied text template given the attribute source object and extra mappings.

JS call:

```
Misc.renderTextTemplate(templateText, attrSource, mappingsExtra)
```

Parameters :

`template` - the template text, containing markers to be replaced

`attrSource` - the String id of or Viewable object containing attributes' values to be used in the evaluation of template markers. This parameter * can be null if template does not base it's rendering on any attribute values belonging to an object

`mappings` - the Map or native javascript object containing custom extra mappings. This parameter is optional and can be omitted

Returns :

the String rendered text

Throws :

`SiatelException` - wrapped error

PluginScripting

```
public class PluginScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning plugin invocation functionalities. The methods in this class are bound to the JavaScript `Plugin` namespace object's methods and properties.

Constructor summary :

Constructor
`PluginScripting ()`
`PluginScripting (Scriptable scope, Scriptable prototype)`

Method summary :

Modifier and type	Method
`public java.lang.String`	`invokePlugin (java.lang.String pluginId, java.lang.String command, NativeJavaObject params)`
`public boolean`	`check (java.lang.String pluginId)`

Constructor detail :

```
public PluginScripting ()
```

Default constructor.

```
public PluginScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

`scope` -

`prototype` -

Method detail :

```
public String invokePlugin (String pluginId, String command,
NativeJavaObject params)
```

Invoke a plugin.

JS call:

```
Plugin.invokePlugin(pluginId, command, params)
```

Parameters :

`pluginId` - the id of the plugin to invoke

`command` - the plugin specific command to invoke

`params` - DTO containing the call parameters. This is actually a map of parameter name to parameter value entries.

Returns :

the result (if any). Can be null if no result is provided

Throws :

`SiatelException` - wrapped error

```
public boolean check (String pluginId)
```

Check availability of a plugin.

JS call:

`Plugin.check(pluginId)`

Parameters :

`pluginId` - the id of the plugin to check

Returns :

`true` if plugin is installed, licensed and enabled

Throws :

`SiatelException` - wrapped error

SearchScripting

```
public class SearchScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning search functionalities. The methods in this class are bound to the JavaScript `Search` namespace object's methods and properties.

Author :

victorc

Constructor summary :

Constructor
SearchScripting ()
SearchScripting (Scriptable scope, Scriptable prototype)

Method summary :

Modifier and type	Method
public Expression	newExpression ()
public Expression	newSimpleExpression (NativeJavaObject condition)
public Expression	newComplexExpression (NativeJavaObject left, int operator, NativeJavaObject right)
public Condition	newAttributeCondition (NativeJavaObject attr, int operator, NativeJavaObject value)
public Condition	newFormCondition (java.lang.String formId, int operator)
public Condition	newFTCondition (java.lang.String dbID, java.lang.String text)
public java.util.List	searchAdvanced (java.lang.String databaseId, NativeJavaObject flags, NativeJavaObject expression)
public java.util.List	searchAdvancedExt (java.lang.String databaseId, NativeJavaObject flags, NativeJavaObject expression, NativeJavaObject additionalData)

Constructor detail :

```
public SearchScripting ()
```

Default constructor.

```
public SearchScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

`scope` -

`prototype` -

Method detail :

```
public Expression newExpression ()
```

Create a search Expression instance.

JS call:

```
Search.newExpression()
```

Returns :

the blank Expression instance

```
public Expression newSimpleExpression (NativeJavaObject condition)
```

Create a search Expression instance that has in it's node the Condition specified as parameter.

JS call:

```
Search.newSimpleExpression(condition)
```

Parameters :

`condition` - the Condition to use as in the expression

Returns :

the created Expression instance

```
public Expression newComplexExpression (NativeJavaObject left, int
operator, NativeJavaObject right)
```

Create a search Expression instance that combines two other expressions using a specified operator or applies the operator (if unary, like negation and `right` is `null`) to the expressions specified as `left`.

JS call:

```
Search.newComplexExpression(left, operator, right)
```

Parameters :

`left` - an Expression

`operator` - the logical operator. One of `Operator.AND`, `Operator.OR`, `Operator.NOT` constants from Operator class.

`right` - an Expression or `null` if operator is `Operator.NOT`

Returns :

the created Expression instance

```
public Condition newAttributeCondition (NativeJavaObject attr, int
operator, NativeJavaObject value)
```

Create a comparison Condition instance that will test an attribute's value using an operator and a reference value.

JS call:

```
Search.newAttributeCondition(attr, operator, value)
```

Parameters :

`attr` - the Attribute whose value to test

`operator` - the comparison operator. One of `Operator.IN` (is null), `Operator.NN` (not null), `Operator.EQ` (equals), `Operator.NE` (not equals), `Operator.LT` (less than), `Operator.LE` (less or equal), `Operator.GE` (greater or equal), `Operator.GT` (greater than), `Operator.SW` (starts with), `Operator.EW` (ends with), `Operator.CT` (contains), `Operator.NC` (does not contain) constants from Operator class. The following operators do not require a reference value: `Operator.IN`, `Operator.NN`. The following operators apply only to string-valued attributes (including 'list' types): `Operator.SW`, `Operator.EW`, `Operator.CT`, `Operator.NC`

`value` - the reference value used for comparison where operators require it

Returns :

the created Condition instance

```
public Condition newFormCondition (String formId, int operator)
```

Create a comparison Condition instance that will test the forms of searchable entities against a specific form specified by id.

JS call:

```
Search.newFormCondition(formId, operator)
```

Parameters :

`formId` - the id of the form used as comparison reference

`operator` - the comparison operator. `Operator.EQ` (equals), `Operator.NE` (not equals) constants from Operator class

Returns :

the created Condition instance

```
public Condition newFTCondition (String dbID, String text)
```

Create a comparison Condition instance that will test the fulltext of searchable entities against a specific text.

JS call:

```
Search.newFTCondition(dbID, text)
```

Parameters :

`dbID` - the database id to search text into

`text` - the text

Returns :

the created Condition instance

```
public List searchAdvanced (String databaseId, NativeJavaObject flags,
NativeJavaObject expression)
```

Advanced search invocation.

JS call:

```
Search.advancedSearch(databaseId, flags, expression)
```

Parameters :

`databaseId` - the database to search

`flags` - Long combination of flags to customize the list of search results. See com.siatel.domain.Flag

`expression` - the search Expression

Returns :

the list of hits as List <`SearchResult`>

Throws :

`SiatelException` - wrapped error

```
public List searchAdvancedExt (String databaseId, NativeJavaObject
flags, NativeJavaObject expression, NativeJavaObject additionalData)
```

Advanced search invocation.

JS call:

`Search.advancedSearchExt(databaseId, flags, expression, additionalData)`

Parameters :

`databaseId` - the database to search

`flags` - Long combination of flags to customize the list of search results. See com.siatel.domain.Flag

`expression` - the search Expression

`additionalData` - additional data for search; will be added to the search context object

Returns :

the list of hits as List <`SearchResult`>

Throws :

`SiatelException` - wrapped error

SiatelFactory

`public class SiatelFactory`

This scripting utility-class wraps methods accessible from the scripting engine concerning object creation functionalities. The methods in this class are bound to the JavaScript `SiatelFactory` namespace object's methods and properties.

Author :

stefanu

Constructor summary :

Constructor
`SiatelFactory ()`
`SiatelFactory (Scriptable scope, Scriptable prototype)`

Method summary :

Modifier and type	Method
`public java.lang.Object`	`newObject (java.lang.String idOrPref)`
`public java.lang.Object`	`newLabeledObject (java.lang.String prefix, java.lang.String label)`
`public java.lang.Object`	`newAccessMask (java.lang.String identityId, java.lang.Boolean visible,`

	java.lang.NativeJavaObject lMask, java.lang.Boolean revoke)

Constructor detail :

```
public SiatelFactory ()
```

Default constructor.

```
public SiatelFactory (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

scope -

prototype -

Method detail :

```
public Object newObject (String idOrPref)
```

Creates an object according to the requested type

JS call:

```
var obj = SiatelFactory.newObject(idOrPref)
```

Parameters :

idOrPref - Either a full identifier or just the object prefix

Returns :

the newly created object

```
public Object newLabeledObject (String prefix, String label)
```

Creates an object according the the requested type and label; useful when reading objects by label from the database; please note that this rreading method only works with objects that have unique labels, such as attributes or databases.

JS call:

```
var obj = SiatelFactory.newLabeledObject(prefix, label)
```

Parameters :

prefix - the required object prefix

label - the label

Returns :

the newly created object

```
public Object newAccessMask (String identityId, Boolean visible,
NativeJavaObject lMask, Boolean revoke)
```

Creates an access mask for a given entity; access masks are grouped in a security profile, either custom or globally defined

JS call:

```
var amask = SiatelFactory.newAccessMask(identityId, visible, lMask,
revoke)
```

Parameters :

`identityId` - indicates the entity identifier; can be either a group, role or an user.

`visible` - indicates whether the entity can view or not the object

`lMask` - the access mask; a combination of right flags

`revoke` - indicates whether it's a revoked mask

Returns :

the access mask itself; use it as part of a security profile

StorageScripting

```
public class StorageScripting
```

This scripting utility-class wraps methods accessible from the scripting engine concerning storage functionalities. The methods in this class are bound to the JavaScript `Storage` namespace object's methods and properties.

Author :

victorc

Constructor summary :

Constructor
StorageScripting ()
StorageScripting (Scriptable scope, Scriptable prototype)

Method summary :

Modifier and type	Method
public java.lang.String	storeFile (java.lang.String databaseId, java.lang.String filePath, java.lang.Boolean forceSystemStorage)
public java.lang.String	newFile (java.lang.String extension, java.lang.String fileText)
public java.lang.String	createFile (java.lang.String databaseId, java.lang.String extension, java.lang.String fileText, java.lang.Boolean forceSystemStorage)
public java.lang.String	createTextFile (java.lang.String databaseId, java.lang.String fileText, java.lang.Boolean forceSystemStorage)
public java.lang.String	copyFile (java.lang.String fileId, java.lang.String databaseId, java.lang.Boolean forceSystemStorage)
public void	deleteFile (java.lang.String fileId)
public void	cacheFile (java.lang.String fileId, java.lang.String dir)
public void	replaceFile (java.lang.String fileId, java.lang.String filePath)
public com.siatel.dto.DTO	readDtoFile (java.lang.String fileId)

public void	writeDtoFile (java.lang.String fileId, NativeJavaObject dto)
public java.lang.String	readTextFile (java.lang.String fileId)
public void	writeTextFile (java.lang.String text, java.lang.String fileIdOrPath)
public java.lang.String	renderStreamingUrl (java.lang.String fileId)

Constructor detail :

```
public StorageScripting ()
```

Default constructor.

```
public StorageScripting (Scriptable scope, Scriptable prototype)
```

Base class constructor.

Parameters :

scope -

prototype -

Method detail :

```
public String storeFile (String databaseId, String filePath, Boolean
forceSystemStorage)
```

Store a file for a specified database.

JS call:

```
var fileId = Storage.storeFile(databaseId, filePath,
forceSystemStorage)
```

Parameters :

databaseId - the database for which to store the file. The administration database (used for resources for example) is specified as database with numeric id '0'

filePath - path to the file to store

forceSystemStorage - force the system storage. This parameter is optional

Returns :

the id of the stored file

Throws :

`SiatelException` - wrapped error

```
public String newFile (String extension, String fileText)
```

Create a new temporary text file with a given extension and text content.

JS call:

```
var tempFilePath = Storage.newFile(extension, fileText)
```

Parameters :

`extension` - the file extension without the '.' prefix

`fileText` - the file text content

Returns :

the path to the newly created file

Throws :

`IOException` - file-system error

```
public String createFile (String databaseId, String extension, String
fileText, Boolean forceSystemStorage)
```

Create a file with the given extension for a specified database given specified text as content.

JS call:

```
var fileId = Storage.createFile(databaseId, extension, text,
forceSystemStorage)
```

Parameters :

`databaseId` - the database for which to store the file. The administration database (used for resources for example) is specified as database with numeric id '0'

`extension` - extension for the file to be created without the '.' prefix

`fileText` - text content for the file to be created

`forceSystemStorage` - force the system storage. This parameter is optional

Returns :

the id of the stored text file

Throws :

`SiatelException` - wrapped error

```
public String createTextFile (String databaseId, String fileText,
```

```
Boolean forceSystemStorage)
```

Create a text file for a specified database given specified text as content.

JS call:

```
var fileId = Storage.createTextFile(databaseId, text,
forceSystemStorage)
```

Parameters :

`databaseId` - the database for which to store the file. The administration database (used for resources for example) is specified as database with numeric id '0'

`fileText` - text content for the file to be created

`forceSystemStorage` - force the system storage. This parameter is optional

Returns :

the id of the stored text file

Throws :

`SiatelException` - wrapped error

```
public String copyFile (String fileId, String databaseId, Boolean
forceSystemStorage)
```

Copy a stored file to a new file to be stored for a specified database.

JS call:

```
var copyFileId = Storage.copyFile(fileId, databaseId,
forceSystemStorage)
```

Parameters :

`fileId` - the id of the file to be used as copy source

`databaseId` - the database for which to store the copy file. The administration database (used for resources for example) is specified as database with numeric id '0'

`forceSystemStorage` - force the system storage. This parameter is optional

Returns :

the id of the stored copy-file

Throws :

`SiatelException` - wrapped error

```
public void deleteFile (String fileId)
```

Delete a file from storage, based on it's id.

JS call:

```
Storage.deleteFile(fileId)
```

Parameters :

`fileId` - the id of the file to delete

Throws :

`SiatelException` - wrapped error

```
public void cacheFile (String fileId, String dir)
```

Extract a stored file to a specified location.

JS call:

```
Storage.cacheFile(fileId, dir)
```

Parameters :

`fileId` - the id of the file to extract

`dir` - the location where to extract the file. The extracted file will be named by it's id in the extraction location

Throws :

`SiatelException` - wrapped error

```
public void replaceFile (String fileId, String filePath)
```

Replace a stored file's content with the content of a file specified by path.

JS call:

```
Storage.replaceFile(fileId, filePath)
```

Parameters :

`fileId` - the id of the file whose content to replace

`filePath` - the file whose content is to be used as replacement source

Throws :

`SiatelException` - wrapped error

```
public DTO readDtoFile (String fileId)
```

Read the content of a stored file and return it's content as a DTO instance. The content of the file should be a previously written DTO or conform to the DTO serialization encoding.

JS call:

```
var dto = Storage.readDtoFile(fileId)
```

Parameters :

`fileId` - the id of the file to read

Returns :

the DTO-parsed content of the file

Throws :

`SiatelException` - wrapped error

```
public void writeDtoFile (String fileId, NativeJavaObject dto)
```

Replace the content of a file specified by id with the serialized rendition of a DTO instance.

JS call:

```
Storage.writeDtoFile(fileId, dto)
```

Parameters :

`fileId` - the id of the file to be used as copy source

`dto` - the dto whose content is serialized and used as source for the content replacement

Throws :

`SiatelException` - wrapped error

```
public String readTextFile (String fileId)
```

Read the content of a stored file and return it's content as text.

JS call:

```
var text = Storage.readTextFile(fileId)
```

Parameters :

`fileId` - the id of the file to read

Returns :

the text content of the file

Throws :

`SiatelException` - wrapped error

```
public void writeTextFile (String text, String fileIdOrPath)
```

Replace the content of a file specified by id with specified text.

JS call:

```
Storage.writeTextFile(text, fileIdOrPath)
```

Parameters :

`text` - the text content to be used as source for content replacement

`fileIdOrPath` - the id or absolute path of the file to be used as content replacement target

Throws :

`SiatelException` - wrapped error

```
public String renderStreamingUrl (String fileId)
```

Render the streaming-usable url of a file.

JS call:

```
var url = Storage.writeTextFile(text, fileIdOrPath)
```

Parameters :

`fileId` - the id of the file whose content streaming is pursued

Returns :

the url to be used for streaming (e.g. in a streaming player application or browser)

Throws :

`SiatelException` - wrapped error

Package com.siatel.defs

This package contains interface definitions that group domain object properties.

Interfaces

Archive

```
public interface Archive
```

Accessor wrapper interface for archive enabled objects (ArchiveFolder, ArchiveDocument, ArchiveDocumentVersion, ArchiveNote).

Since :

7.5

Method summary :

Modifier and type	Method
public java.util.Date	getDateArchive ()
public void	setDateArchive (java.util.Date d)

Method detail :

```
public Date getDateArchive ()
```

Accessor.

Returns :

the date when object was archived

```
public void setDateArchive (Date d)
```

Accessor.

Parameters :

d - the value to set for the date when object was archived.

COMMENTED

```
public interface Commented
```

Commented objects are objects that have a comment; this comment has no other purpose than help the administrator or users understand the purpose of the object.

Since :

7.5

See also :

Labeled

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getComment ()
public void	setComment (java.lang.String comment)

Method detail :

```
public String getComment ()
```

Returns :

the comment for the specified object

```
public void setComment (String comment)
```

Parameters :

comment - the new comment for the specified object

DATED

```
public interface Dated
```

Dated interface is implemented by all objects that can have a creation date and a modification date; the Dated objects can be traced.

Since :

7.0

Author :

stefanu

See also :

DatedEx

Method summary :

Modifier and type	Method
public java.util.Date	getDateC ()
public void	setDateC (java.util.Date dateC)
public java.util.Date	getDateM ()
public void	setDateM (java.util.Date dateM)

Method detail :

```
public Date getDateC ()
```

Returns :

the creation date and time of the specified object

```
public void setDateC (Date dateC)
```

Parameters :

dateC - the new creation date and time for the specified object

```
public Date getDateM ()
```

Returns :

the last modification date and time of the specified object

```
public void setDateM (Date dateM)
```

Parameters :

dateM - the new last modification date and time of the specified object

DATEDEX

```
public interface DatedEx
```

DatexEx objects keep track of the last access date and time, as an extension to the Dated objects that keep track of the creation and last modification date and time

All Superinterfaces :

Dated

Since :

7.0

See also :

Dated

Author :

stefanu

Method summary :

Modifier and type	Method
public java.util.Date	getDateA ()
public void	setDateA (java.util.Date dateA)

Method detail :

```
public Date getDateA ()
```

Returns :

the last access date and time for the specified object

```
public void setDateA (Date dateA)
```

Parameters :

dateA - the new last access date and time for the specified object

Deletable

```
public interface Deletable
```

The Deletable interface is used by all objects that have a flag for delete.

Since :

7.9

Author :

ying

Method summary :

Modifier and type	Method
public boolean	getDeleted ()
public void	setDeleted (boolean deleted)
public java.util.Date	getDateDel ()
public void	setDateDel (java.util.Date dateDel)
public int	getDeletedBy ()
public void	setDeletedBy (int deletedBy)

Method detail :

```
public boolean getDeleted ()
```

Returns :

true if object is in trash

```
public void setDeleted (boolean deleted)
```

Parameters :

`deleted` - the new 'deleted to trash' status for the object

```
public Date getDateDel ()
```

Returns :

the date and time when the object was moved to trash

```
public void setDateDel (Date dateDel)
```

Parameters :

`dateDel` - the date and time when the object was moved to trash

```
public int getDeletedBy ()
```

Returns :

numeric id of user who moved the object to trash

```
public void setDeletedBy (int deletedBy)
```

Parameters :

`deletedBy` - the numeric id of user who moved the object to trash

Deployable

```
public interface Deployable
```

Deployable objects are design objects; most of the design objects are not meant to be put in production immediately; this means that creating such an object may require significant time and effort from the system administrator, and the system state may become unstable if such objects are made available for the users immediately after they've been saved into the database. Such objects are, for instance, processes. processes are complex objects that must not be put into production immediately; the processes are created (which may require even more than one week's work from the system administrator), then 'deployed' once the process is ready to work as intended. Users have access to Deployable objects only when then are deployed; otherwise, these objects are only available in the administrative tools. Some objects may require validation or compilation as they are deployed. Validation errors will block the deploy command and leave the object in an undeployed state.

Since :

7.0

See also :

DeployableEx

Author :

stefanu

Method summary :

Modifier and type	Method
`public boolean`	`isDeployed ()`
`public void`	`setDeployed (boolean deployed)`

Method detail :

```
public boolean isDeployed ()
```

Returns :

the deployment status of the specified object

```
public void setDeployed (boolean deployed)
```

Parameters :

`deployed` - the new deployment status for the specified object

DEPLOYABLEEX

```
public interface DeployableEx
```

DeployableEx objects are objects that take an alternate form when deployed; an example are the form objects; when deployed, these objects are 'compiled' into JSP pages that can be displayed by the JSP engine. When DeployableEx objects are deployed, a secondary file is created and stored for use in production.

All Superinterfaces :

Deployable

Since :

7.0

See also :

Deployable

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getDeploymentId ()
public void	setDeploymentId (java.lang.String deploymentId)
public long	getDeploymentTimestamp ()
public void	setDeploymentTimestamp (long millis)

Method detail :

```
public String getDeploymentId ()
```

Returns :

the deployment ID for the specified object

```
public void setDeploymentId (String deploymentId)
```

Parameters :

`deploymentId` - the new deployment ID for the specified object

```
public long getDeploymentTimestamp ()
```

Returns :

the deployment time stamp for the specified object; this time stamp is required by the deployment operation to assess the need of transformation, checking that the deployed object has been modified since last deployment command

```
public void setDeploymentTimestamp (long millis)
```

Parameters :

`millis` - the new deployment time stamp for the specified object

DESCRIBED

```
public interface Described
```

Described objects are objects that have a description; this description has no other purpose than help the administrator or users understand the purpose of the object.

Descriptions may be displayed as tooltips by the user interface.

Since :

7.0

See also :

Labeled

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getDescription ()
public void	setDescription (java.lang.String description)

Method detail :

```
public String getDescription ()
```

Returns :

the description for the specified object

```
public void setDescription (String description)
```

Parameters :

description - the new description for the specified object

EXPIRABLE

```
public interface Expirable
```

Expirable objects are part of the workflow related objects; these objects have an due date and an expiration date; when the due date is reached, the object is not yet expired, but it will be signaled by the user interface differently; the due date and the expiration date are needed by the task and process state transition algorithm

Since :

7.0

Author :

stefanu

Method summary :

Modifier and type	Method
public java.util.Date	getDateD ()
public void	setDateD (java.util.Date dateD)
public boolean	isOverdue ()
public void	setOverdue (boolean overdue)
public void	setDateX (java.util.Date dateX)
public boolean	isExpired ()
public void	setExpired (boolean expired)

Method detail :

```
public Date getDateD ()
```

Returns :

the due date of the specified object

```
public void setDateD (Date dateD)
```

Parameters :

`dateD` - the new due date for the specified object

```
public boolean isOverdue ()
```

Returns :

the overdue status of the specified object

```
public void setOverdue (boolean overdue)
```

Parameters :

`overdue` - the new overdue status of the specified object

```
public void setDateX (Date dateX)
```

Parameters :

`dateX` - the new expiration date for the specified object

```
public boolean isExpired ()
```

Returns :

the expiration status for the specified object

```
public void setExpired (boolean expired)
```

Parameters :

`expired` - the new expiration status for the specified object

FLAGGED

```
public interface Flagged
```

Flagged objects are objects that may have a set of flags associated with them. Flags are defined by the Flag constants, and are ORed constants

Since :

7.0

See also :

Flag

Author :

stefanu

Method summary :

Modifier and type	Method
public long	getFlags ()
public void	setFlags (long flags)
public boolean	checkFlags (long flags)

Method detail :

```
public long getFlags ()
```

Returns :

the flags for the specified object

```
public void setFlags (long flags)
```

Parameters :

`flags` - the new flags for the specified object

```
public boolean checkFlags (long flags)
```

Parameters :

`flags` - the flags to be checked for the specified object

IDENTIFIED

```
public interface Identified
```

The Identified interface is the very heart of the the GARGANTUA system. It sets an identification system that all objects handled by the system must follow.

The Identified interface is implemented by almost all objects within the GARGANTUA system, as all the objects handled by the server must have a unique identifier. here are two types of identifiers for each type of object : a numeric identifier and a string identifier.

The numeric identifier is used to uniquely identify objects of the same type.

Because unique numeric identifiers cannot globally and uniquely identify objects, a global identifier system was designed. the purpose of such an identifying system is, first, to uniquely identify objects throughout the system, and secondly, to be able to get a maximum amount of information about an object position within the database just be examining the object identification string. These string identifiers have always the same character length, thus enabling consistent handling, and always hold al least the object type, the database ID of the database that stores the object and the object numeric identification; additionally, some objects contain other numeric identifications such as, for a document, the numeric container folder identification.

The structure of the string identification is specific to each object and is explained for each and every one of the objects within the object's documentation

As of release 7.4 this interface extends Parsable which wraps the 'prefix' property and the 'Parsable#parse' method. These methods have been pulled up from previous version of `Identified`.

All Superinterfaces :

Parsable , PropertyEnhancer

Since :

7.0

See also :

IdFormat , Prefix

Author :

stefanu

Method summary :

Modifier and type	Method
`public java.lang.String`	`getId ()`

public void	setId (java.lang.String id)
public int	getDbId ()
public void	setDbId (int id)
public int	getNumeric (int i)
public void	setNumeric (int i, int numeric)

Method detail :

```
public String getId ()
```

Returns :

the string identification of the specified object

```
public void setId (String id)
```

Parameters :

`id` - the new string identification for the object

```
public int getDbId ()
```

Returns :

the numeric database ID that stores the specified object

```
public void setDbId (int id)
```

Parameters :

`id` - the new numeric database ID that stores the specified object

```
public int getNumeric (int i)
```

Parameters :

`i` - the numeric ID index

Returns :

the numeric ID at the specified index

```
public void setNumeric (int i, int numeric)
```

Parameters :

`i` - the numeric ID index

`numeric` - the new numeric ID for the specified index

LABELED

```
public interface Labeled
```

The Labeled interface is used by all objects that have a label, which is almost every public GARGANTUA object.

Since :

7.0

See also :

Described

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getLabel ()
public void	setLabel (java.lang.String label)

Method detail :

```
public String getLabel ()
```

Returns :

the label for the specified object

```
public void setLabel (String label)
```

Parameters :

`label` - the new label for the specified object

LIFECYCLED

```
public interface Lifecycled
```

Lifecycled objects are objects that may change automatically over time. The lifecycle of such objects may be customized by the system administrator, as well as the events that trigger a lifecycle state change. For instance, objects may be automatically archived after a specified period of inactivity or documents may be published if a certain attribute has changed to a specific value.

This interface is obsolete as of 7.8

All Superinterfaces :

Timed

Deprecated :

Author :

stefanu

MANAGED

```
public interface Managed
```

Managed objects are objects that have an associated user that acts as a manager. Such objects are, for instance, processes.

Since :

7.0

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getManagerId ()
public void	setManagerId (java.lang.String managerId)

Method detail :

```
public String getManagerId ()
```

Returns :

the manager identification for the specified object

```
public void setManagerId (String managerId)
```

Parameters :

managerId - the new manager identification for the specified object

Modified

```
public interface Modified
```

Modified objects are objects that keep track of the user identification that modified them; (viewable objects)

ModifiedBy are always physical users, never groups or roles.

All Superinterfaces :

Dated

Since :

7.10

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getModifiedBy ()
public void	setModifiedBy (java.lang.String modifiedBy)
public int	getModifiedByNumeric ()
public void	setModifiedByNumeric (int modifiedBy)

Method detail :

```
public String getModifiedBy ()
```

Returns :

the modifiedBy the user who modify the specified object

```
public void setModifiedBy (String modifiedBy)
```

Parameters :

`modifiedBy` - the user who modify the specified object

```
public int getModifiedByNumeric ()
```

Returns :

the numeric of the modifiedById

```
public void setModifiedByNumeric (int modifiedBy)
```

Parameters :

`modifiedBy` - the numeric of the modifiedById

Numbered

```
public interface Numbered
```

The Numbered interface is used by all objects that have an index within a certain context (for example any object version within object history).

Since :

7.5

Author :

andreea

Method summary :

Modifier and type	Method
public int	getNumber ()
public void	setNumber (int number)

Method detail :

```
public int getNumber ()
```

Returns :

the index of the specified object

```
public void setNumber (int number)
```

Parameters :

number - the new index for the specified object

OWNED

```
public interface Owned
```

Owned objects are objects that keep track of the user identification that created them; most objects of the GARGANTUA system are owned, since the security mechanism uses owners to give them specific rights.

Owners are always physical users, never groups or roles.

Since :

7.0

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getOwnerId ()
public void	setOwnerId (java.lang.String ownerId)
public int	getOwnerIdNumeric ()
public void	setOwnerIdNumeric (int ownerId)

Method detail :

```
public String getOwnerId ()
```

Returns :

the owner identification for the specified object

```
public void setOwnerId (String ownerId)
```

Parameters :

`ownerId` - the new owner identification for the specified object

```
public int getOwnerIdNumeric ()
```

Returns :

the numeric owner identification for the specified object

```
public void setOwnerIdNumeric (int ownerId)
```

Parameters :

`ownerId` - the new numeric owner identification for the specified object

PACKAGEABLE

```
public interface Packageable
```

The Packageable interface is used by all objects that can be grouped in a package.

Since :

7.10

See also :

Described

Author :

danieln

Method summary :

Modifier and type	Method
public java.util.List	getPackages ()
public void	setPackages (java.util.List packages)

Method detail :

```
public List getPackages ()
```

Obtains the packages ids that the object is part of

```
public void setPackages (List packages)
```

Sets the ids of packages that the object is part of

Parameters :

packages - the list of packages

PARSABLE

```
public interface Parsable
```

Wrapper for parsable objects requirements.

Since :

7.4

Author :

elf

Method summary :

Modifier and type	Method
public java.lang.String	getPrefix ()
public void	setPrefix (java.lang.String prefix)
public void	parse (java.lang.String tokens)

Method detail :

```
public String getPrefix ()
```

Returns :

returns the prefix (type) of the specified object

```
public void setPrefix (String prefix)
```

Parameters :

`prefix` - the new prefix (type) for the specified object

```
public void parse (String tokens)
```

Initialize using string parsed tokens.

Parameters :

`tokens` - string parsed tokens used for initialization when object initialized from string

PRESENTED

```
public interface Presented
```

Presented objects are objects that have presentation options. They specify which attributes must be displayed

Since :

7.0

See also :

SearchTemplate

Author :

alexis

Method summary :

Modifier and type	Method
public java.lang.String	getPresentationFileId ()
public void	setPresentationFileId (java.lang.String fileId)

Method detail :

```
public String getPresentationFileId ()
```

Returns :

the presentation file ID for the specified object

```
public void setPresentationFileId (String fileId)
```

Parameters :

`fileId` - the presentation file ID for the specified object

Prioritized

```
public interface Prioritized
```

Prioritized objects are objects that may be sorted according to a user set priority

Since :

7.0

See also :

Priority

Author :

stefanu

Method summary :

Modifier and type	Method
public int	getPriority ()
public void	setPriority (int priority)

Method detail :

```
public int getPriority ()
```

Returns :

the priority for the specified object

```
public void setPriority (int priority)
```

Parameters :

priority - the new priority for the specified object

PRIVILEGED

```
public interface Privileged
```

Privileged objects are objects that have a privileges descriptor

Since :

7.11

Author :

mariusz

REALIZED

```
public interface Realized
```

Realized objects are objects that have an associated file; it is the responsibility of the business code to store and retrieve the files from the GARGANTUA file server accordingly.

Since :

7.0

See also :

RealizedEx , com.siatel.framework.service.storage.StorageLibrary

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getFileId ()
public void	setFileId (java.lang.String fileId)

Method detail :

```
public String getFileId ()
```

Returns :

the file ID for the specified object

```
public void setFileId (String fileId)
```

Parameters :

`fileId` - the new file ID for the specified object

REALIZEDEX

```
public interface RealizedEx
```

RealizedEx objects are objects than need to keep track of the file size and file date that is stored within the GARGANTUA file server. In order to accelerate access to this information, these objects store this information internally, rather than retrieving it from the file server on a need-to-access basis. It is the responsibility of the business code to keep this information up to date and consistent with the actual file information.

All Superinterfaces :

Realized

Since :

7.0

See also :

Realized , com.siatel.framework.service.storage.StorageLibrary

Author :

stefanu

Method summary :

Modifier and type	Method
public long	getFileSize ()
public void	setFileSize (long size)
public java.lang.String	getFileType ()
public void	setFileType (java.lang.String fileType)

Method detail :

```
public long getFileSize ()
```

Returns :

the file size for the specified object

```
public void setFileSize (long size)
```

Parameters :

`size` - new the file size for the specified object

```
public String getFileType ()
```

Returns :

the file type for the specified object

```
public void setFileType (String fileType)
```

Parameters :

`fileType` - the new file type for the specified object

REFERENCED

```
public interface Referenced
```

Referenced objects are objects that must keep a reference count per actual object.

Since :

7.0

Author :

stefanu

Method summary :

Modifier and type	Method
public int	getRefCount ()
public void	setRefCount (int refCount)

Method detail :

```
public int getRefCount ()
```

Returns :

the reference count for the specified object

```
public void setRefCount (int refCount)
```

Parameters :

`refCount` - the new reference count for the specified object.

Replicable

`public interface Replicable`

Replicable objects are objects that may be copied.

Since :

7.0

Author :

stefanu

Method summary :

Modifier and type	Method
`public void`	`replicate (com.siatel.defs.Identified replica)`
`public void`	`replicateFillOnly (com.siatel.defs.Identified replica)`

Method detail :

```
public void replicate (Identified replica)
```

Parameters :

`replica` - the object that will be set to be a replica of the specified object.

```
public void replicateFillOnly (Identified replica)
```

Parameters :

`replica` - the object that will be set to be a replica of the specified object. Only null-value properties will be replicated, hence 'fillOnly'

SAMPLED

```
public interface Sampled
```

Sampled objects are objects that have an associated sample file; it is the responsibility of the business code to store and retrieve the files from the GARGANTUA file server accordingly.

Since :

7.5

See also :

com.siatel.framework.service.storage.StorageLibrary

Author :

elf

Method summary :

Modifier and type	Method
public java.lang.String	getSampleFileId ()
public void	setSampleFileId (java.lang.String fileId)

Method detail :

```
public String getSampleFileId ()
```

Returns :

the sample file ID for the specified object

```
public void setSampleFileId (String fileId)
```

Parameters :

`fileId` - the new sample file ID for the specified object

SECURABLE

`public interface Securable`

Securable interface is implemented by all objects that can have a security profile.

Since :

7.3

Author :

manuelam, elf

Method summary :

Modifier and type	Method
public java.lang.String	getSecurityProfileId ()
public void	setSecurityProfileId (java.lang.String securityProfile)
public int	getSecurityProfileIdNumeric ()
public void	setSecurityProfileIdNumeric (int securityProfileId)
public long	getSecurityMask ()
public void	setSecurityMask (long securityMask)

Method detail :

```
public String getSecurityProfileId ()
```

Accessor.

Returns :

the security profile of the specified object

```
public void setSecurityProfileId (String securityProfile)
```

Accessor.

Parameters :

`securityProfile` - the new security profile for the specified object

```
public int getSecurityProfileIdNumeric ()
```

Accessor.

Returns :

the numeric relevant part of the security profile id

```
public void setSecurityProfileIdNumeric (int securityProfileId)
```

Accessor.

Parameters :

`securityProfileId` - the numeric part of the security profile id

```
public long getSecurityMask ()
```

Accessor.

Returns :

the rendered security mask of the specified object (by connection)

```
public void setSecurityMask (long securityMask)
```

Accessor.

Parameters :

`securityMask` - the security mask for the specified object

SHARED

```
public interface Shared
```

Sharing support field accessors declarations.

Since :

7.0

Author :

victorc

Field summary :

Modifier and type	Field
public static final int	UNLOCK
public static final int	LOCK
public static final int	CHECK_OUT

Method summary :

Modifier and type	Method
public java.lang.String	getLockedBy ()
public void	setLockedBy (java.lang.String userId)
public int	getLockType ()
public void	setLockType (int lockType)
public boolean	isLocked ()
public java.util.Date	getDateL ()
public void	setDateL (java.util.Date dateL)
public java.lang.Long	getValidMillis ()
public void	setValidMillis (java.lang.Long millis)

Field detail :

```
public static final int UNLOCK = -1
```

Lock type constant defining the ‘unlocked’ object state.

```
public static final int LOCK = 0
```

Lock type constant defining the ‘locked’ object state.

```
public static final int CHECK_OUT = 1
```

Lock type constant defining the ‘exclusive lock’ object state.

Method detail :

```
public String getLockedBy ()
```

Accessor.

Returns :

the id of the user responsible for the ‘locked’ or ‘locked exclusively’ state of the object

```
public void setLockedBy (String userId)
```

Accessor.

Parameters :

`userId` - the id of the user responsible for the ‘locked’ or ‘locked exclusively’ state of the object

```
public int getLockType ()
```

Accessor.

Returns :

the lock type/state

```
public void setLockType (int lockType)
```

Accessor.

Parameters :

`lockType` - the lock type/state

```
public boolean isLocked ()
```

Accessor

Returns :

`true` if current object lock type is not {@link #UNLOCK}, `false` otherwise

```
public Date getDateL ()
```

Accessor.

Returns :

the lock date if current object lock type is not {@link #UNLOCK}

```
public void setDateL (Date dateL)
```

Accessor.

Parameters :

`dateL` - the lock date if current object lock type is not #UNLOCK

```
public Long getValidMillis ()
```

Accessor.

Returns :

the validity duration of the current object lock if lock type not {@link #UNLOCK}

```
public void setValidMillis (Long millis)
```

Accessor

Parameters :

`millis` - the validity duration of the current object lock if lock type not #UNLOCK

SIGNABLE

```
public interface Signable
```

Signable objects are objects that keep track of the user identification that signed them; (viewable objects)

SignedBy are always physical users, never groups or roles.

Since :

8.0 / 2018.07

Author :

victorc

Method summary :

Modifier and type	Method
public java.util.Date	getSignedDate ()
public void	setSignedDate (java.util.Date date)
public java.lang.String	getSignedBy ()
public void	setSignedBy (java.lang.String signedBy)
public java.lang.Integer	getSignedByNumeric ()
public void	setSignedByNumeric (int signedBy)
public java.lang.Integer	getSignatureType ()
public void	setSignatureType (java.lang.Integer sigType)

Method detail :

```
public Date getSignedDate ()
```

Returns :

the last signature date and time of the specified object

```
public void setSignedDate (Date date)
```

Parameters :

`date` - the new signature date and time of the specified object

```
public String getSignedBy ()
```

Returns :

the signedBy the user who signed the specified object

```
public void setSignedBy (String signedBy)
```

Parameters :

`signedBy` - the user who signed the specified object

```
public Integer getSignedByNumeric ()
```

Returns :

the numeric of the signedById

```
public void setSignedByNumeric (int signedBy)
```

Parameters :

`signedBy` - numeric of the signedById

```
public Integer getSignatureType ()
```

Returns :

the signature type. One of {@code SIG_TYPE_...} constants.

```
public void setSignatureType (Integer sigType)
```

Parameters :

`the` - signature type. One of {@code SIG_TYPE_...} constants. {@code null} supported.

TAGGED

```
public interface Tagged
```

Tagged objects are objects that may have a set of tags associated with them. Tags are defined by the Tag constants, and are ORed constants

Since :

7.0

See also :

Tag

Author :

stefanu

Method summary :

Modifier and type	Method
public long	getTags ()
public void	setTags (long tags)

Method detail :

```
public long getTags ()
```

Returns :

the tags for the specified object

```
public void setTags (long tags)
```

Parameters :

tags - the new tags for the specified object

TIMEBOXED

```
public interface Timeboxed
```

This interface is for objects that have a due (estimated) and an expiry (validity) duration.

Since :

7.9

Author :

victorc

Method summary :

Modifier and type	Method
public java.lang.Long	getDelayD ()
public void	setDelayD (java.lang.Long delayD)
public java.lang.Long	getDelayX ()
public void	setDelayX (java.lang.Long delayX)

Method detail :

```
public Long getDelayD ()
```

Returns :

due (estimated) delay value in millis

```
public void setDelayD (Long delayD)
```

Parameters :

`delayD` - the new due delay value in millis

```
public Long getDelayX ()
```

Returns :

expiry (validity) delay value in millis

```
public void setDelayX (Long delayX)
```

Parameters :

`delayX` - the new expiry delay value in millis

TIMEBOXEDEX

```
public interface TimeboxedEx
```

This interface is for Timeboxed objects that can specify the due (estimated) and expiry (validity) duration in a attributes specified by id.

All Superinterfaces :

Timeboxed

Since :

7.9

Author :

victorc

Method summary :

Modifier and type	Method
public java.lang.String	getDelayDAttrId ()
public void	setDelayDAttrId (java.lang.String delayDAttrId)
public java.lang.String	getDelayXAttrId ()
public void	setDelayXAttrId (java.lang.String delayXAttrId)

Method detail :

```
public String getDelayDAttrId ()
```

Returns :

due delay attribute id

```
public void setDelayDAttrId (String delayDAttrId)
```

Parameters :

`delayDAttrId` - id of attribute whose value will specify the delay

```
public String getDelayXAttrId ()
```

Returns :

expiry delay attribute id

```
public void setDelayXAttrId (String delayXAttrId)
```

Parameters :

`delayXAttrId` - id of attribute whose value will specify the delay

TIMED

```
public interface Timed
```

Wrapper for timed objects (processes, tasks) support accessors.

Since :

7.0

Author :

stefanu

Method summary :

Modifier and type	Method
public java.util.Date	getDateS ()
public void	setDateS (java.util.Date dateS)
public java.util.Date	getDateE ()
public void	setDateE (java.util.Date dateE)

Method detail :

```
public Date getDateS ()
```

Accessor.

Returns :

the start date for the specified object

```
public void setDateS (Date dateS)
```

Accessor.

Parameters :

`dateS` - the new start date for the specified object

```
public Date getDateE ()
```

Accessor.

Returns :

the end date for the specified object

```
public void setDateE (Date dateE)
```

Accessor.

Parameters :

`dateE` - dateS the new end date for the specified object

UUIDENABLED

```
public interface UUIDEnabled
```

This interface defines requirements for objects that support java.util.UUIDs as means of identification. A default support implementation is provided in UUIDDelegate. Default implementations of Folder, Document, Process support this interface through aggregating an UUIDDelegate and delegating to it.

Since :

7.8 / 2012.05

See also :

"http://en.wikipedia.org/wiki/Uuid"

Author :

victorc

Method summary :

Modifier and type	Method
public java.util.UUID	getUUID ()
public void	setUUID (java.util.UUID uuidSrc)
public java.lang.String	getUUIDString ()
public void	setUUIDString (java.lang.String uuidStr)

Method detail :

```
public UUID getUUID ()
```

Accessor.

Returns :

the object {@link java.util.UUID}

```
public void setUUID (UUID uuidSrc)
```

Accessor.

Parameters :

`uuidSrc` - the source java.util.UUID

```
public String getUUIDString ()
```

Accessor.

Returns :

the 36-character string formated object {@link java.util.UUID}. Short for {@link #getUUID()}.`toString()`.

```
public void setUUIDString (String uuidStr)
```

Accessor.

Parameters :

`uuidStr` - the source java.util.UUID in 36-character string format. Short for #setUUID(`java.util.UUID#fromString`(uuidStr))

VERSION

```
public interface Version
```

Public definition to group identified, realized and commented objects.

All Superinterfaces :

Identified , Realized , Commented

Since :

7.5

Author :

mariusz

Versioned

```
public interface Versioned
```

Versioned objects are objects for which the database keeps track of previous versions. Versions are always represented as a long, but actually they keep track of two components : the major version number and the minor version number; these numbers are meant to be used as with software versions : the major number is incremented once some significant modifications are made to the object, while the minor version number is incremented if non-significant modifications are made.

When versioned objects are created the version number is set to 1.0; the user interface does not display 1.0 version numbers, as these are the defaults. Some objects may use the version number 0 (zero) as a default 1.0 version number.

Since :

7.0

See also :

VersionedEx

Author :

stefanu

Method summary :

Modifier and type	Method
public long	getVersion ()
public void	setVersion (long version)

Method detail :

```
public long getVersion ()
```

Accessor.

Returns :

the version number for the specified object

```
public void setVersion (long version)
```

Accessor.

Parameters :

`version` - the new version number for the specified object

VIEWABLE

```
public interface Viewable
```

Viewable objects are objects that have a set of user-defined attributes. The attributes are set using forms, and may be explicit or default attributes. Default attributes are set by the business components at runtime according to the current running context.

Since :

7.0

See also :

Attribute , Form , Type

Author :

stefanu

Method summary :

Modifier and type	Method
public java.lang.String	getFormId ()
public void	setFormId (java.lang.String formId)
public int	getFormIdNumeric ()
public void	setFormIdNumeric (int formId)
public java.util.List	getAttributes ()
public java.util.List	getCustomAttributes ()
public void	setAttributes (java.util.List attributes)
public java.lang.String	getAttributeValue (java.lang.String attributeId, java.util.Locale locale)
public void	setAttributeValue (java.lang.String attributeId, java.util.Locale locale, java.lang.String value)
public java.lang.Object	getAttributeValue (java.lang.String attributeId)
public java.lang.Object	getAttributeValue (java.lang.String attributeId, java.lang.Object defaultValue, java.util.Locale locale)
public void	setAttributeValue (java.lang.String attributeId, java.lang.Object value)

public java.util.List	setDefaultAttributes (java.util.List attributes, boolean bForced)
public java.util.List	getDefaultAttributes (java.util.List attributes)
public int	getIcon ()
public void	setIcon (int icon)

Method detail :

```
public String getFormId ()
```

Accessor.

Returns :

the form identification for the specified object

```
public void setFormId (String formId)
```

Accessor.

Parameters :

`formId` - the new form identification for the specified object

```
public int getFormIdNumeric ()
```

Accessor.

Returns :

the numeric form identification for the specified object

```
public void setFormIdNumeric (int formId)
```

Accessor.

Parameters :

`formId` - the new numeric form identification for the specified object

```
public List getAttributes ()
```

Accessor.

Returns :

the list of attributes for the specified object; these will always include the default attributes

```
public List getCustomAttributes ()
```

Accessor.

Returns :

the list of attributes for the specified object; these will always exclude the default attributes

```
public void setAttributes (List attributes)
```

Accessor.

Parameters :

`attributes` - the new attributes for the specified object; default attributes are always filtered out, as they are generally not user modifiable

```
public String getAttributeValue (String attributeId, Locale locale)
```

Accessor.

Parameters :

`attributeId` - the sttribute identification

`locale` - the i18n locale identifier for the current running context

Returns :

the value for the specified attribute, using the specified locale, for the specified object.

```
public void setAttributeValue (String attributeId, Locale locale, String
value)
```

Accessor. Sets the attribute value for the specified attribute, using the specified locale identifier, for the specified object.

Parameters :

`attributeId` - the attribute identification

`locale` - the i18n locale identifier for the current running context

`value` - the attribute value

```
public Object getAttributeValue (String attributeId)
```

Accessor. Retrieves the attribute value; no locale conversion is applied

Parameters :

`attributeId` - the attribute identification

Returns :

the attribute value

```
public Object getAttributeValue (String attributeId, Object
defaultValue, Locale locale)
```

Accessor. Retrieves the attribute value; no locale conversion is applied, default value can be supplied

Parameters :

`attributeId` - the attribute identification

`defaultValue` - default value to be used when everything else fails

`locale` - the i18n locale identifier for the current running context

Returns :

the attribute value

```
public void setAttributeValue (String attributeId, Object value)
```

Accessor. Sets the attribute value for the specified object; no locale conversion is applied.

Parameters :

`attributeId` - the attribute identifier

`value` - the attribute value

```
public List setDefaultAttributes (List attributes, boolean bForced)
```

Modifies the input list of attributes, setting the default attribute values for the specified object.

Internal use only

Parameters :

`attributes` - the list of object attributes for the specified object

Returns :

the modified list of attributes, with all default attributes set.

```
public List getDefaultAttributes (List attributes)
```

Internal use only.

Parameters :

`attributes` -

Returns :

list of attributes

```
public int getIcon ()
```

Accessor.

Returns :

the icon index for the specified object

```
public void setIcon (int icon)
```

Accessor.

Parameters :

`icon` - the new icon index for the specified object

ViewableEx

```
public interface ViewableEx
```

ViewableEx extends the Viewable support by adding form page accessibility.

All Superinterfaces :

Viewable

Since :

7.0

See also :

Viewable , Form

Author :

victorc

Method summary :

Modifier and type	Method
public java.lang.String	getFormPage ()
public void	setFormPage (java.lang.String formPage)

Method detail :

```
public String getFormPage ()
```

Accessor.

Returns :

the form apge

```
public void setFormPage (String formPage)
```

Accessor.

Parameters :

formPage - the new form page

PACKAGE COM.SIATEL.DOMAIN

This package contains the domain objects (data structures).

INTERFACES

ARCHIVECOMMENT

```
public interface ArchiveComment
```

The archive version of the comment object. Archive comments are attached to archived documents and folders;

The comment identification structure :

ID component	Value	Description
Prefix	ACDF, ACMF	See Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder ID	numeric	The folder container numeric identification
Pin ID	numeric	The numeric pin ID, if the document is pinned; zero otherwise
Document ID	numeric	The numeric document ID if the comment is on a document; zero otherwise
Comment ID	numeric	The comment numeric identification

All Superinterfaces :

Comment , Archive

Author :

adinam

ArchiveDocument

```
public interface ArchiveDocument
```

The archive version of the document object.

The archive document identification structure :

ID component	Value	Description
Prefix	ADCF, AFCF, ARDF, ARFF	See Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder ID	numeric	The folder container numeric identification
Pin ID	numeric	The numeric pin ID, if the document is pinned; zero otherwise
Document ID	numeric	The numeric document ID if the comment is on a document; zero otherwise
Comment ID	numeric	The comment numeric identification

All Superinterfaces :

Document , Archive

Author :

adinam

ARCHIVEDOCUMENTVERSION

```
public interface ArchiveDocumentVersion
```

The archived document version

The document version identification structure :

ID component	Value	Description
Prefix	VNDF, VNDP	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder or process ID	numeric	The folder or process container numeric identification
Pin ID	numeric	The numeric pin ID, if the document is pinned; zero otherwise
Document ID	numeric	The numeric document ID
Version ID	numeric	The numeric version ID

All Superinterfaces :

DocumentVersion , Archive

Author :

stefanu

ArchiveFolder

```
public interface ArchiveFolder
```

The archive version of a folder.

The archive folder identification structure :

ID component	Value	Description
Prefix	ARFL	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder ID	numeric	The numeric folder ID
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Folder , Archive

Author :

stefanu

ARCHIVENOTE

```
public interface ArchiveNote
```

The assignment identification structure :

ID component	Value	Description
Prefix	ARFL	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder ID	numeric	The numeric folder ID
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Note , Archive

Author :

stefanu

ASSIGNMENT

```
public interface Assignment
```

General assignment object; holds a source and destination identifier; used, for instance, to represent user assignments to groups. The assignment identification structure :

ID component	Value	Description
Prefix	ASGU, ASUR, ASGR, ASUD, ASRD, ASGD, ASUP, ASRP, ASGP, ASFA, ASUV, ASRV, ASGV, ASUF, ASGF, ASRF, ASUP, ASGP, ASRP, ASOP, ASFP, ASUT, ASGT, ASRT	The assignment object prefix; the prefix depends on the source and target object, but always starts with the letters “AS”; see com.siatel.domain.Prefix for more information on prefixes
Database ID	0	Always set to zero; the database identification is not required for these objects
Source ID	numeric	The numeric ID of the source object
Destination ID	numeric	The numeric ID of the destination object
Not used	0	Always set to zero
Not used	0	Always set to zero

All Superinterfaces :

Identified , Cloneable , Replicable

Author :

stefanu

Method summary :

Modifier and type	Method
public int	getDestIdNumeric ()
public void	setDestIdNumeric (int destId)
public int	getSourceIdNumeric ()
public void	setSourceIdNumeric (int sourceId)
public java.lang.Object	getData ()
public void	setData (java.lang.Object data)

Method detail :

```
public int getDestIdNumeric ()
```

Returns :

the destination numeric identification

```
public void setDestIdNumeric (int destId)
```

Parameters :

`destId` - the new destination numeric identification

```
public int getSourceIdNumeric ()
```

Returns :

the source numeric identification

```
public void setSourceIdNumeric (int sourceId)
```

Parameters :

`sourceId` - the new source numeric identification

```
public Object getData ()
```

Returns :

custom assignment data

```
public void setData (Object data)
```

Parameters :

`data` - custom assignment data

ATTRIBUTE

`public interface Attribute`

Attributes are assigned to objects that are Viewable. See the Viewable interface for information on how to retrieve and manipulate the GARGANTUA object attributes.

The attribute identification structure :

ID component	Value	Description
`Prefix`	`DFAT`	See com.siatel.domain.Prefix for more information on prefixes
`Database ID`	`numeric`	The database numeric ID for the GARGANTUA database that stores the object
`Attribute ID`	`numeric`	Numeric ID of the attribute within the specified database
`Not used`	`0`	-
`Not used`	`0`	-
`Not used`	`0`	-

All Superinterfaces :

Identified , Labeled , Owned , Described , Flagged , Dated , Cloneable , Replicable , Packageable , Cacheable

Author :

stefanu

CERTIFICATE

```
public interface Certificate
```

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Described , Dated , Realized , Deployable , Replicable

Author :

olga

CLASSIFICATIONELEMENT

```
public interface ClassificationElement
```

Classification elements are used by GARGANTUA Views to create a virtual folder hierarchy based on attribute values.

The classification element identification structure :

ID component	Value	Description
Prefix	CELM	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database identification

All Superinterfaces :

Identified , Labeled , Cloneable , Replicable , Flagged

Author :

stefanu

CLASSIFICATIONTREEELEMENT

```
public interface ClassificationTreeElement
```

Classification elements are used by GARGANTUA Views to create a virtual folder hierarchy based on attribute values.

The classification element identification structure :

ID component	Value	Description
Prefix	CELM	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database identification

All Superinterfaces :

Identified , Labeled , Cloneable , Replicable , Flagged

Author :

stefanu

COMMENT

```
public interface Comment
```

The Comment object. Comments can be attached to documents ,folders or processes;

The comment identification structure :

ID component	Value	Description
Prefix	CMDF, CMDP , CMTF , CMTP	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder or process ID	numeric	The folder or process container numeric identification
Pin ID	numeric	The numeric pin ID, if the document is pinned; zero otherwise
Document ID	numeric	The numeric document ID if the comment is on a document; zero otherwise
Comment ID	numeric	The comment numeric identification

All Superinterfaces :

Labeled , Identified , Owned , Dated , Flagged , Replicable , Deletable

Author :

adinam

DATABASE

```
public interface Database
```

The 'database' concept.

The database identification structure :

ID component	Value	Description
Prefix	DATA	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Not used	0	-
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Dated , Described , Flagged , Cloneable , Replicable

Since :

15/03/2007

Author :

Victor Citiriga

Dependency

```
public interface Dependency
```

A **dependency** relationship concept definition. It specifies

- a **SOURCE** - the dependant, one who depends on another
- a **TARGET** - the dependency, one on which others depend

The intended usage of this concept is to cover the following common requirements:

- the **SOURCE** depends on the **TARGET** and so the **TARGET** may be required to be **LOCKED** for certain operations (change, deletion, etc)
- provide information on dependency relationships that prevent certain operations from taking place.

The requirements above have been extended in order to support basic-typed custom fields in which, based on specific usage requirements the user may store dependency-relevant information

All Superinterfaces :

Flagged

Since :

02/02/2010

Author :

victorc

DISTRIBUTIONLISTENTRY

```
public interface DistributionListEntry
```

General distribution list object; holds all information needed to to represent an entry of distribution list
The distribution list entry identification structure :

ID component	Value	Description
Prefix	DEDF, DEPF, DEFL	The distribution list entry object prefix; the prefix depends on the object, but always starts with the letters “DE”; see com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
object ID	numeric	The numeric ID of the object
accessLevel ID	numeric	The numeric ID of the access level object
actor ID	numeric	The numeric ID of the actor object (can be user,group,role,attribute)
actor type	numeric	The category type of the actor (can be user,group,role,userlist,groupList,roleList)

All Superinterfaces :

Identified , Labeled , Flagged , Timed , Cloneable , Replicable

Document

```
public interface Document
```

The Gargantua document.

The document identification structure :

ID component	Value	Description
Prefix	DOCF, DOCP, PINF, PINP, FCHF, FCHP	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder or process ID	numeric	The folder or process container numeric identification
Pin ID	numeric	The numeric pin ID, if the document is pinned; zero otherwise
Document ID	numeric	The numeric document ID
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , DatedEx , Flagged , ViewableEx , RealizedEx , Versioned , Lifecycled , Shared , Cloneable , Replicable , Securable , UUIDEnabled , Tagged , Deletable , Modified , Signable

Author :

stefanu

DOCUMENTVERSION

```
public interface DocumentVersion
```

The GARGANTUA document version

The document version identification structure :

ID component	Value	Description
Prefix	AVDF	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder or process ID	numeric	The folder or process container numeric identification
Pin ID	numeric	The numeric pin ID, if the document is pinned; zero otherwise
Document ID	numeric	The numeric document ID
Version ID	numeric	The numeric version ID

All Superinterfaces :

Identified , Labeled , Owned , DatedEx , Flagged , RealizedEx , Versioned , Cloneable , Replicable , Deletable

Author :

stefanu

ExpirableDownloadLink

```
public interface ExpirableDownloadLink
```

The expirable download link is a link that has either a maximum access times or a maximum validity time.

The expirable download link identification structure :

ID component	Value	Description
Prefix	EXDL	The expirable download link prefix
Database ID	0	The database ID that stores the definition; since the links are stored in the administration database, this is always set to zero
User ID	numeric	The numeric ID of the expirable download link
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero

All Superinterfaces :

Identified , Labeled , Flagged , Dated , Replicable , Cloneable , Owned

Author :

mariusz

ExtendedProperty

```
public interface ExtendedProperty
```

This interface represents the definition for a extended property. The extended property holds extra-data for an object and is characterized by a label, type and value.

The structure for any extended property is as follows:

ID component	Value	Description
Prefix	EXPR	The extended property prefix
Database ID	0	The database ID that stores the extened property definition; since the properties are stored in the administration database, this is always set to zero
Extended property ID	numeric	The numeric ID of the extended property object
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero

All Superinterfaces :

Identified , Labeled , Flagged , Described , Dated , Replicable , Cloneable

Author :

mariusz

Folder

```
public interface Folder
```

The Folder object

The assignment identification structure :

ID component	Value	Description
Prefix	FLDR	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Folder ID	numeric	The numeric folder ID
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Flagged , DatedEx , ViewableEx , Lifecycled , Cloneable , Replicable , Securable , UUIDEnabled , Deletable , Modified

Author :

stefanu

FORM

```
public interface Form
```

The Form object. Forms are used by users to assign attributes to objects

The form identification structure :

ID component	Value	Description
Prefix	DFFR	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Not used	0	-
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , DeployableEx , Realized , Cloneable , Replicable , Lifecycle , Layouted , HierarchyCreator , Packageable , Cacheable

Author :

stefanu

Grid

```
public interface Grid
```

The grid object.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Replicable , Realized , Cloneable , Packageable , Cacheable

See also :

Prefix#GRID for id structure

GROUP

```
public interface Group
```

The Group object. Groups statically group users that share the same access rights on GARGANTUA objects.

The group identification structure :

ID component	Value	Description
Prefix	GROU	See com.siatel.domain.Prefix for more information on prefixes
Not used	0	-
Group ID	numeric	The numeric identification of the group object
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Described , Dated , Replicable , Cloneable , Cacheable , Privileged

Author :

stefanu

HierarchyCreator

```
public interface HierarchyCreator
```

This interface is implemented by objects that generate a custom hierarchy after creation

Since :

7.10

Author :

mariusz

INTERVENTIONREPORT

```
public interface InterventionReport
```

The intervention report indicates an intervention done on the GARGANTUA server.

The intervention report object identification structure :

ID component	Value	Description
Prefix	INRE	The intervention report object prefix
Database ID	0	The database ID that stores the intervention report definition; since the delegations are stored in the administration database, this is always set to zero
Intervention ID	numeric	The numeric ID of the intervention report object
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero

All Superinterfaces :

Identified , Labeled , Flagged , Described , Dated , Replicable , Cloneable , Realized

Author :

stefanu

IsJob

```
public interface IsJob
```

The IsJob object.

The form identification structure :

ID component	Value	Description
Prefix	ISJB	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
IsJob ID	numeric	The IS Job numeric identification
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Realized , DeployableEx , Cloneable , Replicable , Packageable , Cacheable

Author :

manuelam

LAYOUTED

```
public interface Layouted
```

The layouter interface is implemented by objects that have a custom presentation (layout), usually a fragment

Since :

7.10

Author :

mariusz

Lifecycle

```
public interface Lifecycle
```

Lifecycle enabling objects requirements.

Author :

elf

MANAGER

```
public interface Manager
```

This interface wraps the model requirements of the Manager object.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Described , Replicable , Cloneable

Since :

R562

Author :

victorc

ModelOCR

```
public interface ModelOCR
```

The ModelOCR object.

The form identification structure :

ID component	Value	Description
Prefix	MOCR	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
ModelOCR ID	numeric	The Model OCR numeric identification
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Realized , DeployableEx , Sampled , Cloneable , Replicable , Packageable , Cacheable

Author :

manuelam

NOTE

```
public interface Note
```

The Note object. Notes can be attached to documents or folders; the note's content is stored as file, thus being indexed in the fulltext database, if available. Depending on the parent object, notes may have a rich text format (in the case of folders)

The note identification structure :

ID component	Value	Description
`Prefix`	`NTDF, NTDP`	See com.siatel.domain.Prefix for more information on prefixes
`Database ID`	`numeric`	The database ID
`Folder or process ID`	`numeric`	The folder or process container numeric identification
`Pin ID`	`numeric`	The numeric pin ID, if the document is pinned; zero otherwise
`Document ID`	`numeric`	The numeric document ID
`Note ID`	`numeric`	The note numeric identification

All Superinterfaces :

Identified , Owned , DatedEx , Realized , Replicable , Cloneable , Deletable

Author :

stefanu

ObjectExtendedProperty

```
public interface ObjectExtendedProperty
```

This interface represents the definition for a assignment of a extended property to a object.

The structure for any extended property is as follows:

ID component	Value	Description
Prefix	ASOP	The extended property prefix
Database ID	0	The database ID that stores the extened property definition; since the properties are stored in the administration database, this is always set to zero
Extended property ID	numeric	The numeric ID of the extended property object
Assignment id	numeric	The numeric ID of the assignment
Extended property id	numeric	The numeric ID of the extended property
Object id	numeric	The numeric ID of the object
Not used	0	Always set to zero

All Superinterfaces :

Assignment

Author :

mariusz

PACKAGE

```
public interface Package
```

The Package object

The assignment identification structure :

ID component	Value	Description
Prefix	PCKG	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Package ID	numeric	The numeric package ID
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Described , Dated , Flagged , Owned , Cloneable , Replicable

Author :

mariusz

Planner

`public interface Planner`

This interface declares the structure of the planner object. A planner consists of a weekly timetable and list of exceptions from it.

In addition to the accessors defined by the inherited interfaces, this interfaces defines accessors to the weekly timetable.

`'Timetable'` is an array of 7 long values. Each element in the array corresponds to a day in the week, 0 representing monday, 6 representing sunday. For each day, the least significant 48 bits represent half-hour intervals. Bit 0 corresponds to 0:00-0:30, 1 to 0:30-1:00, 47 to 23:30-0:00. If a bit is set then it represents activity time, if it is reset then it represents inactivity time.

The planner identification structure :

ID component	Value	Description
`Prefix`	`PLAN`	See com.siatel.domain.Prefix for more information on prefixes
`Not used`	`0`	-
`Planner ID`	`numeric`	The numeric identification of the planner object
`Not used`	`0`	-
`Not used`	`0`	-
`Not used`	`0`	-

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Deployable , Cloneable , Replicable

Since :

27/03/2008

Author :

victorc

See also :

PlannerDayImpl

PLANNERDAY

```
public interface PlannerDay
```

This interface declares the structure of an exception from a planner's weekly timetable.

The numeric part of a planner day object will be represented by the numeric evaluation of the date string in YYYYMMDD format. E.g. the full planner day for the planner with numeric id 1 and date 2000/01/01 will be obtained by this code:

```
String dayId = IdFormat.toString(Prefix.PANNER_DAY, 0, 1, 20000101);
```

All Superinterfaces :

Identified , Cloneable , Replicable

Since :

27/032008

Author :

victorc

See also :

Planner

PORTAL

```
public interface Portal
```

The Portal object

The portal identification structure :

ID component	Value	Description
Prefix	PORT	See com.siatel.domain.Prefix for more information on prefixes
Not used	0	-
Portal ID	numeric	The portal numeric identification
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Described , Realized , Flagged , Dated , Cloneable , Replicable

Author :

stefanu

PROCESS

```
public interface Process
```

The Process object

The process identification structure :

ID component	Value	Description
Prefix	PROC	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Process ID	numeric	The process numeric identification
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , DatedEx , Flagged , Realized , Prioritized , Managed , Referenced , ViewableEx , Timed , Expirable , Replicable , Cloneable , Securable , UUIDEnabled , Modified , FlowEnabled

Author :

stefanu

ProcessDefinition

```
public interface ProcessDefinition
```

The Process definition object

The process definition identification structure :

ID component	Value	Description
Prefix	DFPR	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Definition ID	numeric	The process definition numeric identification
Not used	0	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Dated , Described , Managed , Prioritized , Flagged , Realized , DeployableEx , Replicable , Referenced , Cloneable , Securable , TimeboxedEx , Packageable , Cacheable , FlowEnabled

Author :

stefanu

PROCESSDEFINITIONVERSION

```
public interface ProcessDefinitionVersion
```

The Process definition version object

The process definition version identification structure :

ID component	Value	Description
Prefix	DFPR	See com.siatel.domain.Prefix for more information on prefixes
Database ID	numeric	The database ID
Definition ID	numeric	The process definition numeric identification
Definition Version ID	The process definition version numeric identification	-
Not used	0	-
Not used	0	-

All Superinterfaces :

Identified , Labeled , Owned , Dated , Described , Flagged , Realized , DeployableEx , Replicable , Referenced , Cloneable , Securable , Version , Numbered , TimeboxedEx , Managed , Prioritized , FlowEnabled

Author :

stefanu

Resource

```
public interface Resource
```

The resource object.

All Superinterfaces :

Identified , Labeled , Described , Flagged , Owned , Dated , Realized , Cloneable , Replicable , Cacheable

See also :

Prefix#RESOURCE

ROLE

```
public interface Role
```

The role object.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Described , Dated , Replicable , Cloneable , Cacheable , Privileged

See also :

Prefix#ROLE for id structure , Privilege for privilege definitions

Rule

```
public interface Rule
```

The storage rule object interface.

The object links a database to a storage area based on a list of conditions. The rule priority is used to enforce an evaluation order. Higher priority is specified as lower integer value.

All Superinterfaces :

Identified , Cloneable , Replicable , Flagged , Cacheable

See also :

Prefix#DATABASE_STORAGE_RULE

SCRIPT

```
public interface Script
```

The 'script' concept definition. This concept groups script code (stored in a file) with script meta-data (lang - the programming language, type - the execution context/layer) and Siatel-object identification naming and common metadata (dates, flags etx).

All Superinterfaces :

Identified , Labeled , Owned , Dated , Flagged , Described , Deployable , Realized , Serializable , Cloneable , Replicable

Author :

victorc

STORAGE

```
public interface Storage
```

The storage area object interface.

All Superinterfaces :

Identified , Labeled , Owned , Described , Flagged , Dated , Cloneable , Replicable , Cacheable

See also :

Prefix#STORAGE_AREA

TASK

```
public interface Task
```

The task object.

All Superinterfaces :

Identified , Labeled , Owned , DatedEx , Described , Flagged , Timed , Expirable , Managed , Prioritized , Replicable , Cloneable , FlowEnabled

See also :

Prefix#TASK for id structure

Type

```
public interface Type
```

The type object.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Replicable , Realized , Cloneable , Packageable , Cacheable

See also :

Prefix#TYPE for id structure

USER

```
public interface User
```

The user interface specifies the behavior of user objects. A user is a physical person that interacts with the system; a user has a password that is used in order to authenticate within the system.

The user may belong to groups and may be assigned to certain roles by the system administrator

The user identification structure :

ID component	Value	Description
Prefix	USER	The user object prefix
Database ID	0	The database ID that stores the user definition; since the users are stored in the administration database, this is always set to zero
User ID	numeric	The numeric ID of the user object
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Described , Dated , Replicable , Cloneable , Cacheable , Privileged

Author :

stefanu

UserDelegation

```
public interface UserDelegation
```

The user delegation interface specifies the behavior of user delegation objects. A user can delegate his work to another user during his absence. See the specfile DELEGATE for more details.

The user delegation object identification structure :

ID component	Value	Description
Prefix	DLGT	The user delegation object prefix
Database ID	0	The database ID that stores the user delegation definition; since the delegations are stored in the administration database, this is always set to zero
Delegation ID	numeric	The numeric ID of the user delegation object
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero
Not used	0	Always set to zero

All Superinterfaces :

Identified , Owned , Flagged , Described , Dated , Timed , Replicable , Cloneable

Author :

stefanu

VIEW

```
public interface View
```

This interface defines the 'view' concept and groups concept definition data accessors.

A view consists of a hierarchy of 'view elements'. A view WILL NOT usable until deployed. The view deployment process MUST validate the 'view element' hierarchy and compile database queries for each of the view elements it contains.

Contained view elements will start with a parent id specifying 0 in it's numeric id part. Example:

DFVE00010000000100000000000000000000000 is the parent id for root view elements of the DFVW00010000000100000000000000000000000 view.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Deployable , Replicable , Cloneable , Packageable , Cacheable

Since :

12/07/2007

Author :

vic

See also :

com.siatel.domain.ViewElement , com.siatel.defs.Identified , com.siatel.defs.Labeled , com.siatel.defs.Owned , com.siatel.defs.Flagged , com.siatel.defs.Dated , com.siatel.defs.Described , com.siatel.defs.Deployable , com.siatel.defs.Replicable , java.lang.Cloneable

ViewElement

```
public interface ViewElement
```

This interface defines the 'view element' concept and groups concept definition data accessors.

A 'view element' consists of the following:

1. the parent id (view elements are organzied hierarchically)
2. a domain filter specification
3. a unicity rule specification
4. a list of name tokens
5. a sort order specification

The **parent id** specifies the id of another view element. All view elements MUST have a parent id set. For view elements defined directly in the view root the numeric part of the parent id is 0.

The **domain filter** specifies a list of filter criteria that restricts the values domain for the attributes participating in generating this element's classes and the children classes generated by this element's children. If more than one criterion is specified then criteria are combined using 'AND' logic operator. **No more than 3 filter criteria can be specified by a view item.**

The **unicity rule** consists of a list if unicity criteria, each of them specifying what attribute or attribute part participates in generating a class. If more than one criterion is specified then the combination of all criteria specified must render to a unique set. **No more than 3 unicity criteria can be specified by a view item.**The rule will be ignored for view elements that specify `IS_OBJECTS` in their `flags` member (see below for more flags).

The **name tokens** must be specified in the order they will appear in the generated classes name. Recognizable and renderable tokens are declared in this interface by the `TOKEN_UNIQUE` constants. The number in the name of the constant specifies which criterion from the unique criteria list to render in the class name. Additional tokens are declared in the `ValueToken` utility class. Plain text tokens can be specified and they will render identically in the generated class name.

The **sort order** refers to the order in which the generated classes are enumerated. It is specified by the 'flags' member of this class by the `Flag.ORDER` flags grouped by `Flag.CLASS_SORT_MASK`. The number of the flag name specifies to which of the unicity criteria the sort order applies.

The hierachical strucure of the view elements will result in the fact that the domain filters and unique criteria of a view element will act as additional restrictions for it's children when generating the children classes.

Additional flags MUST be specified in the `flags` member of this interface in `Flag.CLASSES_MASK` range. See below table for allowed combinations.

Combination	Significance
FOR_FOLDERS	Specifies that the classes to which this view element evaluates are based on folders attributes only.
FOR_FOLDERS | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual folders. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for folders not for classes when instantiating this view element.
FOR_PROCESSES	Specifies that the classes to which this view element evaluates are based on processes attributes only.
FOR_PROCESSES | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual processes. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for processes not for classes when instantiating this view element.
FOR_FOLDERS | FOR_PROCESSES	Specifies that the classes to which this view element evaluates are based on folders and processes attributes only.
FOR_FOLDERS | FOR_PROCESSES | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual folders and processes. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for folders and processes not for classes when instantiating this view element.
FOR_DOCUMENTS	Specifies that the classes to which this view element evaluates are based on documents attributes only.
FOR_DOCUMENTS | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual documents. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for documentts not for classes when instantiating this view element.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Replicable , Cloneable

Since :

12/07/2007

See also :

com.siatel.domain.View , com.siatel.domain.FilterCriterion , com.siatel.domain.UniqueCriterion , com.siatel.domain.Flag , com.siatel.domain.ValueToken , com.siatel.defs.Identified , com.siatel.defs.Labeled , com.siatel.defs.Owned , com.siatel.defs.Flagged , com.siatel.defs.Dated , com.siatel.defs.Described , com.siatel.defs.Replicable , java.lang.Cloneable

Author :

vic

ViewElementVersion

```
public interface ViewElementVersion
```

This interface defines the 'view element version' concept and groups concept definition data accessors.

A 'view element' consists of the following:

1. the parent id (view elements are organized hierarchically)
2. a domain filter specification
3. a unicity rule specification
4. a list of name tokens
5. a sort order specification

The **parent id** specifies the id of another view element. All view elements MUST have a parent id set. For view elements defined directly in the view root the numeric part of the parent id is 0.

The **domain filter** specifies a list of filter criteria that restricts the values domain for the attributes participating in generating this element's classes and the children classes generated by this element's children. If more than one criterion is specified then criteria are combined using 'AND' logic operator. **No more than 3 filter criteria can be specified by a view item.**

The **unicity rule** consists of a list if unicity criteria, each of them specifying what attribute or attribute part participates in generating a class. If more than one criterion is specified then the combination of all criteria specified must render to a unique set. **No more than 3 unicity criteria can be specified by a view item.**The rule will be ignored for view elements that specify `IS_OBJECTS` in their `flags` member (see below for more flags).

The **name tokens** must be specified in the order they will appear in the generated classes name. Recognizable and renderable tokens are declared in this interface by the `TOKEN_UNIQUE` constants. The number in the name of the constant specifies which criterion from the unique criteria list to render in the class name. Additional tokens are declared in the `ValueToken` utility class. Plain text tokens can be specified and they will render identically in the generated class name.

The **sort order** refers to the order in which the generated classes are enumerated. It is specified by the 'flags' member of this class by the `Flag.ORDER` flags grouped by `Flag.CLASS_SORT_MASK`. The number of the flag name specifies to which of the unicity criteria the sort order applies.

The hierachical strucure of the view elements will result in the fact that the domain filters and unique criteria of a view element will act as additional restrictions for it's children when generating the children classes.

Additional flags MUST be specified in the `flags` member of this interface in `Flag.CLASSES_MASK` range. See below table for allowed combinations.

Combination	Significance
FOR_FOLDERS	Specifies that the classes to which this view element evaluates are based on folders attributes only.
FOR_FOLDERS | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual folders. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for folders not for classes when instantiating this view element.
FOR_PROCESSES	Specifies that the classes to which this view element evaluates are based on processes attributes only.
FOR_PROCESSES | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual processes. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for processes not for classes when instantiating this view element.
FOR_FOLDERS | FOR_PROCESSES	Specifies that the classes to which this view element evaluates are based on folders and processes attributes only.
FOR_FOLDERS | FOR_PROCESSES | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual folders and processes. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for folders and processes not for classes when instantiating this view element.
FOR_DOCUMENTS	Specifies that the classes to which this view element evaluates are based on documents attributes only.
FOR_DOCUMENTS | IS_OBJECTS	Specifies that the classes to which this view element evaluates are actual documents. The unique criteria and name tokens will be ignored in this case. Persistence call will be made for documentts not for classes when instantiating this view element.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Replicable , Cloneable

Since :

12/07/2007

See also :

com.siatel.domain.View , com.siatel.domain.FilterCriterion , com.siatel.domain.UniqueCriterion , com.siatel.domain.Flag , com.siatel.domain.ValueToken , com.siatel.defs.Identified , com.siatel.defs.Labeled , com.siatel.defs.Owned , com.siatel.defs.Flagged , com.siatel.defs.Dated , com.siatel.defs.Described , com.siatel.defs.Replicable , java.lang.Cloneable , com.siatel.domain.ViewVersion

Author :

vic

ViewVersion

```
public interface ViewVersion
```

This interface defines the ‘view version’ concept and groups concept definition data accessors.

All Superinterfaces :

Identified , Labeled , Owned , Flagged , Dated , Described , Deployable , Replicable , Cloneable , Commented , Numbered

Since :

12/07/2007

Author :

vic

See also :

com.siatel.domain.View , com.siatel.domain.ViewElement , com.siatel.defs.Identified , com.siatel.defs.Labeled , com.siatel.defs.Owned , com.siatel.defs.Flagged , com.siatel.defs.Dated , com.siatel.defs.Described , com.siatel.defs.Deployable , com.siatel.defs.Replicable , java.lang.Cloneable

CLASSES

ATTRIBUTEMODIFIER

```
public class AttributeModifier
```

This utility class defines attribute modifiers used by view elements unique criteria.

Since :

12/07/2007

Author :

vic

Constructor summary :

Constructor
AttributeModifier ()

Constructor detail :

```
public AttributeModifier ()
```

AttributeProp

```
public class AttributeProp
```

Attribute properties; depending on the attribute type, several properties can be set

Author :

cristi

Constructor summary :

Constructor
AttributeProp ()

Constructor detail :

```
public AttributeProp ()
```

ATTRIBUTES

```
public class Attributes
```

This utility class defines default attribute constants.

Since :

12/07/2007

Author :

stefanu

Constructor summary :

Constructor
Attributes ()

Constructor detail :

```
public Attributes ()
```

CONSTANTS

```
public class Constants
```

Various constants used across the GARGANTUA system.

All Superinterfaces :

Bits

Author :

stefanu, victorc

Field summary :

Modifier and type	Field
public static final int	ANONYMOUS_NUMERIC
public static final int	UNKNOWN_NUMERIC
public static final int	SYSTEM_NUMERIC

public static final int	EVERYONE_NUMERIC
public static final int	OWNER_NUMERIC
public static final int	DBA_NUMERIC
public static final int	SUPER_NUMERIC
public static final String	ANONYMOUS
public static final String	UNKNOWN
public static final String	SYSTEM
public static final String	OWNER
public static final String	EVERYONE
public static final String	DBA
public static final String	SUPER
public static final String	DOC_TYPE_TEXT
public static final String	DOC_TYPE_IMAGE
public static final String	DOC_TYPE_OTHER
public static final String	DOC_TYPE_UNKNOWN
public static final String	DOC_TYPE_WATERMARK
public static final String	LANG_EN
public static final String	LANG_FR
public static final	LANG_ES

String	
public static final String	LANG_AR
public static final String	LANG_RO
public static final String	LANG_HU
public static final String	LANG_RU
public static final String	LANG_TR
public static final String	LANG_EL
public static final String	LANG_EN_CODE
public static final String	LANG_FR_CODE
public static final String	LANG_ES_CODE
public static final String	LANG_AR_CODE
public static final String	LANG_RO_CODE
public static final String	LANG_HU_CODE
public static final String	LANG_RU_CODE
public static final String	LANG_TR_CODE
public static final String	LANG_EL_CODE
public static final String	LANG_EN_TRANSLATED
public static final String	LANG_FR_TRANSLATED
public static final String	LANG_ES_TRANSLATED

public static final String	LANG_AR_TRANSLATED
public static final String	LANG_RO_TRANSLATED
public static final String	LANG_HU_TRANSLATED
public static final String	LANG_RU_TRANSLATED
public static final String	LANG_TR_TRANSLATED
public static final String	LANG_EL_TRANSLATED
public static final long	RIGHT_MASK
public static final long	RIGHT_LIST
public static final long	RIGHT_REVOKE
public static final long	RIGHT_CREATE
public static final long	RIGHT_CREATE_VERSION
public static final long	RIGHT_CREATE_POSTIT
public static final long	RIGHT_VIEW_VERSIONS
public static final long	RIGHT_VIEW_DOC_ATTR
public static final long	RIGHT_VIEW_FOLDER_ATTR
public static final long	RIGHT_VIEW_FILE
public static final long	RIGHT_VIEW_FILE_CONTENTS
public static final long	RIGHT_MODIFY_DOC_ATTR
public static final	RIGHT_MODIFY_FOLDER_ATTR

long	
public static final long	RIGHT_MODIFY_TEMP
public static final long	RIGHT_DELETE
public static final long	RIGHT_DELETE_FOLDER
public static final long	RIGHT_DELETE_CONTENTS
public static final long	RIGHT_PRINT
public static final long	RIGHT_PRINT_CONTENTS
public static final long	RIGHT_PIN
public static final long	RIGHT_ANNOTATE
public static final long	RIGHT_ARCHIVE
public static final long	RIGHT_ADMIN
public static final long	RIGHT_TAKE_OWNERSHIP
public static final long	RIGHT_MODIFY_FILE
public static final long	RIGHT_DELETE_VERSION
public static final long	RIGHT_VIEW_LOG_FOLDER
public static final long	RIGHT_VIEW_LOG_DOC
public static final long	RIGHT_DOWNLOAD_EMAIL
public static final long	RIGHT_CHANGE_FOLDER_TYPE
public static final long	RIGHT_CHANGE_DOC_TYPE

public static final long	RIGHT_SIGNATURE
public static final long	RIGHT_COMMENT_READ
public static final long	RIGHT_COMMENT_WRITE
public static final long	RIGHT_VIEW_FILE_ORIGINAL
public static final long	RIGHT_START_VALIDATION

Field detail :

```
public static final int ANONYMOUS_NUMERIC = 0x7FFFFFF0
```

The numeric part of the anonymous user ID

```
public static final int UNKNOWN_NUMERIC = 0x7FFFFFF1
```

The numeric part of the unknown user ID

```
public static final int SYSTEM_NUMERIC = 0x7FFFFFFD
```

The numeric part of the system user ID

```
public static final int EVERYONE_NUMERIC = 0x7FFFFFFE
```

The numeric part of the everyone user ID

```
public static final int OWNER_NUMERIC = 0x7FFFFFFF
```

The numeric part of the owner user ID

```
public static final int DBA_NUMERIC = 0x00000001
```

The numeric part of the DBA group ID

```
public static final int SUPER_NUMERIC = 0x00000001
```

The numeric part of the Admin user ID

```
public static final String
ANONYMOUS = ”USER00007FFFFFF0000000000000000000000000”
```

The full ID of the anonymous user

```
public static final String
UNKNOWN = ”USER00007FFFFFF1000000000000000000000000”
```

The full ID of the unknown user

```
public static final String
SYSTEM = ”USER00007FFFFFFD00000000000000000000000”
```

The full ID of the system user

```
public static final String
OWNER = ”USER00007FFFFFFE00000000000000000000000”
```

The full ID of the everyone user

```
public static final String
EVERYONE = ”USER00007FFFFFFF00000000000000000000000”
```

The full ID of the owner user

```
public static final String
DBA = ”USER00000000001000000000000000000000000”
```

The full ID of the DBA group

```
public static final String
SUPER = ”USER00000000001000000000000000000000000”
```

The full ID of the Admin user

```
public static final String DOC_TYPE_TEXT = ”TXT”
```

The text document type descriptor

```
public static final String DOC_TYPE_IMAGE = ”IMG”
```

The image document type descriptor

```
public static final String DOC_TYPE_OTHER = ”OTH”
```

The other document type descriptor

```
public static final String DOC_TYPE_UNKNOWN = ”UNK”
```

The unknown document type descriptor

```
public static final String DOC_TYPE_WATERMARK = ”WMK”
```

The watermark image type descriptor

```
public static final String DOC_TYPE_LOGINIMAGE = ”LGN”
```

The login image type descriptor

```
public static final String LANG_EN = ”en”
```

Short identifier of the english language

```
public static final String LANG_FR = ”fr”
```

Short identifier of the french language

```
public static final String LANG_ES = ”es”
```

Short identifier of the spanish language

```
public static final String LANG_AR = ”ar”
```

Short identifier of the arabic language

```
public static final String LANG_RO = ”ro”
```

Short identifier of the romanian language

```
public static final String LANG_HU = ”hu”
```

Short identifier of the hungarian language

```
public static final String LANG_RU = ”ru”
```

Short identifier of the russian language

```
public static final String LANG_TR = ”tr”
```

Short identifier of the turkish language

```
public static final String LANG_EL = ”el”
```

Short identifier of the custom language

```
public static final String LANG_EN_CODE = ”en_US”
```

Language code identifier of the english language

```
public static final String LANG_FR_CODE = ”fr_FR”
```

Language code identifier of the french language

```
public static final String LANG_ES_CODE = ”es_ES”
```

Language code identifier of the spanish language

```
public static final String LANG_AR_CODE = ”ar_AR”
```

Language code identifier of the arabic language

```
public static final String LANG_RO_CODE = ”ro_RO”
```

Language code identifier of the romanian language

```
public static final String LANG_HU_CODE = ”hu_HU”
```

Language code identifier of the hungarian language

```
public static final String LANG_RU_CODE = ”ru_RU”
```

Language code identifier of the russian language

```
public static final String LANG_TR_CODE = ”tr_TR”
```

Language code identifier of the turkish language

```
public static final String LANG_EL_CODE = ”el_XX”
```

Language code identifier of the custom language

```
public static final String LANG_EN_TRANSLATED = ”English”
```

Translated name in own language for english

```
public static final String LANG_FR_TRANSLATED = ”Français”
```

Translated name in own language for french

```
public static final String LANG_ES_TRANSLATED = ”Español”
```

Translated name in own language for spanish

```
public static final String LANG_AR_TRANSLATED = ”\u0639\u0631\u0628\
u064a”
```

Translated name in own language for arabic

```
public static final String LANG_RO_TRANSLATED = ”Română”
```

Translated name in own language for romanian

```
public static final String LANG_HU_TRANSLATED = ”Magyar”
```

Translated name in own language for hungarian

```
public static final String LANG_RU_TRANSLATED = ”\u0420\u0443\u0441\
u0441\u043A\u0438\u0439”
```

Translated name in own language for russian

```
public static final String LANG_TR_TRANSLATED = ”T\u00FCrk\u00E7e”
```

Translated name in own language for turkish

```
public static final String LANG_EL_TRANSLATED = ”Personnalisé/Custom”
```

Translated name in own language for custom

```
public static final long RIGHT_MASK = 0x7FFFFFFF7FFFFFFFL
```

The mask for all bits in the RIGHTS... constants

```
public static final long RIGHT_LIST = 0x8000000000000000L
```

The right to view that a specific object exists; without this right, a given user should never be alerted to the object’s presence

```
public static final long RIGHT_REVOKE = 0x0000000080000000L
```

Indicates whether the rights mask is a direct or a revoked mask; applies only to user masks.

```
public static final long RIGHT_CREATE = 0x0000000000000001L
```

Gives the assigned entity the right to create objects within the parent object; for instance, on a folder it will allow the creation of sub-folders or documents

```
public static final long RIGHT_CREATE_VERSION = 0x0000000000000002L
```

Gives the assigned entity the right to create versions of the object, if appropriate

```
public static final long RIGHT_CREATE_POSTIT = 0x0000000000000004L
```

Gives the assigned entity the right to crate postit notes on the selected object, if possible

```
public static final long RIGHT_VIEW_VERSIONS = 0x0000000000000008L
```

Gives the assigned entity the right to view versions of the assigned object, if appropriate; this right applies only to documents

```
public static final long RIGHT_VIEW_DOC_ATTR = 0x0000000000000010L
```

Gives the assigned entity the right to view a document’s attributes

```
public static final long RIGHT_VIEW_FOLDER_ATTR = 0x0000000000000020L
```

Gives the assigned entity the right to view a folder’s attributes

```
public static final long RIGHT_VIEW_FILE = 0x0000000000000040L
```

Gives the assigned entity the right to view the associated file; must be used together with the RIGHT_VIEW_FILE_CONTENTS right that is applied to parent object

```
public static final long RIGHT_VIEW_FILE_CONTENTS = 0x0000000000000080L
```

Gives the assigned entity the right to view the file for any object contained within the selected object

```
public static final long RIGHT_MODIFY_DOC_ATTR = 0x0000000000000100L
```

Gives the assigned entity the right to modify a document's attributes

```
public static final long RIGHT_MODIFY_FOLDER_ATTR = 0x0000000000000200L
```

Gives the assigned entity the right to modify a folder's attributes

```
public static final long RIGHT_MODIFY_TEMP = 0x0000000000000400L
```

Gives the assigned entity the right to temporarily modify the assigned object; users that have this right will be assigned additional rights for a limited period of time, as configured on a per-database basis.

```
public static final long RIGHT_DELETE = 0x0000000000000800L
```

Gives the assigned entity the right to delete a document; used together with the RIGHT_DELETE_CONTENTS right assigned to the parent object

```
public static final long RIGHT_DELETE_FOLDER = 0x0000000000001000L
```

Gives the assigned entity the right to delete a folder

```
public static final long RIGHT_DELETE_CONTENTS = 0x0000000000002000L
```

Gives the assigned entity the right to delete an object's content, if any; for instance documents from a folder

```
public static final long RIGHT_PRINT = 0x0000000000004000L
```

Gives the assigned entity the right to print the object's file; this can be effectively controlled only within the GARGANTUA application; if document is open in native application this right cannot be enforced; used together with the RIGHT_PRINT_CONTENTS assigned to the parent object

```
public static final long RIGHT_PRINT_CONTENTS = 0x0000000000008000L
```

Gives the assigned entity the right to print the objects contained within the assiciated entity

```
public static final long RIGHT_PIN = 0x0000000000010000L
```

Gives the assigned entity the right to pin or unpin documents

```
public static final long RIGHT_ANNOTATE = 0x0000000000020000L
```

Gives the assigned entity the right to add annotations in TIFF files; obsolete.

```
public static final long RIGHT_ARCHIVE = 0x0000000000040000L
```

Gives the assigned entity the right to archive the assigned object

```
public static final long RIGHT_ADMIN = 0x0000000000080000L
```

Gives the assigned entity the right to manage rights on the assigned object

```
public static final long RIGHT_TAKE_OWNERSHIP = 0x0000000000100000L
```

Gives the assigned entity the right to take the ownership of the assigned object

```
public static final long RIGHT_MODIFY_FILE = 0x0000000000200000L
```

Gives the assigned entity the right to modify the associated object's file

```
public static final long RIGHT_DELETE_VERSION = 0x0000000000400000L
```

Gives the assigned entity the right to delete an object's version

```
public static final long RIGHT_VIEW_LOG_FOLDER = 0x0000000000800000L
```

Gives the assigned entity the right to view a folder's log

```
public static final long RIGHT_VIEW_LOG_DOC = 0x0000000001000000L
```

Gives the assigned entity the right to view a document's log

```
public static final long RIGHT_DOWNLOAD_EMAIL = 0x0000000002000000L
```

Gives the assigned entity the right to download or email the assigned object's file

```
public static final long RIGHT_CHANGE_FOLDER_TYPE = 0x0000000004000000L
```

Gives the assigned entity the right to change the form associated with a folder object; please note that when changing forms, the existing attributes that are attached to the new form will remain valued

```
public static final long RIGHT_CHANGE_DOC_TYPE = 0x0000000008000000L
```

Gives the assigned entity the right to change the form associated with a document object; please note that when changing forms, the existing attributes that are attached to the new form will remain valued

```
public static final long RIGHT_SIGNATURE = 0x0000000010000000L
```

Gives the assigned entity the right to apply a digital signature to the document

```
public static final long RIGHT_COMMENT_READ = 0x0000000020000000L
```

Gives the assigned entity the right to access the forum section for the assigned object

```
public static final long RIGHT_COMMENT_WRITE = 0x0000000040000000L
```

Gives the assigned entity the right to add comments to the assigned object's forum

```
public static final long RIGHT_VIEW_FILE_ORIGINAL = 0x0000000100000000L
```

Gives the assigned entity the right to view the original file when the database uses PDF conversion; by default, entities without this right will only be able to display the converted version

```
public static final long RIGHT_START_VALIDATION = 0x0000000200000000L
```

Gives the assigned entity the right to start a validation on the assigned entity

FilterCriterion

```
public class FilterCriterion
```

This class defines a filter criterion used by view elements definition.

This class consists of:

1. the **id of the attribute** used to filter a domain.
2. 4 **reference values** which specify the domain (min, max, eq, ne).

Supported combinations of the reference values are:

combination	significance
(min != null, max = null, eq = null, ne = null, vN = null, vNNull = null)	specifies attribute must have values greater than min
(min = null, max != null, eq = null, ne = null, vN = null, vNNull = null)	specifies attribute must have values lower than min
(min != null, max != null, eq = null, ne = null, vN = null, vNNull = null)	specifies attribute must have values in the range between min and max
(min = null, max = null, eq != null, ne = null, vN = null, vNNull = null)	specifies attribute values must be equal to eq
(min = null, max = null, eq = null, ne != null, vN = null, vNNull = null)	specifies attribute values must not be equal to eq
(min = null, max = null, eq = null, ne = null, vN != null, vNNull = null)	specifies attribute values must not be null
(min = null, max = null, eq = null, ne = null, vN = null, vNNull != null)	specifies attribute values must not be null

All Superinterfaces :

Cloneable

Since :

12/07/2006

Author :

vic

Constructor summary :

Constructor
FilterCriterion ()

```
FilterCriterion (java.lang.String attrId, java.lang.Object valueMin,
java.lang.Object valueMax, java.lang.Object valueEq, java.lang.Object
valueNe)
FilterCriterion (java.lang.String attrId, java.lang.Object valueMin,
java.lang.Object valueMax, java.lang.Object valueEq, java.lang.Object
valueNe, java.lang.Object valueCt, java.lang.Object valueNc)
FilterCriterion (java.lang.String attrId, java.lang.Object valueMin,
java.lang.Object valueMax, java.lang.Object valueEq, java.lang.Object
valueNe, java.lang.Object valueCt, java.lang.Object valueNc,
java.lang.Object valueIsNull, java.lang.Object valueIsNotNull)
```

Constructor detail :

```
public FilterCriterion ()
```

No arguments constructor.

```
public FilterCriterion (String attrId, Object valueMin, Object valueMax,
Object valueEq, Object valueNe)
```

Constructor with initialization parameters.

Parameters :

`attrId` - the attribute id

`valueMin` - the min value

`valueMax` - the max value

`valueEq` - the eq value

`valueNe` - the ne value

```
public FilterCriterion (String attrId, Object valueMin, Object valueMax,
Object valueEq, Object valueNe, Object valueCt, Object valueNc)
```

```
public FilterCriterion (String attrId, Object valueMin, Object valueMax,
Object valueEq, Object valueNe, Object valueCt, Object valueNc, Object
valueIsNull, Object valueIsNotNull)
```

FLAG

```
public class Flag
```

Various flags used across the GARGANTUA system.

Bits 0-39 match to core definition interfaces' accessors (Identified, Labeled, Owned etc.).

Bits 40-63 match to core objects specific interfaces' accessors (Folder, Process, Document etc.)

All Superinterfaces :

IEFlag , SearchFlag , AuditFlag , OtherFlag

Author :

stefanu, victorc

Field summary :

Modifier and type	Field
public static final long	ID
public static final long	LABEL
public static final long	OWNER_ID
public static final long	FLAGS
public static final long	DESCRIPTION
public static final long	DEPLOYED
public static final long	VERSION
public static final long	FORM_ID
public static final long	ATTRIBUTES
public static final long	FILE_ID
public static final long	FILE_TYPE

public static final long	FILE_SIZE
public static final long	DATE_C
public static final long	DATE_M
public static final long	DATE_A
public static final long	ICON
public static final long	DATE_S
public static final long	DATE_E
public static final long	DATE_D
public static final long	DATE_X
public static final long	OVERDUE
public static final long	EXPIRED
public static final long	REFCOUNT
public static final long	PRIORITY
public static final long	SECURITY_PROFILE
public static final long	RENDER_SECURITY
public static final long	UUID
public static final long	TAGS
public static final long	DATE_AR
public static final	EMAIL

long	
public static final long	REAL_NAME
public static final long	DIRECTORY
public static final long	PASSWORD
public static final long	PRIVILEGES
public static final long	PROFILE
public static final long	ACTIVATED_BY
public static final long	ACTIVATED_AT
public static final long	DEACTIVATED_BY
public static final long	DEACTIVATED_AT
public static final long	DATE_LI
public static final long	DATE_LO
public static final long	WORKING_PLANNER
public static final long	ACCESS_RESTRICTED_BY_WP
public static final long	ORDER_INDEX
public static final long	PARENT_ID
public static final long	DESCRIPTION_URL
public static final long	CONN_URL
public static final long	CRON_EXPRESSION

public static final long	SCRIPT_LANG
public static final long	SCRIPT_TYPE
public static final long	SAMPLE_FILE_ID
public static final long	DISPLAYPROFILE_ID
public static final long	LIFECYCLE_RULE_FILEID
public static final long	ARCHIVE_STORAGE_ID
public static final long	FRAGMENT_FILE_ID
public static final long	HIERARCHY_FILE_ID
public static final long	VERS_OBJ_COMMENT
public static final long	VERS_OBJ_INDEX
public static final long	DATA
public static final long	VERSION_OF
public static final long	DELAY_X
public static final long	DELAY_D
public static final long	PROCESS_ID_PLANNER
public static final long	DEPLOYMENT_FORM_ID
public static final long	MONITOR_ROLE_ID
public static final long	DEFINITION_ID
public static final	RUNTIME_STATE

long	
public static final long	RUNTIME_FILE_ID
public static final long	PROCESS_STATUS
public static final long	PROCESS_START_POINT
public static final long	DRAFT_FORM_PAGE
public static final long	PROFILE_ID
public static final long	TASK_STATUS
public static final long	TASK_ACTORID
public static final long	TASK_ROLEID
public static final long	TASK_EXECUTORID
public static final long	TASK_LOCATION
public static final long	TASK_SUBJECT
public static final long	TASK_PROCESS_LABEL
public static final long	TASK_FORM_PAGE
public static final long	TASK_FORM_ID
public static final long	TASK_GROUPID
public static final long	TASK_CONFIDENTIAL
public static final long	TIMETABLE
public static final long	DATABASE_ID

public static final long	DAY_TIMETABLE
public static final long	DAY_DATE
public static final long	DAY_RECURRENT
public static final long	RES_TYPE
public static final long	COMMENT_CONTENT
public static final long	OWNER_NAME
public static final long	HIGHER_ID
public static final long	CARDINALITY
public static final long	ROLE_TYPE
public static final long	PROCESS_TESTING_CREATED_BY
public static final long	PROCESS_TESTING_CREATED_BY_LABEL
public static final long	PROCESS_TESTING_CREATED_BY_REALNAME
public static final long	PROCESS_TESTING_MODIFIED_BY
public static final long	PROCESS_TESTING_MODIFIED_BY_LABEL
public static final long	PROCESS_TESTING_MODIFIED_BY_REALNAME
public static final long	PROCESS_TESTING_DEFINITION_TOTAL_RUNS
public static final long	PROCESS_TESTING_DEFINITION_SUCCESS_RUNS
public static final long	PROCESS_TESTING_DEFINITION_FAIL_RUNS
public static final	PROCESS_TESTING_DEFINITION_LAST_SUCCESS

long	
public static final long	PROCESS_TESTING_DEFINITION_LAST_FAIL
public static final long	PROCESS_TESTING_DEFINITION_LAST_DURATION
public static final long	PROCESS_TESTING_INSTANCE_STARTED_BY_ID
public static final long	PROCESS_TESTING_INSTANCE_STARTED_BY_LABEL
public static final long	PROCESS_TESTING_INSTANCE_STARTED_BY_REALNAME
public static final long	PROCESS_TESTING_INSTANCE_STARTED_AT
public static final long	PROCESS_TESTING_INSTANCE_FINISHED_BY_ID
public static final long	PROCESS_TESTING_INSTANCE_FINISHED_BY_LABEL
public static final long	PROCESS_TESTING_INSTANCE_FINISHED_BY_REALNAME
public static final long	PROCESS_TESTING_INSTANCE_FINISHED_AT
public static final long	USER_ALL
public static final long	USER_UPDATE_ALL
public static final long	GROUP_ALL
public static final long	GROUP_UPDATE_ALL
public static final long	ROLE_ALL
public static final long	ROLE_UPDATE_ALL
public static final long	POSITION_ALL
public static final long	POSITION_UPDATE_ALL

public static final long	PORTAL_ALL
public static final long	PORTAL_UPDATE_ALL
public static final long	PLANNER_ALL
public static final long	PLANNER_DAY_ALL
public static final long	DATABASE_ALL
public static final long	DATABASE_UPDATE_ALL
public static final long	STORAGE_ALL
public static final long	RULE_ALL
public static final long	RESOURCE_ALL
public static final long	TYPE_ALL
public static final long	GRID_ALL
public static final long	ATTRIBUTE_ALL
public static final long	ATTRIBUTE_UPDATE_MASK
public static final long	FORM_ALL
public static final long	FORM_UPDATE_ALL
public static final long	FORM_DEPLOY
public static final long	FORM_UNDEPLOY
public static final long	PROCESSDEF_ALL
public static final	PROCESSDEF_UPDATE_ALL

long	
public static final long	PROCESSDEF_DEPLOY
public static final long	PROCESSDEF_UNDEPLOY
public static final long	PROCESS_MANAGEMENT
public static final long	SCRIPT_ALL
public static final long	SCRIPT_VERSION_ALL
public static final long	VIEW_ALL
public static final long	VIEW_UPDATE_ALL
public static final long	VIEW_ELEMENT_ALL
public static final long	VIEW_ELEMENT_UPDATE_ALL
public static final long	VIEW_RUNTIME_TREE
public static final long	VIEW_DEPLOY
public static final long	VIEW_UNDEPLOY
public static final long	ISJOB_ALL
public static final long	MODELOCR_ALL
public static final long	SECURITY_PROFILE_ALL
public static final long	FOLDER_PROFILE_ALL
public static final long	FOLDER_TREE
public static final long	FOLDER_ALL

public static final long	FOLDER_FULL
public static final long	FOLDER_UPDATE_ALL
public static final long	FOLDER_UPDATE_FULL
public static final long	DOCUMENT_THUMB
public static final long	DOCUMENT_VC_SYSATTRS
public static final long	DOCUMENT_LIST
public static final long	DOCUMENT_ALL
public static final long	DOCUMENT_FULL
public static final long	VERSION_ALL
public static final long	TRASH_INFO
public static final long	NOTE_ALL
public static final long	SEARCHTEMPLATE_ALL
public static final long	SEARCHTEMPLATE_UPDATEALL
public static final long	PROCESS_ALL
public static final long	TASK_INBOX
public static final long	TASK_ALL
public static final long	USER_PROFILE_ALL
public static final long	JASY_ALL
public static final	COMPOSITION_RULE_ALL

long	
public static final long	COMPOSITION_RULE_VERSION_ALL
public static final long	FORM_DISPLAY_PROFILE_ALL
public static final long	FORM_VERSION_ALL
public static final long	PROCESSDEF_VERSION_ALL
public static final long	ISJOB_VERSION_ALL
public static final long	MODELOCR_VERSION_ALL
public static final long	VIEW_VERSION_ALL
public static final long	COMMENT_ALL
public static final long	COMMENT_UPDATE_ALL
public static final long	MAIL_TEMPLATE_ALL
public static final long	DEF_FILE_ID
public static final long	DOCUMENT_TEMPLATE_ALL
public static final long	FOR_FOLDERS
public static final long	FOR_PROCESSES
public static final long	FOR_DOCUMENTS
public static final long	FOR_SEARCH
public static final long	HIDE_IN_SEARCH
public static final long	CLASSES_MASK

public static final long	IS_OBJECTS
public static final long	SHOW_SUBFOLDERS
public static final long	CLASS_SORT_MASK
public static final long	ORDER_ASC_1
public static final long	ORDER_DESC_1
public static final long	ORDER_ASC_2
public static final long	ORDER_DESC_2
public static final long	ORDER_ASC_3
public static final long	ORDER_DESC_3
public static final long	ROOT_VE
public static final long	HAS_CLASSES
public static final long	HAS_DOCUMENTS
public static final long	IS_FOLDER
public static final long	IS_PROCESS
public static final long	ADD_ENABLED
public static final long	HAS_FULLTEXT
public static final long	SECURITY_MODE_MASK
public static final long	SECURITY_MODE_NONE
public static final	SECURITY_MODE_SIMPLE

long	
public static final long	SECURITY_MODE_MEDIUM
public static final long	SECURITY_MODE_COMPLEX
public static final long	IS_PUBLIC
public static final long	DATABASE_RESTRICTED_ARCHIVE
public static final long	DATABASE_OFFLINE
public static final long	DATABASE_WORKFLOW_INACTIVE
public static final long	DATABASE_LIFECYCLE_INACTIVE
public static final long	DATABASE_LIFECYCLE_DEBUG
public static final long	DATABASE_UNAVAILABLE
public static final long	ENABLE_COMMENTS
public static final long	ENABLE_TRASH
public static final long	DISPLAY_TREE_COUNTERS
public static final long	DATABASE_LUCENE_SIMPLESEARCH
public static final long	DATABASE_BUILD_LUCENE_SIMPLESEARCH
public static final long	DATABASE_LUCENE_CONTENTSEARCH
public static final long	DATABASE_BUILD_LUCENE_CONTENTSEARCH
public static final long	FULLTEXT_ACCENT_INSENSITIVE
public static final long	FULLTEXT_ACCENT_SENSITIVE

public static final long	FULLTEXT_ACCENT_BOTH
public static final long	IS_DEFAULT
public static final long	FOLDER_HAS_ATTRIBUTES
public static final long	FOLDER_HAS_NOTES
public static final long	FOLDER_HAS_CHILDREN
public static final long	HAS_COMMENTS
public static final long	HAS_NOTIFICATIONS
public static final long	FORM_ATTR_MANDATORY
public static final long	FORM_ATTR_LINKED
public static final long	FORM_ATTR_LOCKED
public static final long	FORM_ATTR_XNORMAL
public static final long	FORM_ATTR_XSIMPLE
public static final long	FORM_ATTR_XFULLPATH
public static final long	FORM_ATTR_XMULTPILE
public static final long	FORM_ATTR_XCATEGORY
public static final long	FORM_ATTR_XMULTILINE
public static final long	FORM_ATTR_XRADIO
public static final long	FORM_ATTR_XRADIO_INLINE
public static final	FORM_ATTR_XMASK

long	
public static final long	PROFILE_DEFAULT
public static final long	PROFILE_ANONYMOUS
public static final long	PROFILE_EXTERNAL_EMAIL
public static final long	DOCSTAT_ATTRIBUTES
public static final long	DOCSTAT_CDARCHIVED
public static final long	DOCSTAT_LOCK_OTHER
public static final long	DOCSTAT_LOCK_MYSELF
public static final long	DOCSTAT_LOCKED
public static final long	DOCSTAT_NOTES
public static final long	DOCSTAT_FULLTEXT
public static final long	DOCSTAT_HAS_VERSIONS
public static final long	DOCSTAT_SYS_ATTRIBUTES
public static final long	DOCSTAT_COPYOFDOC
public static final long	DOCSTAT_REFTODOC
public static final long	DOCSTAT_PDF_HAS_TEXT
public static final long	DOCSTAT_0000000000004000_BIT_AVAILABLE
public static final long	DOCSTAT_HAS_COMMENTS
public static final long	DOCSTAT_HAS_NOTIFICATIONS

public static final long	USER_IS_EXTERNAL_LDAP
public static final long	USER_IS_SERVER_ACCOUNT
public static final long	USER_SALUTATION_MASK
public static final long	USER_SALUTATION_NONE
public static final long	USER_SALUTATION_MR
public static final long	USER_SALUTATION_MRS
public static final long	USER_SALUTATION_MS
public static final long	USER_LOCKED
public static final long	USER_CHANGEPASS
public static final long	USER_IS_EXTERNAL_EMAIL
public static final long	USER_DENY_LOGIN
public static final long	ROLE_IS_ATTACHED
public static final long	POSITION_WF_MASTER_ACCESS
public static final long	POSITION_CAN_SIGN
public static final long	ROLE_LAYOUT_MASK
public static final long	CAN_CANCEL
public static final long	CAN_FINISH
public static final long	CAN_STOP
public static final	FLOW_VIEW

long	
public static final long	FLOW_EDIT
public static final long	IS_SHARED_TASK
public static final long	WAIT_SHARED_TASK_FINISH
public static final long	NO_QUIT_TASK
public static final long	IGNORE_FAIL
public static final long	MANDATORY_USE_OF_PROCESS_PLANNER
public static final long	KEEP_VALUES_ON_RESTART
public static final long	ALLOW_ESCALATE
public static final long	OVERDUE_BY_TASK
public static final long	EXPIRING_BY_TASK
public static final long	DELETE_ON_FINISH
public static final long	DELETE_ON_CANCEL
public static final long	DELETE_ON_EXPIRE
public static final long	ATTR_ADVANCED_SEARCH
public static final long	ATTR_DATABASE_INDEX_PENDING
public static final long	ATTR_DATABASE_INDEX
public static final long	ATTR_CAPITAL_LETTERS
public static final long	VIRTUAL

public static final long	NO_CHILDREN_ALIVE
public static final long	HAS_VIRTUAL
public static final long	IS_LADRAD
public static final long	USER_PLANNER
public static final long	PROCESS_PLANNER
public static final long	VERSION_PLANNER
public static final long	REORDER_LABEL_ASC
public static final long	REORDER_LABEL_DESC
public static final long	EMAIL_NOTIFICATION
public static final long	COMMENT_IS_CLOSED
public static final long	CERT_ISSUED_BY
public static final long	CERT_SERVER
public static final long	CERT_FLAG_VALID_2
public static final long	CERT_IDENTITY_PREFIX
public static final long	CERT_IDENTITY_N0
public static final long	CERT_IDENTITY
public static final long	CERT_GEN_CACERT
public static final long	CERTIFICATE_ALL
public static final	IS_JOB_SCAN

long	
public static final long	IS_JOB_IMPORT
public static final long	IS_JOB_EMAIL
public static final long	IS_JOB_LAD_RAD
public static final long	IS_JOB_SIGN
public static final long	IS_JOB_SIGN_MANDATORY
public static final long	SIMPLE_DEBUG
public static final long	ENABLE_DOCUMENT_ACCESS
public static final long	HAS_PLUGIN_OPTIONS
public static final long	NOTIFICATION_DEFINITION_ALL
public static final long	NOTIFICATION_DEFINITION_FULL
public static final long	NOTIFICATION_ALL
public static final long	NOTIFICATION_FULL
public static final long	FAVORITE_ALL
public static final long	VC_FLAG_START
public static final long	VC_FLAG_MODIF
public static final long	VC_FLAG_AVIZ
public static final long	VC_FLAG_VALID
public static final long	VC_FLAG_PUBLIC

public static final long	VC_FLAG_GED
public static final long	VC_FLAG_BROADCAST
public static final long	PACKAGE_ALL
public static final long	SECURITY_PROFILE_FOR_WORKSPACE
public static final long	OPINION_SET_BY_SUPERVISOR
public static final long	TYPE_LOCALIZED
public static final long	TYPE_INDEXED
public static final long	TYPE_RESTRICT_DUPLICATES
public static final long	ENABLE_TRANSLATIONS
public static final long	MAIL_TEMPLATE_TYPE_MASK
public static final long	MAIL_TEMPLATE_FOR_PROCESS
public static final long	MAIL_TEMPLATE_FOR_VCYCLE
public static final long	MAIL_TEMPLATE_FOR_GENERAL_PURPOSE
public static final long	ENABLE_HTML
public static final long	PROCESS_TESTING_DEFINITION_ALL
public static final long	PROCESS_TESTING_INSTANCE_ALL

Field detail :

```
public static final long ID = 1L
```

Bit 0. Specifies the 'id' field of com.siatel.defs.Identified implementing objects.

```
public static final long LABEL = 2L
```

Bit 1. Specifies the 'label' property of com.siatel.defs.Labeled implementing objects.

```
public static final long OWNER_ID = 4L
```

Bit 2. Specifies the 'owner' property of com.siatel.defs.Owned implementing objects.

```
public static final long FLAGS = 8L
```

Bit 3. Specifies the 'flags' property of com.siatel.defs.Flagged implementing objects.

```
public static final long DESCRIPTION = 16L
```

Bit 4. Specifies the 'description' property of com.siatel.defs.Described implementing objects.

```
public static final long DEPLOYED = 32L
```

Bit 5. Specifies the 'deployed' property of com.siatel.defs.Deployable implementing objects.

```
public static final long VERSION = 64L
```

Bit 6. Specifies the 'version' property of com.siatel.defs.Versioned implementing objects.

```
public static final long FORM_ID = 128L
```

Bit 7. Specifies the 'form id' property of com.siatel.defs.Viewable implementing objects.

```
public static final long ATTRIBUTES = 256L
```

Bit 8. Specifies the 'attributes' property of com.siatel.defs.Viewable implementing objects.

```
public static final long FILE_ID = 512L
```

Bit 9. Specifies the 'file id' property of com.siatel.defs.Realized implementing objects.

```
public static final long FILE_TYPE = 1024L
```

Bit 10. Specifies the 'file type' property of com.siatel.defs.RealizedEx implementing objects.

```
public static final long FILE_SIZE = 2048L
```

Bit 11. Specifies the 'file size' property of com.siatel.defs.RealizedEx implementing objects.

```
public static final long DATE_C = 4096L
```

Bit 12. Specifies the 'date created' property of com.siatel.defs.Dated implementing objects.

```
public static final long DATE_M = 8192L
```

Bit 13. Specifies the 'date modified' property of com.siatel.defs.Dated implementing objects.

```
public static final long DATE_A = 16384L
```

Bit 14. Specifies the 'date accessed' property of com.siatel.defs.DatedEx implementing objects.

```
public static final long ICON = 32768L
```

Bit 15. Specifies the 'icon' property of com.siatel.defs.Viewable implementing objects.

```
public static final long DATE_S = 65536L
```

Bit 16. Specifies the 'date started' property of com.siatel.defs.Timed implementing objects.

```
public static final long DATE_E = 131072L
```

Bit 17. Specifies the 'date ended' property of com.siatel.defs.Timed implementing objects.

```
public static final long DATE_D = 262144L
```

Bit 18. Specifies the 'date due' property of com.siatel.defs.Expirable implementing objects.

```
public static final long DATE_X = 524288L
```

Bit 19. Specifies the 'date expired' property of com.siatel.defs.Expirable implementing objects.

```
public static final long OVERDUE = 1048576L
```

Bit 20. Specifies the 'overdue' property of com.siatel.defs.Expirable implementing objects.

```
public static final long EXPIRED = 2097152L
```

Bit 21. Specifies the 'expired' property of com.siatel.defs.Expirable implementing objects.

```
public static final long REFCOUNT = 4194304L
```

Bit 22. Specifies the 'refcount' property of com.siatel.defs.Referenced implementing objects.

```
public static final long PRIORITY = 33554432L
```

Bit 25. Specifies the 'priority' property of com.siatel.defs.Prioritized implementing objects.

```
public static final long SECURITY_PROFILE = 34359738368L
```

Bit 35. Specifies the 'security profile' property of com.siatel.defs.Securable implementing objects.

```
public static final long RENDER_SECURITY = 68719476736L
```

Bit 36. Specifies the 'render security' operation which will fill the 'security mask' property of com.siatel.defs.Securable implementing objects.

```
public static final long UUID = 137438953472L
```

Bit 37. Specifies the 'uuid' property of com.siatel.defs.UUIDEnabled implementing objects.

```
public static final long TAGS = 274877906944L
```

Bit 38. Specifies the 'tags' property of com.siatel.defs.Tagged implementing objects.

```
public static final long DATE_AR = 549755813888L
```

Bit 39. Specifies the 'date archive' property of com.siatel.defs.Archive implementing objects.

```
public static final long EMAIL = 1099511627776L
```

User: Bit 40. Specifies the User#getEMail() property.

```
public static final long REAL_NAME = 2199023255552L
```

User: Bit 41. Specifies the User#getRealName() property.

```
public static final long DIRECTORY = 4398046511104L
```

User, Group, Role: Bit 42. Specifies the User#getExternalDirectoryId() (similar for groups and roles) property.

```
public static final long PASSWORD = 8796093022208L
```

User: Bit 43. Specifies the 'password' property for.

```
public static final long PRIVILEGES = 17592186044416L
```

User, Group, Role: Bit 44. Specifies the User#getPrivileges(), (similar for groups and roles) property.

```
public static final long PROFILE = 35184372088832L
```

User: Bit 45. Specifies the User#getProfileId() property.

```
public static final long ACTIVATED_BY = 4503599627370496L
```

User: Bit 52. Specifies the 'activated by' property.

```
public static final long ACTIVATED_AT = 9007199254740992L
```

User: Bit 53. Specifies the 'activation date-time ' property.

```
public static final long DEACTIVATED_BY = 18014398509481984L
```

User: Bit 54. Specifies the 'deactivated by' property.

```
public static final long DEACTIVATED_AT = 36028797018963968L
```

User: Bit 55. Specifies the 'deactivation date-time' property.

```
public static final long DATE_LI = 72057594037927936L
```

User: Bit 56. Specifies the User#getDateLogin() property.

```
public static final long DATE_LO = 144115188075855872L
```

User: Bit 57. Specifies the User#getDateLogout() property.

```
public static final long WORKING_PLANNER = 288230376151711744L
```

User: Bit 58. Specifies the 'working planner' property.

```
public static final long ACCESS_RESTRICTED_BY_WP = 576460752303423488L
```

User: Bit 59. Specifies the 'planner restricted access' property.

```
public static final long ORDER_INDEX = 70368744177664L
```

Position: Bit 46. Specifies the 'order index' property.

```
public static final long PARENT_ID = 1099511627776L
```

Folder, ViewElement, ClassificationElement, Comment, Group: Bit 40. Specifies the 'parentId' property.

```
public static final long DESCRIPTION_URL = 1099511627776L
```

Database, View: Bit 40. Specifies the Database#getDescriptionUrl() (similar for View) property.

```
public static final long CONN_URL = 35184372088832L
```

Database: Bit 45. Specifies the Database#getConnUrl() property.

```
public static final long CRON_EXPRESSION = 70368744177664L
```

Database: Bit 46. Specifies the Database#getCronExpression() property.

```
public static final long SCRIPT_LANG = 1099511627776L
```

Script, Script2: Bit 40. Specifies the Script#getLang(), Script2#getLang() property.

```
public static final long SCRIPT_TYPE = 2199023255552L
```

Script, Script2: Bit 41. Specifies the Script#getType(), Script2#getType() property.

```
public static final long SAMPLE_FILE_ID = 1099511627776L
```

ModelOCR: Bit 40. Specifies the ModelOCR#getSampleFileId() property.

```
public static final long DISPLAYPROFILE_ID = 1099511627776L
```

Bit 40. Specifies the Form#getDisplayProfileId() property (similar for FormVersion).

```
public static final long LIFECYCLE_RULE_FILEID = 2199023255552L
```

Bit 41. Specifies the Form#getLifecycleRuleFileId() property (similar for FormVersion).

```
public static final long ARCHIVE_STORAGE_ID = 4398046511104L
```

Bit 42. Specifies the Form#getArchiveStorageId() property (similar for FormVersion).

```
public static final long FRAGMENT_FILE_ID = 8796093022208L
```

Bit 43. Specifies the Form#getFragmentFileId() property (similar for FormVersion).

```
public static final long HIERARCHY_FILE_ID = 17592186044416L
```

Bit 44. Specifies the Form#getHierarchyFileId() property (similar for FormVersion).

```
public static final long VERS_OBJ_COMMENT = 36028797018963968L
```

Commented: Bit 55. Specifies the Commented#getComment() property. Usually applies to design object versions.

```
public static final long VERS_OBJ_INDEX = 72057594037927936L
```

Numbered: Bit 56. Specifies the Numbered#getNumber() property. Usually applies to design object versions.

```
public static final long DATA = 18014398509481984L
```

Bit 54. Specifies the Attribute#getData() property. Reused by Position, Database

```
public static final long VERSION_OF = 1099511627776L
```

ProcessDefinition: Bit 40: Specifies the ProcessDefinition#getVersionOfId() property

```
public static final long DELAY_X = 2199023255552L
```

ProcessDefinition: Bit 41: Specifies the ProcessDefinition#getDelayX() property

```
public static final long DELAY_D = 4398046511104L
```

ProcessDefinition: Bit 42: Specifies the ProcessDefinition#getDelayD() property

```
public static final long PROCESS_ID_PLANNER = 8796093022208L
```

ProcessDefinition: Bit 43: Specifies the ProcessDefinition#getPlannerId() property

```
public static final long DEPLOYMENT_FORM_ID = 17592186044416L
```

ProcessDefinition: Bit 44: Specifies the ProcessDefinition#getDeploymentFormId() property

```
public static final long MONITOR_ROLE_ID = 281474976710656L
```

ProcessDefinition: Bit 52: Specifies the ProcessDefinition#getMonitorRoleId(), Process#getMonitorRoleId() property

```
public static final long DEFINITION_ID = 1099511627776L
```

Process: Bit 40: Specifies the Process#getDefinitionId() property

```
public static final long RUNTIME_STATE = 2199023255552L
```

Process: Bit 41: Specifies the Process#getRuntimeState() property

```
public static final long RUNTIME_FILE_ID = 4398046511104L
```

Process: Bit 42: Specifies the Process#getRuntimeFileId() property

```
public static final long PROCESS_STATUS = 8796093022208L
```

Process: Bit 43: Specifies the Process#getStatus() property

```
public static final long PROCESS_START_POINT = 17592186044416L
```

Process: Bit 44: Specifies the Process#getStartPoint() property

```
public static final long DRAFT_FORM_PAGE = 35184372088832L
```

Process: Bit 45: Specifies the Process#getDraftFormPage() property

```
public static final long PROFILE_ID = 2251799813685248L
```

Folder, Process, ProcessDefinition: Bit 51: Specifies the Folder#getProfileId() property (similar for Process, ProcessDefinition)

```
public static final long TASK_STATUS = 1099511627776L
```

Task: Bit 40: Specifies the Task#getStatus() property.

```
public static final long TASK_ACTORID = 2199023255552L
```

Task: Bit 41: Specifies the Task#getActorId() property.

```
public static final long TASK_ROLEID = 2199023255552L
```

Task: Bit 41: Specifies the Task#getRoleId() property. Overrides value of previous #TASK_ACTORID.

```
public static final long TASK_EXECUTORID = 4398046511104L
```

Task: Bit 42: Specifies the Task#getExecutorId() property.

```
public static final long TASK_LOCATION = 8796093022208L
```

Task: Bit 43: Specifies the Task#getLocation() property.

```
public static final long TASK_SUBJECT = 17592186044416L
```

Task: Bit 44: Specifies the Task#getSubject() property.

```
public static final long TASK_PROCESS_LABEL = 35184372088832L
```

Task: Bit 45: Specifies the Task#getProcessLabel() property.

```
public static final long TASK_FORM_PAGE = 70368744177664L
```

Task: Bit 46: Specifies the Task#getFormPage() property.

```
public static final long TASK_FORM_ID = 140737488355328L
```

Task: Bit 47: Specifies the Task#getFormId() property.

```
public static final long TASK_GROUPID = 72057594037927936L
```

Task: Bit 48: Specifies the Task#getGroupId() property.

```
public static final long TASK_CONFIDENTIAL = 1441151880758558 72L
```

Task: Bit 49: Specifies the Task#isConfidential() property.

```
public static final long TIMETABLE = 1099511627776L
```

Planner: Bit 40: Specifies the Planner#getTimetable() property.

```
public static final long DATABASE_ID = 2199023255552L
```

Planner: Bit 41: Internal flag, it's purpose does not go beyond the persistence level. Developer note: if used in planners, it does not require explicit definition, it is added if the ID flag is present (the id of the planner depends on the database object that uses it)

```
public static final long DAY_TIMETABLE = 1099511627776L
```

PlannerDay: Bit 40: Specifies the PlannerDay#getTimetable() property.

```
public static final long DAY_DATE = 2199023255552L
```

PlannerDay: Bit 41: Specifies the PlannerDay#getDate() property.

```
public static final long DAY_RECURRENT = 4398046511104L
```

PlannerDay: Bit 42: Specifies the PlannerDay#isRecurrent() property.

```
public static final long RES_TYPE = 1099511627776L
```

Resource: Bit 40: Specifies the Resource#getType() property.

```
public static final long COMMENT_CONTENT = 2199023255552L
```

Comment: Bit 41: Specifies the Comment#getContent() property.

```
public static final long OWNER_NAME = 4398046511104L
```

Comment: Bit 42: Specifies the Comment#getOwnerName() property.

```
public static final long HIGHER_ID = 175921860444160L
```

Role: Bit 45 and 47: Specifies the Role#getHigherId() property.

```
public static final long CARDINALITY = 281474976710656L
```

Role: Bit 48: Specifies the Role#getCardinality() property.

```
public static final long ROLE_TYPE = 562949953421312L
```

Role: Bit 48: Specifies the Role#getType() property.

```
public static final long PROCESS_TESTING_CREATED_BY = 32L
```

ProcessTesting: Specifies the ProcessTesting#getCreatedById() property.

```
public static final long PROCESS_TESTING_CREATED_BY_LABEL = 64L
```

ProcessTesting: Specifies the ProcessTesting#getCreatedByLabel() property.

```
public static final long PROCESS_TESTING_CREATED_BY_REALNAME = 128L
```

ProcessTesting: Specifies the ProcessTesting#getCreatedByRealName() property.

```
public static final long PROCESS_TESTING_MODIFIED_BY = 256L
```

ProcessTesting: Specifies the ProcessTesting#getModifiedById() property.

```
public static final long PROCESS_TESTING_MODIFIED_BY_LABEL = 1024L
```

ProcessTesting: Specifies the ProcessTesting#getModifiedByLabel() property.

```
public static final long PROCESS_TESTING_MODIFIED_BY_REALNAME = 2048L
```

ProcessTesting: Specifies the ProcessTesting#getModifiedByRealName() property.

```
public static final long PROCESS_TESTING_DEFINITION_TOTAL_RUNS = 32768L
```

ProcessTestingDefinition: Specifies the ProcessTestingDefinition#getTotalRuns() property.

```
public static final long
PROCESS_TESTING_DEFINITION_SUCCESS_RUNS = 65536L
```

ProcessTestingDefinition: Specifies the ProcessTestingDefinition#getSuccessRuns() property.

```
public static final long PROCESS_TESTING_DEFINITION_FAIL_RUNS = 131072L
```

ProcessTestingDefinition: Specifies the ProcessTestingDefinition#getFailRuns() property.

```
public static final long
PROCESS_TESTING_DEFINITION_LAST_SUCCESS = 262144L
```

ProcessTestingDefinition: Specifies the ProcessTestingDefinition#getLastSuccess() property.

```
public static final long PROCESS_TESTING_DEFINITION_LAST_FAIL = 524288L
```

ProcessTestingDefinition: Specifies the ProcessTestingDefinition#getLastFail() property.

```
public static final long
PROCESS_TESTING_DEFINITION_LAST_DURATION = 1048576L
```

ProcessTestingDefinition: Specifies the ProcessTestingDefinition#getLastDuration() property.

```
public static final long
PROCESS_TESTING_INSTANCE_STARTED_BY_ID = 2097152L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getStartedById() property.

```
public static final long
PROCESS_TESTING_INSTANCE_STARTED_BY_LABEL = 4194304L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getStartedByLabel() property.

```
public static final long
PROCESS_TESTING_INSTANCE_STARTED_BY_REALNAME = 8388608L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getStartedByRealName() property.

```
public static final long PROCESS_TESTING_INSTANCE_STARTED_AT = 16777216L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getStartedAt() property.

```
public static final long
PROCESS_TESTING_INSTANCE_FINISHED_BY_ID = 33554432L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getFinishedById() property.

```
public static final long
PROCESS_TESTING_INSTANCE_FINISHED_BY_LABEL = 67108864L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getFinishedByLabel() property.

```
public static final long
PROCESS_TESTING_INSTANCE_FINISHED_BY_REALNAME = 134217728L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getFinishedByRealName() property.

```
public static final long
PROCESS_TESTING_INSTANCE_FINISHED_AT = 268435456L
```

ProcessTestingInstance: Specifies the ProcessTestingInstance#getFinishedAt() property.

```
public static final long USER_ALL = 1085507149246771231L
```

```
ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M |
PASSWORD | PRIVILEGES | EMAIL | REAL_NAME | DIRECTORY | DATE_LI |
```

DATE_LO | WORKING_PLANNER | ACCESS_RESTRICTED_BY_WP

```
public static final long USER_UPDATE_ALL = 86471641722258298L
```

LABEL | FLAGS | DESCRIPTION | DATE_M | PRIVILEGES | EMAIL | REAL_NAME | DIRECTORY | WORKING_PLANNER | ACCESS_RESTRICTED_BY_WP

```
public static final long GROUP_ALL = 23089744195615L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | PRIVILEGES | DIRECTORY | PARENT_ID

```
public static final long GROUP_UPDATE_ALL = 23089744191515L
```

ID | LABEL | FLAGS | DESCRIPTION | DATE_M | PRIVILEGES | DIRECTORY | PARENT_ID

```
public static final long ROLE_ALL = 1042337023143967L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | PRIVILEGES | DIRECTORY | HIGHER_ID

```
public static final long ROLE_UPDATE_ALL = 479387069718555L
```

ID | LABEL | FLAGS | DESCRIPTION | DATE_M | PRIVILEGES | DIRECTORY | HIGHER_ID

```
public static final long POSITION_ALL = 18084767253671967L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | ORDER_INDEX | DATA

```
public static final long POSITION_UPDATE_ALL = 18084767253667867L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | ORDER_INDEX | DATA

```
public static final long PORTAL_ALL = 12831L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | FILE_ID

```
public static final long PORTAL_UPDATE_ALL = 8731L
```

ID | LABEL | FLAGS | DESCRIPTION | DATE_M | FILE_ID.

```
public static final long PLANNER_ALL = 3298534895679L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | DEPLOYED | TIMETABLE | DATABASE_ID.

```
public static final long PLANNER_DAY_ALL = 7696581394441L
```

ID | FLAGS | DAY_TIMETABLE | DAY_DATE | DAY_RECURRENT.

```
public static final long DATABASE_ALL = 18154036486221855L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | ADMINISTRATOR_ID | DESCRIPTION_URL | ANONYMOUS_COMMANDS | RESTRICTED_COMMANDS | FULL_COMMANDS | CONN_URL | CRON_EXPRESSION | DATA

```
public static final long DATABASE_UPDATE_ALL = 18154036486217755L
```

ID | LABEL | FLAGS | DESCRIPTION | DATE_M | ADMINISTRATOR_ID | DESCRIPTION_URL | CONN_URL | ANONYMOUS_COMMANDS | RESTRICTED_COMMANDS | FULL_COMMANDS | CRON_EXPRESSION | DATA

```
public static final long STORAGE_ALL = 12319L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M.

```
public static final long RULE_ALL = 9L
```

ID | FLAGS

```
public static final long RESOURCE_ALL = 1099511640607L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | FILE_ID | RES_TYPE | DATE_C | DATE_M

```
public static final long TYPE_ALL = 137438966303L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | FILE_ID | DATE_C | DATE_M | PACKAGE

```
public static final long GRID_ALL = 137438966303L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | FILE_ID | DATE_C | DATE_M | PACKAGE

```
public static final long ATTRIBUTE_ALL = 18014536485318687L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | TYPE_ID | DATE_C | DATE_M | DATA | PACKAGE

```
public static final long ATTRIBUTE_UPDATE_MASK = 536870938L
```

LABEL | FLAGS | DESCRIPTION | TYPE_ID

```
public static final long FORM_ALL = 34235184362047L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | DEPLOYED | FILE_ID | DEPLOYMENT_ID | DEPLOYMENT_STAMP | ICON | DISPLAYPROFILE_ID | LIFECYCLE_RULE_FILEID | ARCHIVE_STORAGE_ID | FRAGMENT_FILE_ID | HIERARCHY_FILE_ID | PACKAGE

```
public static final long FORM_UPDATE_ALL = 45599L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | FILE_ID | ICON

```
public static final long FORM_DEPLOY = 12884901921L
```

ID | DEPLOYED | DEPLOYMENT_ID | DEPLOYMENT_STAMP

```
public static final long FORM_UNDEPLOY = 33L
```

ID | DEPLOYED

```
public static final long PROCESSDEF_ALL = 2673097572364991L
```

ID | LABEL | OWNER_ID | FLAGS | FORM_ID | DESCRIPTION | DATE_C | DATE_M | FILE_ID | DEPLOYED | DEPLOYMENT_ID | DEPLOYMENT_STAMP | DELAY_D | DELAY_X | PRIORITY | REFCOUNT | VERSION_OF | MANAGER_ID | PROCESS_ID_PLANNER | DEPLOYMENT_FORM_ID | SECURITY_PROFILE | PROFILE_ID | DELAY_D_ATTR_ID | DELAY_X_ATTR_ID | PACKAGE

```
public static final long PROCESSDEF_UPDATE_ALL = 2654255479567007L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | FILE_ID | MANAGER_ID | FORM_ID | PROCESS_ID_PLANNER | PRIORITY | ICON | DELAY_D | DELAY_X | SECURITY_PROFILE | PROFILE_ID

```
public static final long PROCESSDEF_DEPLOY = 17605070946337L
```

ID | DEPLOYED | DEPLOYMENT_ID | DEPLOYMENT_STAMP | DEPLOYMENT_FORM_ID

```
public static final long PROCESSDEF_UNDEPLOY = 33L
```

ID | DEPLOYED

```
public static final long PROCESS_MANAGEMENT = 291370702844575L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | FILE_ID | MANAGER_ID | FORM_ID | PRIORITY | PROCESS_ID_PLANNER | ICON | DATE_S | DATE_D | DATE_X | OVERDUE | EXPIRED | PROCESS_STATUS | DEFINITION_ID

```
public static final long SCRIPT_ALL = 3448858751551L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | FILE_ID | SCRIPT_LANG | SCRIPT_TYPE | DEPLOYED | DEPLOYMENT_ID | DEPLOYMENT_STAMP | PACKAGE

```
public static final long SCRIPT_VERSION_ALL = 1080898399156434555L
```

SCRIPT_ALL | VERS_OBJ_COMMENT | VERS_OBJ_INDEX

```
public static final long VIEW_ALL = 1236950626367L
```

ID | LABEL | OWNER_ID | FLAGS | DATE_C | DATE_M | DESCRIPTION | DEPLOYED | DESCRIPTION_URL | ICON | PACKAGE

```
public static final long VIEW_UPDATE_ALL = 1099511672863L
```

ID | LABEL | OWNER_ID | FLAGS | DATE_C | DATE_M | DESCRIPTION | DESCRIPTION_URL | ICON

```
public static final long VIEW_ELEMENT_ALL = 1099511640095L
```

ID | LABEL | OWNER_ID | FLAGS | DATE_C | DATE_M | DESCRIPTION | DISPLAYPROFILE_ID

```
public static final long VIEW_ELEMENT_UPDATE_ALL = 8219L
```

ID | LABEL | FLAGS | DATE_M | DESCRIPTION

```
public static final long VIEW_RUNTIME_TREE = 11L
```

ID | LABEL | FLAGS

```
public static final long VIEW_DEPLOY = 33L
```

ID | DEPLOYED

```
public static final long VIEW_UNDEPLOY = 33L
```

ID | DEPLOYED

```
public static final long ISJOB_ALL = 141733933631L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | DEPLOYED | FILE_ID | DEPLOYMENT_ID | PACKAGE

```
public static final long MODELOCR_ALL = 141733933631L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | DEPLOYED | FILE_ID | DEPLOYMENT_ID | PACKAGE

```
public static final long SECURITY_PROFILE_ALL = 137438965823L
```

ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS | CUSTOM_SECURITY_PROCEDURE | PACKAGE

```
public static final long FOLDER_PROFILE_ALL = 139775415693343L
```

ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS | FP_COLUMNS | FP_OVERRIDDEN_OPTIONS | FP_CHILD_FORM_IDS | FP_CHILD_DEF_FORM_ID | FP_CHILD_DEF_PROFILE_ID | FP_DOC_FORM_IDS | FP_DOC_DEF_FORM_ID | PACKAGE

```
public static final long FOLDER_TREE = 32779L
```

ID | LABEL | FLAGS | ICON

```
public static final long FOLDER_ALL = 20266241273098383L
```

ID | LABEL | FLAGS | ICON | OWNER_ID | FORM_ID | FORM_PAGE | DATE_C | DATE_M | MODIFY_BY | DATE_A | DATE_S | DATE_E | SECURITY_PROFILE | PROFILE_ID

```
public static final long FOLDER_FULL = 20266241273098639L
```

ID | LABEL | FLAGS | ICON | OWNER_ID | FORM_ID | FORM_PAGE | DATE_C | DATE_M | MODIFY_BY | DATE_A | DATE_S | DATE_E | SECURITY_PROFILE | PROFILE_ID | ATTRIBUTES

```
public static final long FOLDER_UPDATE_ALL = 2251842763583626L
```

LABEL | FLAGS | DATE_C | DATE_M | DATE_A | FORM_ID | FORM_PAGE | STATE_ID | DATE_S | DATE_E | SECURITY_PROFILE | PROFILE_ID

```
public static final long FOLDER_UPDATE_FULL = 2251842763583882L
```

LABEL | FLAGS | DATE_C | DATE_M | DATE_A | FORM_ID | FORM_PAGE | ATTRIBUTES | STATE_ID | DATE_S | DATE_E | SECURITY_PROFILE | PROFILE_ID

```
public static final long DOCUMENT_THUMB = 39594266187L
```

ID | LABEL | FLAGS | VERSION | FILE_ID | FILE_TYPE | FILE_SIZE | ICON | LOCKED_BY | DATE_L | VALID_MILLIS | LOCK_TYPE | SECURITY_PROFILE

```
public static final long DOCUMENT_VC_SYSATTRS = 20536414300817326O8L
```

Flag.REQUESTED_BY | Flag.VALIDATED_BY | Flag.DATE_VALIDATION | Flag.DATE_PUBLISHED | Flag.DATE_EXPIRED | Flag.VALIDITY

```
public static final long DOCUMENT_LIST = 18014712984813131L
```

ID | LABEL | FLAGS | VERSION | FILE_ID | FILE_TYPE | FILE_SIZE | ICON | LOCKED_BY | DATE_L | VALID_MILLIS | LOCK_TYPE | SECURITY_PROFILE | DATE_C | DATE_M | OCR_FILE_ID | LINKED_ID | TAGS

```
public static final long DOCUMENT_ALL = 2071656151723605711L
```

ID | LABEL | FLAGS | VERSION | FILE_ID | FILE_TYPE | FILE_SIZE | ICON | LOCKED_BY | DATE_L | VALID_MILLIS | LOCK_TYPE | SECURITY_PROFILE | DATE_C | DATE_M | OCR_FILE_ID | LINKED_ID | TAGS | OWNER_ID | FORM_ID | FORM_PAGE | DATE_A

```
public static final long DOCUMENT_FULL = 2071656151723605967L
```

ID | LABEL | FLAGS | VERSION | FILE_ID | FILE_TYPE | FILE_SIZE | ICON | LOCKED_BY | DATE_L | VALID_MILLIS | LOCK_TYPE | SECURITY_PROFILE | DATE_C | DATE_M | OCR_FILE_ID | LINKED_ID | TAGS | OWNER_ID | FORM_ID | FORM_PAGE | DATE_A | ATTRIBUTES

```
public static final long VERSION_ALL = 36028797018996303L
```

ID | LABEL | OWNER_ID | FLAGS | DATE_C | DATE_M | DATE_A | FILE_ID | FILE_SIZE | FILE_TYPE | VERSION | VERS_OBJ_COMMENT

```
public static final long TRASH_INFO = 4573969445289984L
```

DELETED | DATE_DEL | DELETE_BY

```
public static final long NOTE_ALL = 29189L
```

ID | OWNER_ID | DATE_C | DATE_M | DATE_A | FILE_ID

```
public static final long SEARCHTEMPLATE_ALL = 67121695L
```

LABEL | FLAGS | DATE_C | DATE_M | DESCRIPTION | ID | OWNER_ID | FILE_ID | PRESENTATION

```
public static final long SEARCHTEMPLATE_UPDATEALL = 67117595L
```

LABEL | FLAGS | DATE_M | DESCRIPTION | ID | FILE_ID | PRESENTATION

```
public static final long PROCESS_ALL = 20687354352071567L
```

ID | LABEL | OWNER_ID | FLAGS | DATE_C | DATE_M | DATE_A | DATE_D | DATE_X | DATE_E | DATE_S | FILE_ID | FORM_ID | FORM_PAGE | ATTRIBUTES | PRIORITY | REFCOUNT | OVERDUE | EXPIRED | PROCESS_STATUS | RUNTIME_FILE_ID | RUNTIME_STATE | DEFINITION_ID | MANAGER_ID | PROCESS_ID_PLANNER | PROCESS_START_POINT | DRAFT_FORM_PAGE | SECURITY_PROFILE | PROFILE_ID

```
public static final long TASK_INBOX = 280375567646731L
```

ID | LABEL | FLAGS | DATE_S | DATE_D | DATE_X | PRIORITY | OVERDUE | TASK_ACTORID | TASK_EXECUTORID | TASK_STATUS | TASK_LOCATION | TASK_PROCESS_LABEL | TASK_SUBJECT | TASK_FORM_PAGE | TASK_FORM_ID

```
public static final long TASK_ALL = 2164531577004646 71L
```

ID | LABEL | FLAGS | OWNER_ID | DATE_C | DATE_M | DATE_A | DATE_S | DATE_E | DATE_D | DATE_X | PRIORITY | OVERDUE | EXPIRED | MANAGER_ID | TASK_ROLEID | TASK_GROUPID | TASK_EXECUTORID | TASK_STATUS | TASK_LOCATION | TASK_PROCESS_LABEL | TASK_SUBJECT | TASK_FORM_PAGE | TASK_FORM_ID | FLOW_MODE

```
public static final long USER_PROFILE_ALL = 1099511640095L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | PARENT_ID

```
public static final long JASY_ALL = 69269232758799L
```

ID | LABEL | OWNER_ID | FLAGS | DATE_C | DATE_M | DATE_S | DATE_E | JASY_TYPE | JASY_STATUS | JASY_PROG_PERCENT | JASY_PROG_STATUS | JASY_EMAIL | JASY_CONFIG

```
public static final long COMPOSITION_RULE_ALL = 150323868223L
```

ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS | DEPLOYED | FILE_ID | DEPLOYMENT_ID | DEPLOYMENT_STAMP| PACKAGE

```
public static final long
COMPOSITION_RULE_VERSION_ALL = 108086391056904735L
```

ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS | FILE_ID | VERS_OBJ_COMMENT | VERS_OBJ_INDEX

```
public static final long FORM_DISPLAY_PROFILE_ALL = 2251799813685377L
```

ID | FORM_ID | PROFILE_ID

```
public static final long FORM_VERSION_ALL = 1081206262412539 51L
```

FORM_ALL | VERS_OBJ_COMMENT | VERS_OBJ_INDEX

```
public static final long PROCESSDEF_VERSION_ALL = 110740659492631231L
```

ID | LABEL | OWNER_ID | FLAGS | FORM_ID | DESCRIPTION | DATE_C | DATE_M | FILE_ID | DEPLOYED | DEPLOYMENT_ID | DEPLOYMENT_STAMP | DELAY_D | DELAY_X | PRIORITY | REFCOUNT | MANAGER_ID |

PROCESS_ID_PLANNER | SECURITY_PROFILE | PROFILE_ID | VERS_OBJ_COMMENT | VERS_OBJ_INDEX

```
public static final long ISJOB_VERSION_ALL = 108086391056904767L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | DEPLOYED | FILE_ID | VERS_OBJ_COMMENT | VERS_OBJ_INDEX

```
public static final long MODELOCR_VERSION_ALL = 108086391056904767L
```

ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | DEPLOYED | FILE_ID | VERS_OBJ_COMMENT | VERS_OBJ_INDEX

```
public static final long VIEW_VERSION_ALL = 108087490568532031L
```

ID | LABEL | OWNER_ID | FLAGS | DATE_C | DATE_M | DESCRIPTION | DEPLOYED | DESCRIPTION_URL | VERS_OBJ_COMMENT | VERS_OBJ_INDEX

```
public static final long COMMENT_ALL = 7696581406735L
```

ID | LABEL | FLAGS | DATE_C | DATE_M | OWNER_ID | OWNER_NAME | PARENT_ID | COMMENT_CONTENT

```
public static final long COMMENT_UPDATE_ALL = 6597069774858L
```

LABEL | FLAGS | DATE_M | OWNER_NAME | COMMENT_CONTENT

```
public static final long MAIL_TEMPLATE_ALL = 137438966303L
```

ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS | FILE_ID

```
public static final long DEF_FILE_ID = 1099511627776L
```

Specifies the DocumentTemplate#getDefFileId() property).

```
public static final long DOCUMENT_TEMPLATE_ALL = 1236950594079L
```

ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS | FILE_ID | DEF_FILE_ID | PACKAGE

```
public static final long FOR_FOLDERS = 16777216L
```

Form#getFlags(): Specifies form can be set on folders. Can be combined with #FOR_PROCESSES|#FOR_DOCUMENTS|#FOR_SEARCH in this case.

ViewElement#getFlags(): Specifies view element will render on folders. Can be combined with #FOR_PROCESSES but not #FOR_DOCUMENTS in this case.

```
public static final long FOR_PROCESSES = 33554432L
```

Form#getFlags(): Specifies form can be set on processes. Can be combined with #FOR_FOLDERS|#FOR_DOCUMENTS|#FOR_SEARCH in this case.

ViewElement#getFlags(): Specifies view element will render on processes. Can be combined with #FOR_FOLDERS but not #FOR_DOCUMENTS in this case.

```
public static final long FOR_DOCUMENTS = 67108864L
```

Form#getFlags(): Specifies form can be set on documents. Can be combined with #FOR_PROCESSES|#FOR_FOLDERS|#FOR_SEARCH in this case.

ViewElement#getFlags(): Specifies view element will render on folders. Cannot be combined with neither #FOR_PROCESSES nor #FOR_FOLDERS in this case.

```
public static final long FOR_SEARCH = 134217728L
```

Form#getFlags(): Specifies form can be set on folders. Can be combined with #FOR_PROCESSES|#FOR_DOCUMENTS|#FOR_SEARCH in this case.

```
public static final long HIDE_IN_SEARCH = 268435456L
```

Form#getFlags(): Specifies form cannot be used as value in search conditions. Can be combined with #FOR_PROCESSES|#FOR_FOLDERS|#FOR_DOCUMENTS in this case.

```
public static final long CLASSES_MASK = 2399141888L
```

Bit-mask for ViewElement#getFlags()

```
public static final long IS_OBJECTS = 134217728L
```

ViewElement#getFlags(): specifies that view element renders to actual objects (folders, document, processes)

```
public static final long SHOW_SUBFOLDERS = 2147483648L
```

ViewElement#getFlags(): specifies that rendered class element shall display the real folders in the tree. Applies only to view elements containing 'FOR_FOLDERS', otherwise this should be signaled as error by compiler

```
public static final long CLASS_SORT_MASK = 1879048192L
```

Bit mask for criteria sorting specified in ViewElement#getFlags()

```
public static final long ORDER_ASC_1 = 0L
```

ViewElement#getFlags(): order by first unicity criterion, ascending (by default)

```
public static final long ORDER_DESC_1 = 268435456L
```

ViewElement#getFlags(): order by first unicity criterion, descending

```
public static final long ORDER_ASC_2 = 0L
```

ViewElement#getFlags(): order by second unicity criterion, ascending (by default)

```
public static final long ORDER_DESC_2 = 536870912L
```

ViewElement#getFlags(): order by second unicity criterion, descending

```
public static final long ORDER_ASC_3 = 0L
```

ViewElement#getFlags(): order by third unicity criterion, ascending (by default)

```
public static final long ORDER_DESC_3 = 1073741824L
```

ViewElement#getFlags(): order by third unicity criterion, descending (by default)

```
public static final long ROOT_VE = 8796093022208L
```

ViewElement#getFlags(): root* view element

```
public static final long HAS_CLASSES = 4294967296L
```

ClassificationElement#getFlags(): Specifies item has child elements to render.

```
public static final long HAS_DOCUMENTS = 8589934592L
```

ClassificationElement#getFlags(): Specifies item has document list render.

```
public static final long IS_FOLDER = 17179869184L
```

ClassificationElement#getFlags(): Specifies item is actual folder.

```
public static final long IS_PROCESS = 34359738368L
```

ClassificationElement#getFlags(): Specifies item is actual process.

```
public static final long ADD_ENABLED = 68719476736L
```

ClassificationElement#getFlags(): Specifies documents can be added in item.

```
public static final long HAS_FULLTEXT = 1L
```

Database#getFlags(): Database has fulltext capability (bit 0)

```
public static final long SECURITY_MODE_MASK = 6L
```

Database#getFlags(): 2-bit mask for the database security support (bits 1 and 2)

```
public static final long SECURITY_MODE_NONE = 0L
```

Database#getFlags(): Security mode 0, database has no security support

```
public static final long SECURITY_MODE_SIMPLE = 2L
```

Database#getFlags(): Security mode 2 (1 << 1), database has simple security support

```
public static final long SECURITY_MODE_MEDIUM = 4L
```

Database#getFlags(): Security mode 4 (2 << 1), database has medium security support

```
public static final long SECURITY_MODE_COMPLEX = 6L
```

Database#getFlags(): Security mode 6 (3 << 1), database has complex security support

```
public static final long IS_PUBLIC = 8L
```

Database#getFlags(): database is public (bit 3).

View#getFlags(): view is public.

SearchTemplate#getFlags(): search template is public.

```
public static final long DATABASE_RESTRICTED_ARCHIVE = 64L
```

Database#getFlags(): Database restricted archive (bit 6) - Only DBA users can perform delete in archive.

```
public static final long DATABASE_OFFLINE = 16384L
```

Database#getFlags(): Database offline (bit 13)

```
public static final long DATABASE_WORKFLOW_INACTIVE = 32768L
```

Database#getFlags(): Workflow paused on database (bit 14)

```
public static final long DATABASE_LIFECYCLE_INACTIVE = 131072L
```

Database#getFlags(): Lifecycle jobs paused on database (bit 15)

```
public static final long DATABASE_LIFECYCLE_DEBUG = 262144L
```

Database#getFlags(): Lifecycle is in debug mode, as in it does not actually archive/delete objects, only dumps rendered result to console (bit 16)

```
public static final long DATABASE_UNAVAILABLE = 4611686018427387904L
```

Database#getFlags(): Database schema is external and is unavailable (bit 17)

```
public static final long ENABLE_COMMENTS = 536870912L
```

Database#getFlags(): specifies if forum is enabled on a database

```
public static final long ENABLE_TRASH = 17179869184L
```

Database#getFlags(): specifies if recycle bin is enabled on a database

```
public static final long DISPLAY_TREE_COUNTERS = 34359738368L
```

Database#getFlags(): specifies if counters value is displayed in tree

```
public static final long
DATABASE_LUCENE_SIMPLESEARCH = 72057594037927936L
```

Database#getFlags(): Database has lucene simple search capability

```
public static final long
DATABASE_BUILD_LUCENE_SIMPLESEARCH = 144115188075855872L
```

Database#getFlags(): lcn_ss reindexing database in progress...

```
public static final long
DATABASE_LUCENE_CONTENTSEARCH = 288230376151711744L
```

Database#getFlags(): Database has lucene content search capability

```
public static final long
DATABASE_BUILD_LUCENE_CONTENTSEARCH = 576460752303423488L
```

Database#getFlags(): lcn_content reindexing database in progress...

```
public static final long FULLTEXT_ACCENT_INSENSITIVE = 4503599627370496L
```

Database#getFlags(): accent insensitive solr index (index type)

```
public static final long FULLTEXT_ACCENT_SENSITIVE = 9007199254740992L
```

Database#getFlags(): accent sensitive solr index (index type)

```
public static final long FULLTEXT_ACCENT_BOTH = 13510798882111488L
```

Database#getFlags(): accent sensitive and insensitive solr index (query time selection) (index type)

```
public static final long IS_DEFAULT = 32L
```

Planner#getFlags(): specifies this planner is the server default planner.

Portal#getFlags(): specifies this portal is the server default portal.

```
public static final long FOLDER_HAS_ATTRIBUTES = 1L
```

Folder#getFlags(): specifies folder has custom attributes which are not empty

```
public static final long FOLDER_HAS_NOTES = 2L
```

Folder#getFlags(): specifies folder has notes, useful to add a ‘note’ symbol to the folder icon, wherever it is displayed.

```
public static final long FOLDER_HAS_CHILDREN = 4L
```

Folder#getFlags(): specifies folder has sub-folders, useful to add a '+' sign to the folder tree item.

```
public static final long HAS_COMMENTS = 32768L
```

Folder#getFlags(): specifies folder has comments, useful to add a 'comments' symbol to the folder icon, wherever it is displayed.

Document#getFlags(): specifies document has comments, useful to add a 'comments' symbol to the document icon, wherever it is displayed.

Process#getFlags(): specifies process has comments, useful to add a 'comments' symbol to the process icon, wherever it is displayed.

```
public static final long HAS_NOTIFICATIONS = 262144L
```

Folder#getFlags(): specifies folder has notifications, useful to add a 'notification' symbol to the folder icon, wherever it is displayed.

Document#getFlags(): specifies document has notifications, useful to add a 'notification' symbol to the document icon, wherever it is displayed.

Process#getFlags(): specifies process has notifications, useful to add a 'notification' symbol to the process icon, wherever it is displayed.

```
public static final long FORM_ATTR_MANDATORY = 1L
```

(Long) Assignment#getData(): specifies attribute is mandatory in context of form.

```
public static final long FORM_ATTR_LINKED = 2L
```

(Long) Assignment#getData(): specifies attribute is linked to another attribute in context of form.

```
public static final long FORM_ATTR_LOCKED = 4L
```

(Long) Assignment#getData(): specifies attribute is locked, in context of form can not have a "keep value" checkbox.

```
public static final long FORM_ATTR_XNORMAL = 0L
```

(Long) Assignment#getData(): specifies the list attribute is normal, has no special behavior.

```
public static final long FORM_ATTR_XSIMPLE = 16L
```

(Long) Assignment#getData(): specifies the list attribute is plain, as in the hierarchy value is irrelevant, all items are considered on same level and are represented as a linear list.

```
public static final long FORM_ATTR_XFULLPATH = 32L
```

(Long) Assignment#getData(): specifies the list attribute is 'full path', as in items are specified with all

parent items in the hierarchy

```
public static final long FORM_ATTR_XMULTPILE = 48L
```

(Long) Assignment#getData(): specifies the list attribute is 'multiple selection', as in the attribute value can contain more than one value from the list.

```
public static final long FORM_ATTR_XCATEGORY = 64L
```

(Long) Assignment#getData(): specifies the list attribute is 'category type', as in the attribute value can contain any value from the list except first level values.

```
public static final long FORM_ATTR_XMULTILINE = 80L
```

(Long) Assignment#getData(): bit mask for the above properties, since they are exclusive.

```
public static final long FORM_ATTR_XRADIO = 96L
```

(Long) Assignment#getData(): specifies the list attribute is a list of radio buttons

```
public static final long FORM_ATTR_XRADIO_INLINE = 112L
```

(Long) Assignment#getData(): specifies the list attribute is a list of inline radio buttons

```
public static final long FORM_ATTR_XMASK = 240L
```

(Long) Assignment#getData(): bit mask for the above properties, since they are exclusive.

```
public static final long PROFILE_DEFAULT = 1L
```

UserProfile#getFlags(): specifies it is the default user profile.

```
public static final long PROFILE_ANONYMOUS = 2L
```

UserProfile#getFlags(): specifies it is the user profile for anonymous connections

```
public static final long PROFILE_EXTERNAL_EMAIL = 4L
```

UserProfile#getFlags(): specifies it is the user profile for external email users connections

```
public static final long DOCSTAT_ATTRIBUTES = 1L
```

Document#getFlags(): document has custom attributes which are not empty.

```
public static final long DOCSTAT_CDARCHIVED = 4L
```

Document#getFlags(): document file is stored on optical media.

DocumentVersion#getFlags(): document file is stored on optical media.

```
public static final long DOCSTAT_LOCK_OTHER = 8L
```

Document#getFlags(): document is locked to user other than one accessing the document. This flag is connection dependent.

```
public static final long DOCSTAT_LOCK_MYSELF = 16L
```

Document#getFlags(): document is locked to user accessing the document. This flag is connection dependent.

```
public static final long DOCSTAT_LOCKED = 32L
```

Document#getFlags(): document is locked for changes.

```
public static final long DOCSTAT_NOTES = 64L
```

Document#getFlags(): document has notes.

```
public static final long DOCSTAT_FULLTEXT = 256L
```

Document#getFlags(): document has indexed content.

```
public static final long DOCSTAT_HAS_VERSIONS = 512L
```

Document#getFlags(): document has versions.

```
public static final long DOCSTAT_SYS_ATTRIBUTES = 1024L
```

Document#getFlags(): search hit in document's system attributes. This flag is relevant in the context of search.

```
public static final long DOCSTAT_COPYOFDOC = 2048L
```

Document#getFlags(): document is copy of another document specified by Document#getLinkedId().

```
public static final long DOCSTAT_REFTODOC = 4096L
```

Document#getFlags(): document is a reference of another document specified by Document#getLinkedId().

```
public static final long DOCSTAT_PDF_HAS_TEXT = 8192L
```

Document#getFlags(): document content contains text blocks. May indicate that document has been converted to text. Relevant for PDF documents.

```
public static final long DOCSTAT_0000000000004000_BIT_AVAILABLE = 16384L
```

Document#getFlags(): unused

```
public static final long DOCSTAT_HAS_COMMENTS = 32768L
```

Document#getFlags(): document has comments

```
public static final long DOCSTAT_HAS_NOTIFICATIONS = 262144L
```

Document#getFlags(): document has notifications

```
public static final long USER_IS_EXTERNAL_LDAP = 1L
```

User#getFlags(), Role#getFlags(), Group#getFlags(): user/role/group is imported from an external directory.

```
public static final long USER_IS_SERVER_ACCOUNT = 2L
```

User#getFlags(): this is a sever account

```
public static final long USER_SALUTATION_MASK = 12L
```

User#getFlags(): this is a sever account

```
public static final long USER_SALUTATION_NONE = 0L
```

User#getFlags(): no salutation/unknown

```
public static final long USER_SALUTATION_MR = 4L
```

User#getFlags(): mister/monsieur

```
public static final long USER_SALUTATION_MRS = 8L
```

User#getFlags(): mistress/madame

```
public static final long USER_SALUTATION_MS = 12L
```

User#getFlags(): miss/mademoiselle

```
public static final long USER_LOCKED = 4294967296L
```

User#getFlags(): user's account is locked (explicitly by admin or due to successive failed login attempts)

```
public static final long USER_CHANGEPASS = 8589934592L
```

User#getFlags(): user must change pass on next login

```
public static final long USER_IS_EXTERNAL_EMAIL = 17179869184L
```

User#getFlags(): user is external (identified by email)

```
public static final long USER_DENY_LOGIN = 34359738368L
```

User#getFlags(): user cannot directly login to GARGANTUA

```
public static final long ROLE_IS_ATTACHED = 2L
```

Role#getFlags(): role is attached to the role set as authority (like secretary)

```
public static final long POSITION_WF_MASTER_ACCESS = 4L
```

Position#getFlags(): position has access to it's master-s inbox (e.g. secretary can access director's task inbox)

```
public static final long POSITION_CAN_SIGN = 8L
```

Position#getFlags(): position has the right to sign public/official documents

```
public static final long ROLE_LAYOUT_MASK = 65280L
```

Role#getFlags(): role sub-hierarchy layout mask. Layout possible values are (0-0xFF) << 8.

```
public static final long CAN_CANCEL = 256L
```

Task#getFlags(): task can be canceled by user. In the context of a dynamic process it also means that the process will be canceled if the task is canceled.

```
public static final long CAN_FINISH = 512L
```

Task#getFlags(): task can be the last in a dynamic process thread and allows for no subsequent tasks if not defined. This does not imply that the process will stop if it still has active threads.

```
public static final long CAN_STOP = 1024L
```

Task#getFlags(): task can be the last in a dynamic process thread and will end all other active threads (and consequently all acive tasks) if any.

```
public static final long FLOW_VIEW = 4096L
```

Task#getFlags(): allow access to view the flow control path.

```
public static final long FLOW_EDIT = 8192L
```

Task#getFlags(): allow access to change the flow control path.

```
public static final long IS_SHARED_TASK = 262144L
```

Task#getFlags(): task is shared among users assigned to task role.

```
public static final long WAIT_SHARED_TASK_FINISH = 524288L
```

Task#getFlags(): wait for all users to finish the shared task.

```
public static final long NO_QUIT_TASK = 1048576L
```

Task#getFlags(): task cannot be quit by user, only manager can re-assign task.

```
public static final long IGNORE_FAIL = 2097152L
```

Task#getFlags(): assume system task finished gracefully even if execution failed.

```
public static final long MANDATORY_USE_OF_PROCESS_PLANNER = 4194304L
```

Task#getFlags(): task delays are rendered using the process planner not the assigned user's planner

```
public static final long KEEP_VALUES_ON_RESTART = 8388608L
```

Task#getFlags(): keep values on restart option for poll tasks.

```
public static final long ALLOW_ESCALATE = 16777216L
```

Task#getFlags(): allow assigned user to escalate task to higher role/position.

```
public static final long OVERDUE_BY_TASK = 512L
```

Process#getFlags(): process is marked overdue because of task who is past it's due date

```
public static final long EXPIRING_BY_TASK = 1024L
```

Process#getFlags(): process is marked as expiring because of task who is close (less than 1 day) to it's expiry date.

```
public static final long DELETE_ON_FINISH = 4096L
```

Process#getFlags(): process should be automatically deleted when it is finished.

ProcessDefinition#getFlags(): instances should be automatically deleted when they are finished.

```
public static final long DELETE_ON_CANCEL = 8192L
```

Process#getFlags(): process should be automatically deleted when it is canceled.

ProcessDefinition#getFlags(): instances should be automatically deleted when they are canceled.

```
public static final long DELETE_ON_EXPIRE = 16384L
```

Process#getFlags(): process should be automatically deleted when it expires.

ProcessDefinition#getFlags(): instances should be automatically deleted when they expire.

```
public static final long ATTR_ADVANCED_SEARCH = 268435456L
```

Attribute#getFlags(): hide the attribute in advanced search.

```
public static final long ATTR_DATABASE_INDEX_PENDING = 536870912L
```

Attribute#getFlags(): there is an index operation using this attribute currently running on the database.

```
public static final long ATTR_DATABASE_INDEX = 1073741824L
```

Attribute#getFlags(): this attribute is indexed to speed up search using it as criteria.

```
public static final long ATTR_CAPITAL_LETTERS = 2147483648L
```

Attribute#getFlags(): this attribute contains only capital letters

```
public static final long VIRTUAL = 36028797018963968L
```

ArchiveFolder#getFlags(): the folder is a virtual place-holder for it's corresponding active object.

```
public static final long NO_CHILDREN_ALIVE = 2147483648L
```

ArchiveFolder#getFlags(): the folder has only the children in the trash or is empty.

```
public static final long HAS_VIRTUAL = 72057594037927936L
```

Folder#getFlags(): the folder has a virtual place-holder corresponding in the archive.

```
public static final long IS_LADRAD = 64L
```

ModelOCR#getFlags(): the OCR model is result of a FlexiCapture import

```
public static final long USER_PLANNER = 64L
```

Planner#getFlags(): the planner is a user's own custom planner.

```
public static final long PROCESS_PLANNER = 128L
```

Planner#getFlags(): the planner is a process's own custom planner.

```
public static final long VERSION_PLANNER = 256L
```

Planner#getFlags(): the planner is a process version's own custom planner.

```
public static final long REORDER_LABEL_ASC = 0L
```

'Folder sort' operation flag. Means sort folders by label alphabetically ascending.

```
public static final long REORDER_LABEL_DESC = 1L
```

'Folder sort' operation flag. Means sort folders by label alphabetically descending.

```
public static final long EMAIL_NOTIFICATION = 1L
```

Comment#getFlags(): Notify author of reply to the comment.

```
public static final long COMMENT_IS_CLOSED = 2L
```

Comment#getFlags(): Specifies if the comment is closed.

```
public static final long CERT_ISSUED_BY = 1099511627776L
```

Certificate: Bit 40: Specifies the Certificate#getAuthority() property.

```
public static final long CERT_SERVER = 2199023255552L
```

Certificate: Bit 41: Specifies the Certificate#isOnServer() property.

```
public static final long CERT_FLAG_VALID_2 = 4398046511104L
```

Certificate: Bit 42: Specifies the Certificate#getValid2() property.

```
public static final long CERT_IDENTITY_PREFIX = 17592186044416L
```

Certificate: Bit 44: Specifies the 'prefix' db column of the identity field.

```
public static final long CERT_IDENTITY_N0 = 35184372088832L
```

Certificate: Bit 45: Specifies the 'numeric0' db column of the identity field.

```
public static final long CERT_IDENTITY = 52776558133248L
```

Certificate: `CERT_IDENTITY_PREFIX | CERT_IDENTITY_N0` : Specifies the Certificate#getIdentityId() property.

```
public static final long CERT_GEN_CACERT = 70368744177664L
```

Certificate: Bit 46: Specifies the Certificate#getGenCACert() property.

```
public static final long CERTIFICATE_ALL = 139637976740415L
```

Certificate: `ID | LABEL | OWNER_ID | FLAGS | DESCRIPTION | DATE_C | DATE_M | FILE_ID | DEPLOYED | CERT_ISSUED_BY | CERT_FLAG_VALID_2 | CERT_IDENTITY | PASSWORD | CERT_SERVER | CERT_GEN_CACERT`

```
public static final long IS_JOB_SCAN = 1L
```

IsJob#getFlags(): Scan IS job type.

```
public static final long IS_JOB_IMPORT = 2L
```

IsJob#getFlags(): Import IS job type.

```
public static final long IS_JOB_EMAIL = 4L
```

IsJob#getFlags(): Email IS job type.

```
public static final long IS_JOB_LAD_RAD = 256L
```

IsJob#getFlags(): IS Job with lad rad model.

```
public static final long IS_JOB_SIGN = 512L
```

IsJob#getFlags(): IS Job with sign.

```
public static final long IS_JOB_SIGN_MANDATORY = 1024L
```

IsJob#getFlags(): IS Job with mandatory sign.

```
public static final long SIMPLE_DEBUG = 8388608L
```

Process#getFlags(): process is running is simple debug mode. This means that all tasks come back to the user that started the process.

ProcessDefinition#getFlags(): instances will be run in simple debug mode.

```
public static final long ENABLE_DOCUMENT_ACCESS = 16777216L
```

Process#getFlags(): process allows external access from other instances to its documents. This means that documents are accessible through security profile rather than the restrictive derfault mode which allows only the manager and the current task executors to access the documents

ProcessDefinition#getFlags(): instances will allow access to documents based on security profile.

```
public static final long HAS_PLUGIN_OPTIONS = 33554432L
```

Process#getFlags(): specifies if a process has custom options that will be used in process review plugin

ProcessDefinition#getFlags(): instances will make use of this flag

```
public static final long
NOTIFICATION_DEFINITION_ALL = 18014669092434057L
```

```
ID | FLAGS | DATE_C | DATE_M | NOTIF_TYPE | NOTIF_EVENT | FORM_ID |
NOTIF_MAIL_TEMPLATE_ID | NOTIF_RECIPIENT_ID | NOTIF_OBJECT_ID |
NOTIF_RECURSIVE | DATA
```

```
public static final long
NOTIFICATION_DEFINITION_FULL = 18022915429642377L
```

```
NOTIFICATION_ALL | NOTIF_TIMESTAMP | NOTIF_EXECUTOR_ID |
NOTIF_EXECUTOR_LABEL | NOTIF_EXECUTOR_REAL_NAME | DATA
```

```
public static final long NOTIFICATION_ALL = 52630529257609L
```

```
ID | FLAGS | DATE_C | DATE_M | NOTIF_EVENT | NOTIF_MAIL_TEMPLATE_ID |
NOTIF_RECIPIENT_ID | NOTIF_OBJECT_ID | NOTIF_OBJECT_LABEL | FORM_ID |
NOTIF_FORM_LABEL | NOTIF_TIMESTAMP | NOTIF_EXECUTOR_ID |
NOTIF_EXECUTOR_LABEL | NOTIF_EXECUTOR_REAL_NAME | NOTIF_PROCESSED
```

```
public static final long NOTIFICATION_FULL = 70222715302025L
```

```
NOTIFICATION_ALL | NOTIF_OBJECT_DATA
```

```
public static final long FAVORITE_ALL = 520205L
```

```
ID | DATE_C | DATE_M | FLAGS | OWNER_ID | FAVORITE_OBJECT_ALL |
FAVORITE_POSITION
```

```
public static final long VC_FLAG_START = 1L
```

ValidationCycle#getFlags(): specifies validationcycle has active start stage .

```
public static final long VC_FLAG_MODIF = 2L
```

ValidationCycle#getFlags(): specifies validationcycle has active modification stage .

```
public static final long VC_FLAG_AVIZ = 4L
```

ValidationCycle#getFlags(): specifies validationcycle has active avization stage .

```
public static final long VC_FLAG_VALID = 8L
```

ValidationCycle#getFlags(): specifies validationcycle has active validation stage .

```
public static final long VC_FLAG_PUBLIC = 16L
```

ValidationCycle#getFlags(): specifies validationcycle has active publication stage .

```
public static final long VC_FLAG_GED = 32L
```

ValidationCycle#getFlags(): specifies validationcycle has active ged (read) stage .

```
public static final long VC_FLAG_BROADCAST = 65536L
```

ValidationCycle#getFlags(): specifies validationcycle broadcast mode

```
public static final long PACKAGE_ALL = 4294979615L
```

ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS

```
public static final long SECURITY_PROFILE_FOR_WORKSPACE = 1L
```

SecurityProfile#getFlags(): SecurityProfile for workspace (workspace securityprofile).

```
public static final long OPINION_SET_BY_SUPERVISOR = 1L
```

This flag indicates that the value of the ‘opinion’ type attribute has been set by a supervisor.

```
public static final long TYPE_LOCALIZED = 1L
```

This flag indicates that the given type is localized.

```
public static final long TYPE_INDEXED = 2L
```

This flag indicates that the given type is indexed (grids only).

```
public static final long TYPE_RESTRICT_DUPLICATES = 4L
```

This flag indicates that the given type do not allow any duplicate values

```
public static final long ENABLE_TRANSLATIONS = 1L
```

This flag indicates that the given object(mail template, document template, composition rules) is

localized.

```
public static final long MAIL_TEMPLATE_TYPE_MASK = 14L
```

MailTemplate#getFlags(): 3-bit mask for the mail template type (bits 1 and 2 and 3)

```
public static final long MAIL_TEMPLATE_FOR_PROCESS = 2L
```

MailTemplate#getFlags(): Mail template type 2 (`1 << 1`), mail template for process

```
public static final long MAIL_TEMPLATE_FOR_VCYCLE = 4L
```

MailTemplate#getFlags(): Mail template type 4 (`2 << 1`), mail template for validation cycle

```
public static final long MAIL_TEMPLATE_FOR_GENERAL_PURPOSE = 8L
```

MailTemplate#getFlags(): Mail template type 8 (`3 << 1`), mail template for general purpose

```
public static final long ENABLE_HTML = 16L
```

This flag indicates that the given object(mail template) has the body content in html format.

```
public static final long PROCESS_TESTING_DEFINITION_ALL = 2097151L
```

```
ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS |
FILE_ID | PROCESS_TESTING_CREATED_BY |
PROCESS_TESTING_CREATED_BY_LABEL | PROCESS_TESTING_CREATED_BY_REALNAME
| PROCESS_TESTING_MODIFIED_BY | PROCESS_TESTING_MODIFIED_BY_LABEL |
PROCESS_TESTING_MODIFIED_BY_REALNAME |
PROCESS_TESTING_PROCESS_DEFINITION_ID |
PROCESS_TESTING_DEFINITION_TOTAL_RUNS |
PROCESS_TESTING_DEFINITION_SUCCESS_RUNS |
PROCESS_TESTING_DEFINITION_FAIL_RUNS |
PROCESS_TESTING_DEFINITION_LAST_SUCCESS |
PROCESS_TESTING_DEFINITION_LAST_FAIL |
PROCESS_TESTING_DEFINITION_LAST_DURATION
```

```
public static final long PROCESS_TESTING_INSTANCE_ALL = 534806527L
```

```
ID | LABEL | DESCRIPTION | DATE_C | DATE_M | OWNER_ID | FLAGS |
FILE_ID | PROCESS_TESTING_CREATED_BY |
PROCESS_TESTING_CREATED_BY_LABEL | PROCESS_TESTING_CREATED_BY_REALNAME
| PROCESS_TESTING_MODIFIED_BY | PROCESS_TESTING_MODIFIED_BY_LABEL |
PROCESS_TESTING_MODIFIED_BY_REALNAME |
PROCESS_TESTING_PROCESS_DEFINITION_ID |
PROCESS_TESTING_INSTANCE_STARTED_BY_ID |
PROCESS_TESTING_INSTANCE_STARTED_BY_LABEL |
PROCESS_TESTING_INSTANCE_STARTED_BY_REALNAME |
```

```
PROCESS_TESTING_INSTANCE_STARTED_AT |
PROCESS_TESTING_INSTANCE_FINISHED_BY_ID |
PROCESS_TESTING_INSTANCE_FINISHED_BY_LABEL |
PROCESS_TESTING_INSTANCE_FINISHED_BY_REALNAME |
PROCESS_TESTING_INSTANCE_FINISHED_AT
```

PREFIX

```
public class Prefix
```

The Prefix class provides support for object identification; it contains all prefixes used for object IDs.

Field summary :

Modifier and type	Field
public static final java.lang.String	FILE
public static final java.lang.String	USER
public static final java.lang.String	USER_DELEGATION
public static final java.lang.String	ROLE_DELEGATION
public static final java.lang.String	USER_PROFILE
public static final java.lang.String	GROUP
public static final java.lang.String	ROLE
public static final java.lang.String	MANAGER
public static final java.lang.String	GROUP_USER
public static final java.lang.String	USER_ROLE
public static final java.lang.String	GROUP_ROLE
public static final java.lang.String	MANAGER_USER
public static final java.lang.String	FORM_DISPLAY_PROFILE
public static final java.lang.String	DATABASE
public static final	GROUP_DATABASE

java.lang.String	
public static final java.lang.String	USER_DATABASE
public static final java.lang.String	ROLE_DATABASE
public static final java.lang.String	STORAGE_AREA
public static final java.lang.String	DATABASE_STORAGE_RULE
public static final java.lang.String	GROUP_PORTAL
public static final java.lang.String	USER_PORTAL
public static final java.lang.String	ROLE_PORTAL
public static final java.lang.String	PLANNER
public static final java.lang.String	PLANNER_DAY
public static final java.lang.String	RESOURCE
public static final java.lang.String	SCRIPT
public static final java.lang.String	SECURITY_PROFILE
public static final java.lang.String	MAIL_TEMPLATE
public static final java.lang.String	DOCUMENT_TEMPLATE
public static final java.lang.String	FOLDER_PROFILE
public static final java.lang.String	COMPOSITION_RULE
public static final java.lang.String	SEARCH_TEMPLATE
public static final java.lang.String	USER_SEARCHTEMPLATE

public static final java.lang.String	GROUP_SEARCHTEMPLATE
public static final java.lang.String	ROLE_SEARCHTEMPLATE
public static final java.lang.String	ASYNC_JOB
public static final java.lang.String	EXTENDED_PROPERTY
public static final java.lang.String	OBJECT_EXTENDED_PROPERTY
public static final java.lang.String	CERTIFICATE
public static final java.lang.String	TYPE
public static final java.lang.String	GRID
public static final java.lang.String	ATTRIBUTE
public static final java.lang.String	FORM
public static final java.lang.String	FORM_ATTRIBUTE
public static final java.lang.String	VIEW
public static final java.lang.String	VIEW_ELEMENT
public static final java.lang.String	GROUP_VIEW
public static final java.lang.String	USER_VIEW
public static final java.lang.String	ROLE_VIEW
public static final java.lang.String	PROCESS_DEF
public static final java.lang.String	SCRIPT2
public static final	SCRIPT2_VERSION

java.lang.String	
public static final java.lang.String	COMPOSITION_RULE_VERSION
public static final java.lang.String	FORM_VERSION
public static final java.lang.String	PROCESS_DEF_VERSION
public static final java.lang.String	IS_JOB_VERSION
public static final java.lang.String	MODEL_OCR_VERSION
public static final java.lang.String	VIEW_VERSION
public static final java.lang.String	VIEW_ELEMENT_VERSION
public static final java.lang.String	IS_JOB
public static final java.lang.String	MODEL_OCR
public static final java.lang.String	GROUP_ISJOB
public static final java.lang.String	ROLE_ISJOB
public static final java.lang.String	USER_ISJOB
public static final java.lang.String	CLASS_ELEMENT
public static final java.lang.String	PROCESS
public static final java.lang.String	TASK
public static final java.lang.String	FOLDER
public static final java.lang.String	FOLDER_VALIDATIONCYCLE
public static final java.lang.String	FOLDER_MYDOCS

public static final java.lang.String	FOLDER_WORKSPACE
public static final java.lang.String	FOLDER_IN_WORKSPACE
public static final java.lang.String	PIN_DOCF
public static final java.lang.String	PIN_DOCP
public static final java.lang.String	PIN_DOCW
public static final java.lang.String	FILE_DOCF
public static final java.lang.String	FILE_DOCP
public static final java.lang.String	FICHE_DOCF
public static final java.lang.String	FICHE_DOCD
public static final java.lang.String	FICHE_DOCV
public static final java.lang.String	FICHE_DOCW
public static final java.lang.String	FILE_DOCD
public static final java.lang.String	FILE_DOCV
public static final java.lang.String	FILE_DOCW
public static final java.lang.String	FICHE_DOCP
public static final java.lang.String	REF_DOCF
public static final java.lang.String	REF_DOCP
public static final java.lang.String	REF_FICHEF
public static final	REF_FICHEP

java.lang.String	
public static final java.lang.String	FOLDER_NOTE
public static final java.lang.String	PROCESS_NOTE
public static final java.lang.String	DOC_NOTE_BASE
public static final java.lang.String	FICHEF_NOTE
public static final java.lang.String	DOCP_NOTE
public static final java.lang.String	FICHEP_NOTE
public static final java.lang.String	DOCF_VERSION
public static final java.lang.String	DOCP_VERSION
public static final java.lang.String	TASK_DEF
public static final java.lang.String	START_DEF
public static final java.lang.String	TASK_GROUP_DEF
public static final java.lang.String	DELEGATION
public static final java.lang.String	ARCHIVE_FOLDER
public static final java.lang.String	ARCHIVE_PIN
public static final java.lang.String	ARCHIVE_FILE
public static final java.lang.String	INTERVENTION_REPORT
public static final java.lang.String	SERVER_REPORT
public static final java.lang.String	FOLDER_COMMENT

public static final java.lang.String	FOLDER_COMMENT_ARCH
public static final java.lang.String	DOCF_COMMENT
public static final java.lang.String	DOCF_COMMENT_ARCH
public static final java.lang.String	PROCESS_COMMENT
public static final java.lang.String	DOCP_COMMENT
public static final java.lang.String	VALIDCYCLE_TEMPLATE
public static final java.lang.String	VALIDCYCLE
public static final java.lang.String	VALIDCYCLE_STAGE_TEMPLATE
public static final java.lang.String	VALIDCYCLE_STAGE
public static final java.lang.String	VALIDCYCLE_STAGE_DATA_TEMPLATE
public static final java.lang.String	VALIDCYCLE_STAGE_DATA
public static final java.lang.String	VALIDCYCLE_STAGE_TEMPLATE_NOTIFICATION
public static final java.lang.String	VALIDCYCLE_STAGE_NOTIFICATION
public static final java.lang.String	EXPIRABLE_DOWNLOAD_LINK
public static final java.lang.String	NOTIFICATION_DEFINITION
public static final java.lang.String	NOTIFICATION
public static final java.lang.String	FAVORITE
public static final java.lang.String	OBJECT_LINK
public static final	PACKAGE

java.lang.String	
public static final java.lang.String	JUKE
public static final java.lang.String	POSITION
public static final java.lang.String	PROCESS_TESTING_DEFINITION
public static final java.lang.String	PROCESS_TESTING_INSTANCE
public static final java.lang.String	PROCESS_TESTING_MESSAGE
public static final java.lang.String	PROCESS_TESTING_BATCH
public static final java.lang.String	PROCESS_TESTING_DEFINITION_BATCH

Field detail :

```
public static final String FILE = "FILE"
```

File id prefix.

Id structure is

FILEDDDDXXXXXXXXAAAAAAAABBBBBBBBCCCCCCCC

where

- DDDD is the hexadecimal representation of the database numeric id,
- XXXXXXXX is the hexadecimal representation of the storage area numeric id,
- AAAAAAAA the most significant id-generator counter hexadecimal value,
- BBBBBBBB the mid significant id-generator counter hexadecimal value,
- CCCCCCCC the least significant id-generator counter hexadecimal value.

```
public static final String USER = "USER"
```

User object prefix.

Id structure is

USER0000XXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the user numeric id.

```
public static final String USER_DELEGATION = "DLGT"
```

User delegation object prefix.

Id structure is

`DLGT0000XXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the user delegation numeric id.

```
public static final String ROLE_DELEGATION = "DGRO"
```

Role delegation object prefix.

Id structure is

`DGRO0000XXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the role delegation numeric id.

```
public static final String USER_PROFILE = "USPR"
```

User profile object prefix.

Id structure is

`USPR0000XXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the user profile numeric id.

```
public static final String GROUP = "GROU"
```

Group object prefix.

Id structure is

`GROU0000XXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the group numeric id.

```
public static final String ROLE = "ROLE"
```

Role object prefix.

Id structure is

`USER0000XXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the role numeric id.

```
public static final String MANAGER = "MGRR"
```

Manager object prefix.

Id structure is

`MGRR0000XXXXXXXX00000000000000000000000000`

where XXXXXXXX is the hexadecimal representation of the manager role numeric id.

```
public static final String GROUP_USER = "ASGU"
```

User-to-group assignment object prefix.

Id structure is

ASGU0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the group numeric id and YYYYYYYY is the hexadecimal representation of the user numeric id.

```
public static final String USER_ROLE = "ASUR"
```

Role-to-user assignment object prefix.

Id structure is

ASUR0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the user numeric id and YYYYYYYY is the hexadecimal representation of the role numeric id.

```
public static final String GROUP_ROLE = "ASGR"
```

Role-to-group assignment object prefix.

Id structure is

ASRU0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the group numeric id and YYYYYYYY is the hexadecimal representation of the role numeric id.

```
public static final String MANAGER_USER = "ASMU"
```

User-to-manager role assignment object prefix.

Id structure is

ASMU0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the manager role numeric id and YYYYYYYY is the hexadecimal representation of the user numeric id.

```
public static final String FORM_DISPLAY_PROFILE = "ASFP"
```

Form-to-display profile assignment object prefix.

Id structure is:

ASUDDDDDXXXXXXXXYYYYYYYY0000000000000000

,

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the form id, YYYYYYYY is the hexadecimal representation of the display profile id

```
public static final String DATABASE = "DATA"
```

Database object prefix.

Id structure is

DATADDDD00000000000000000000000000000000

where DDDD is the hexadecimal representation of the database numeric id.

```
public static final String GROUP_DATABASE = "ASGD"
```

Group-to-database assignment object prefix.

Id structure is

ASGDDDDDXXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the group numeric id and DDDD is the hexadecimal representation of the database numeric id.

```
public static final String USER_DATABASE = "ASUD"
```

User-to-database assignment object prefix.

Id structure is

ASUDDDDDXXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the user numeric id and DDDD is the hexadecimal representation of the database numeric id.

```
public static final String ROLE_DATABASE = "ASRD"
```

Role-to-database assignment object prefix.

Id structure is

ASRDDDDDXXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the role numeric id and DDDD is the hexadecimal representation of the database numeric id.

```
public static final String STORAGE_AREA = "AREA"
```

Storage area object prefix.

Id structure is

AREA0000XXXXXXXX000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the storage area numeric id.

```
public static final String DATABASE_STORAGE_RULE = "RULE"
```

Storage rule object prefix.

Id structure is

RULE0000XXXXXXXX000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the storage rule numeric id.

```
public static final String GROUP_PORTAL = "ASGP"
```

Group-to-portal assignment object prefix.

Id structure is

ASGP0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the group numeric id and YYYYYYYY is the hexadecimal representation of the portal numeric id.

```
public static final String USER_PORTAL = "ASUP"
```

User-to-portal assignment object prefix.

Id structure is

ASUP0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the user numeric id and YYYYYYYY is the hexadecimal representation of the portal numeric id.

```
public static final String ROLE_PORTAL = "ASRP"
```

Role-to-portal assignment object prefix.

Id structure is

ASRP0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the role numeric id and YYYYYYYY is the hexadecimal representation of the portal numeric id.

```
public static final String PLANNER = "PLAN"
```

Planner object prefix.

Id structure is

PLAN0000XXXXXXXX000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the planner numeric id.

```
public static final String PLANNER_DAY = "PLND"
```

Planner day object prefix.

Id structure is

PLND0000XXXXXXXXYYYYYYYY0000000000000000

where XXXXXXXX is the hexadecimal representation of the planner numeric id and YYYYYYYY is the 00YYYYMMDD representation of the day.

```
public static final String RESOURCE = "RSRC"
```

Resource object prefix.

Id structure is

RSRC0000XXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the resource numeric id.

```
public static final String SCRIPT = "SCPT"
```

Script object prefix.

Id structure is

SCPT0000XXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the script numeric id.

```
public static final String SECURITY_PROFILE = "SECP"
```

Security profile object prefix.

Id structure is

SECPDDDDXXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the security profile numeric id and DDDD is the database numeric id.

```
public static final String MAIL_TEMPLATE = "MTPL"
```

Mail template object prefix.

Id structure is

MTPLDDDDXXXXXXXX00000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the mail template numeric id and DDDD is the database numeric id.

```
public static final String DOCUMENT_TEMPLATE = "DTPL"
```

Document template object prefix.

Id structure is

`DTPLDDDDXXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the document template numeric id and `DDDD` is the database numeric id.

```
public static final String FOLDER_PROFILE = "FLPR"
```

Folder profile object prefix.

Id structure is

`FLPRDDDDXXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the folder profile numeric id and `DDDD` is the database numeric id.

```
public static final String COMPOSITION_RULE = "CPRU"
```

Composition rule object prefix.

Id structure is

`CPRUDDDDXXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the composition rule numeric id and `DDDD` is the database numeric id.

```
public static final String SEARCH_TEMPLATE = "SRCH"
```

Search template object prefix.

Id structure is

`SRCHDDDDXXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the search template numeric id and `DDDD` is the database numeric id.

```
public static final String USER_SEARCHTEMPLATE = "ASUT"
```

Searchtemplate-to-user assignment object prefix.

Id structure is

`ASUT0000XXXXXXXXYYYYYYYY00000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the user numeric id and `YYYYYYYY` is the hexadecimal representation of the Searchtemplate numeric id.

```
public static final String GROUP_SEARCHTEMPLATE = "ASGT"
```

Searchtemplate-to-group assignment object prefix.

Id structure is

`ASGT0000XXXXXXXXYYYYYYYY0000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the group numeric id and `YYYYYYYY` is the hexadecimal representation of the Searchtemplate numeric id.

```
public static final String ROLE_SEARCHTEMPLATE = "ASRT"
```

Searchtemplate-to-role assignment object prefix.

Id structure is

`ASRT0000XXXXXXXXYYYYYYYY0000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the role numeric id and `YYYYYYYY` is the hexadecimal representation of the Searchtemplate numeric id.

```
public static final String ASYNC_JOB = "JASY"
```

Asynchronous job object prefix.

Id structure is

`JASY0000XXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the asynchronous job numeric id.

```
public static final String EXTENDED_PROPERTY = "EXPR"
```

Extended property object prefix.

Id structure is

`EXPR0000XXXXXXXX00000000000000000000000000`

where `XXXXXXXX` is the hexadecimal representation of the extended property id.

```
public static final String OBJECT_EXTENDED_PROPERTY = "ASOP"
```

Object to extended property assignment prefix.

Id structure is

`ASOP0000XXXXXXXXYYYYYYYY0000000000000000`

where `XXXXXXXX` is the hexadecimal representation of extended property id and `YYYYYYYY` is the hexadecimal representation of the object numeric id.

```
public static final String CERTIFICATE = "CERT"
```

'Certificate' object prefix.

Id structure is

CERT0000 XXXXXXXX 00000000 00000000 00000000

where XXXXXXXX is the hexadecimal representation of certificate id

```
public static final String TYPE = "DFTY"
```

Type (list) object prefix.

Id structure is

DFTYDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the type numeric id.

```
public static final String GRID = "GRID"
```

Grid object prefix.

Id structure is

GRIDDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the type numeric id.

```
public static final String ATTRIBUTE = "DFAT"
```

Attribute object prefix.

Id structure is

DFATDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the attribute numeric id.

```
public static final String FORM = "DFFR"
```

Form object prefix.

Id structure is

DFFRDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the form numeric id.

```
public static final String FORM_ATTRIBUTE = "ASFA"
```

Attribute-to-form assignment object prefix.

Id structure is

ASGVDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the form numeric id and YYYYYYYY is the hexadecimal representation of the attribute numeric id.

```
public static final String VIEW = "DFVW"
```

View object prefix.

Id structure is

DFVWDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the view numeric id.

```
public static final String VIEW_ELEMENT = "DFVE"
```

View element object prefix.

Id structure is

DFVEDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the view numeric id and YYYYYYYY is the hexadecimal representation of the view element numeric id

```
public static final String GROUP_VIEW = "ASGV"
```

Group-to-view assignment object prefix.

Id structure is

ASGVDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the group numeric id and YYYYYYYY is the hexadecimal representation of the view numeric id.

```
public static final String USER_VIEW = "ASUV"
```

User-to-view assignment object prefix.

Id structure is

ASUVDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the

hexadecimal representation of the user numeric id and YYYYYYYY is the hexadecimal representation of the view numeric id.

```
public static final String ROLE_VIEW = "ASRV"
```

Role-to-view assignment object prefix.

Id structure is

ASRVDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the role numeric id and YYYYYYYY is the hexadecimal representation of the view numeric id.

```
public static final String PROCESS_DEF = "DFPR"
```

Process definition object prefix.

Id structure is

DFPRDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the process definition numeric id.

```
public static final String SCRIPT2 = "SCPD"
```

Design script object prefix.

Id structure is

SCPDDDDDXXXXXXXX00000000000000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the script numeric id.

```
public static final String SCRIPT2_VERSION = "VDSC"
```

version of Design script object prefix.

Id structure is

VDSCDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the script numeric id. YYYYYYYY is the hexadecimal representation of the script version numeric id.

```
public static final String COMPOSITION_RULE_VERSION = "VDCR"
```

version of Design composition rule object prefix.

Id structure is

VDCRDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the composition rule numeric id. YYYYYYYY is the hexadecimal representation of the composition rule version numeric id.

```
public static final String FORM_VERSION = "VDFR"
```

version of Design form object prefix.

Id structure is

VDFRDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the form numeric id. YYYYYYYY is the hexadecimal representation of the form version numeric id.

```
public static final String PROCESS_DEF_VERSION = "VDPR"
```

version of Design process definition object prefix.

Id structure is

VDPRDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the process definition numeric id. YYYYYYYY is the hexadecimal representation of the process definition version numeric id.

```
public static final String IS_JOB_VERSION = "VDIS"
```

version of Design isjob object prefix.

Id structure is

VDISDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the isjob numeric id. YYYYYYYY is the hexadecimal representation of the isjob version numeric id.

```
public static final String MODEL_OCR_VERSION = "VDMO"
```

version of Design modelocr object prefix.

Id structure is

VDMODDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the

hexadecimal representation of the modelocr numeric id. YYYYYYYY is the hexadecimal representation of the modelocr version numeric id.

```
public static final String VIEW_VERSION = "VDVW"
```

Version of View object prefix.

Id structure is

VDVWDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the database numeric id , XXXXXXXX is the hexadecimal representation of the view numeric id. YYYYYYYY is the hexadecimal representation of the view version numeric id.

```
public static final String VIEW_ELEMENT_VERSION = "VDVE"
```

Version of View element object prefix.

Id structure is

VDVEDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the view version numeric id, YYYYYYYY is the hexadecimal representation of the view version element numeric id

```
public static final String IS_JOB = "ISJB"
```

IntelliScan job object prefix.

Id structure is

ISJBDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the job numeric id.

```
public static final String MODEL_OCR = "MOCR"
```

ModelOCR object prefix.

Id structure is

MOCRDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id and XXXXXXXX is the hexadecimal representation of the model ocr numeric id.

```
public static final String GROUP_ISJOB = "ASGI"
```

Group-to-IsJob assignment object prefix.

Id structure is

ASGIDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the group numeric id and YYYYYYYY is the hexadecimal representation of the IsJob numeric id.

```
public static final String ROLE_ISJOB = "ASRI"
```

Role-to-view assignment object prefix.

Id structure is

ASRIDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the role numeric id and YYYYYYYY is the hexadecimal representation of the IsJob numeric id.

```
public static final String USER_ISJOB = "ASUI"
```

User-to-IsJob assignment object prefix.

Id structure is

ASUIDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the hexadecimal representation of the database numeric id, XXXXXXXX is the hexadecimal representation of the user numeric id and YYYYYYYY is the hexadecimal representation of the IsJob numeric id.

```
public static final String CLASS_ELEMENT = "CELM"
```

Classification element object prefix.

Id structure is

CELMDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the view numeric id, YYYYYYYY is the hexadecimal representation of the view element numeric id, ZZZZZZZZ is the hexadecimal representation of the class element numeric id.

```
public static final String PROCESS = "PROC"
```

Process object prefix.

Id structure is

PROCDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id.

```
public static final String TASK = "TASK"
```

Task object prefix.

Id structure is

TASKDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id. YYYYYYYY is the hexadecimal representation of the task numeric id.

```
public static final String FOLDER = "FLDR"
```

Folder object prefix.

Id structure is

FLDRDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id. Numeric '0' means database root folder.

```
public static final String FOLDER_VALIDATIONCYCLE = "FLVC"
```

Validation Cycle (Virtual) Folder prefix.

Id structure is

FLVCDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

where

DDDD is the database numeric id,

XXXXXXXX is the hexadecimal representation of the folder .

YYYYYYYY is the hexadecimal representation of the document stage,

ZZZZZZZZ is the hexadecimal representation of the document stage status (normal , delayed, expired - see Constants),

WWWWWWWW is the hexadecimal representation of the actor (i as executor, i as manager ,i as owner of validation cycle - see Constants).

```
public static final String FOLDER_MYDOCS = "FLMY"
```

My documents (Virtual) Folder prefix.

Id structure is

FLMYDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id. Numeric '0' means 'my documents' folder.

```
public static final String FOLDER_WORKSPACE = "FLWS"
```

Workspace (the Root) Folder prefix.

Id structure is

`FLWSDDDDXXXXXXXX00000000000000000000000000`

where `DDDD` is the database numeric id, `XXXXXXXX` is the hexadecimal representation of the folder numeric id. Numeric '0' means 'workspace' folder.

```
public static final String FOLDER_IN_WORKSPACE = "FLWR"
```

Workspace (not Root) Folder prefix.

Id structure is

`FLWRDDDDXXXXXXXX00000000000000000000000000`

where `DDDD` is the database numeric id, `XXXXXXXX` is the hexadecimal representation of the folder numeric id. Numeric '0' means 'workspace' folder.

```
public static final String PIN_DOCF = "PINF"
```

Pin contained in folder object prefix.

Id structure is

`PINFDDDDXXXXXXXXYYYYYYYY0000000000000000`

where `DDDD` is the database numeric id, `XXXXXXXX` is the hexadecimal representation of the folder numeric id, `YYYYYYYY` is the hexadecimal representation of the pin numeric id.

```
public static final String PIN_DOCP = "PINP"
```

Pin contained in process object prefix.

Id structure is

`PINPDDDDXXXXXXXXYYYYYYYY0000000000000000`

where `DDDD` is the database numeric id, `XXXXXXXX` is the hexadecimal representation of the process numeric id, `YYYYYYYY` is the hexadecimal representation of the pin numeric id.

```
public static final String PIN_DOCW = "PINW"
```

Pin contained in a workspace

Id structure is

`PINWDDDDXXXXXXXXYYYYYYYY0000000000000000`

where `DDDD` is the database numeric id, `XXXXXXXX` is the hexadecimal representation of the process numeric id, `YYYYYYYY` is the hexadecimal representation of the pin numeric id.

```
public static final String FILE_DOCF = "DOCF"
```

File document contained in folder object prefix.

Id structure is

DOCFDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FILE_DOCP = "DOCP"
```

File document contained in process object prefix.

Id structure is

DOCPDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FICHE_DOCF = "FCHF"
```

File-less document (index fiche) contained in folder object prefix.

Id structure is

FCHFDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FICHE_DOCD = "FCHD"
```

File-less document (index fiche) contained in distribution list (modified copy of original) /.

Id structure is

FCHDDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FICHE_DOCV = "FCHV"
```

File-less document (index fiche) contained in Validation Cycle /.

Id structure is

FCHVDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FICHE_DOCW = "FCHW"
```

File-less document (index fiche) document contained in a workspace /.

Id structure is

FCHWDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FILE_DOCD = "DOCD"
```

File document contained in distribution list /.

Id structure is

DOCDDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FILE_DOCV = "DOCV"
```

File document contained in validation cycle /.

Id structure is

DOCVDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FILE_DOCW = "DOCW"
```

File document contained in a workspace /.

Id structure is

DOCWDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means

document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String FICHE_DOCP = "FCHP"
```

File document contained in workspace /.

Id structure is

DOCVDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String REF_DOCF = "RFDF"
```

Folder contained reference to a file document object prefix.

Id structure is

RFDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the reference document numeric id.

```
public static final String REF_DOCP = "RFDP"
```

Process contained reference to a file document object prefix.

Id structure is

RFDPDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the reference document numeric id.

```
public static final String REF_FICHEF = "RFFF"
```

Folder contained reference to a file-less (index fiche) document object prefix.

Id structure is

RFFFDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the reference document numeric id.

```
public static final String REF_FICHEP = "RFFP"
```

Process contained reference to a file-less (index fiche) document object prefix.

Id structure is

RFFPDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the reference document numeric id.

```
public static final String FOLDER_NOTE = "NTFL"
```

Folder note object prefix.

Id structure is

NTFLDDDDXXXXXXXX0000000000000000YYYYYYYY

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the note numeric id.

```
public static final String PROCESS_NOTE = "NTPR"
```

Process note object prefix.

Id structure is

NTPRDDDDXXXXXXXX0000000000000000YYYYYYYY

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the note numeric id.

```
public static final String DOC_NOTE_BASE = "NTD"
```

Folder contained document note object prefix.

Id structure is

NTDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id, WWWWWWWW is the hexadecimal representation of the note numeric id.

```
public static final String FICHEF_NOTE = "NTFF"
```

Folder contained card (fiche) note object prefix.

Id structure is

NTDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means card (fiche) is not in pin), ZZZZZZZZ is the hexadecimal representation of the card numeric id, WWWWWWWW is the hexadecimal representation of the note numeric id.

```
public static final String DOCP_NOTE = "NTDP"
```

Process contained document note object prefix.

Id structure is

NTDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id, WWWWWWWW is the hexadecimal representation of the note numeric id.

```
public static final String FICHEP_NOTE = "NTFP"
```

Process contained document note object prefix.

Id structure is

NTDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means card (fiche) is not in pin), ZZZZZZZZ is the hexadecimal representation of the card (fiche) numeric id, WWWWWWWW is the hexadecimal representation of the note numeric id.

```
public static final String DOCF_VERSION = "VNDF"
```

Folder contained document version object prefix.

Id structure is

VNDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id, WWWWWWWW is the hexadecimal representation of the version numeric id.

```
public static final String DOCP_VERSION = "VNDP"
```

Process contained document version object prefix.

Id structure is

VNDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZWWWWWWWW

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id, WWWWWWWW is the hexadecimal representation of the version numeric id.

```
public static final String TASK_DEF = "TKDF"
```

Task definition object prefix.

Id structure is

TKDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process definition numeric id, YYYYYYYY is the hexadecimal representation of the process definition version numeric id (0 means working copy/current deployment), and ZZZZZZZZ is the hexadecimal representation of the task definition numeric id.

```
public static final String START_DEF = "STDF"
```

Start point definition object prefix.

Id structure is

STDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process definition numeric id, YYYYYYYY is the hexadecimal representation of the process definition version numeric id (0 means working copy/current deployment), and ZZZZZZZZ is the hexadecimal representation of the start point definition numeric id.

```
public static final String TASK_GROUP_DEF = "TGDF"
```

Task group definition object prefix.

Id structure is

TGDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process definition numeric id, YYYYYYYY is the hexadecimal representation of the process definition version numeric id (0 means working copy/current deployment), and ZZZZZZZZ is the hexadecimal representation of the task group definition numeric id.

```
public static final String DELEGATION = "DLGN"
```

User object prefix.

Id structure is

USER0000XXXXXXXX000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the user numeric id.

```
public static final String ARCHIVE_FOLDER = "ARFL"
```

Archive folder object prefix.

Id structure is

ARFLDDDDXXXXXXXX000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id. Numeric '0' means database root folder.

```
public static final String ARCHIVE_PIN = "APIN"
```

Pin contained in folder object prefix.

Id structure is

APINDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id.

```
public static final String ARCHIVE_FILE = "ADOC"
```

File document contained in folder object prefix.

Id structure is

ADOCDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id.

```
public static final String INTERVENTION_REPORT = "INRE"
```

Intervention report.

Id structure is

INRE0000XXXXXXXX000000000000000000000000

where XXXXXXXX is the hexadecimal representation of the intervention report numeric id.

```
public static final String SERVER_REPORT = "SVRE"
```

Server report.

Id structure is

SVRE0000XXXXXXXX00000000000000000000000

where XXXXXXXX is the hexadecimal representation of the server report numeric id.

```
public static final String FOLDER_COMMENT = "CMTF"
```

A folder comment.

CMTFDDDDXXXXXXXX0000000000000000CCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, CCCCCCCC is the hexadecimal representation of the comment numeric id.

```
public static final String FOLDER_COMMENT_ARCH = "ACMF"
```

An archived folder's comment.

ACMFDDDDXXXXXXXX0000000000000000CCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, CCCCCCCC is the hexadecimal representation of the comment numeric id.

```
public static final String DOCF_COMMENT = "CMDF"
```

A document-in-a-folder's comment.

CMDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZCCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id, CCCCCCCC is the hexadecimal representation of the comment numeric id.

```
public static final String DOCF_COMMENT_ARCH = "ACDF"
```

An archived document-in-a-folder's comment.

ACDFDDDDXXXXXXXXYYYYYYYYZZZZZZZZCCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the folder numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id, CCCCCCCC is the hexadecimal representation of the comment numeric id.

```
public static final String PROCESS_COMMENT = "CMTP"
```

A processes comment.

CMTPDDDDXXXXXXXX0000000000000000CCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, CCCCCCCC is the hexadecimal representation of the comment numeric id.

```
public static final String DOCP_COMMENT = "CMDP"
```

A document-in-a-process's comment.

CMDPDDDDXXXXXXXXYYYYYYYYZZZZZZZZCCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the process numeric id, YYYYYYYY is the hexadecimal representation of the pin numeric id, (numeric 0 means document is not in pin), ZZZZZZZZ is the hexadecimal representation of the document numeric id, CCCCCCCC is the hexadecimal representation of the comment numeric id.

```
public static final String VALIDCYCLE_TEMPLATE = "VCCT"
```

validation cycle template .

Id structure is

VCCTDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation cycle template id.

```
public static final String VALIDCYCLE = "VCCR"
```

validation cycle (runtime).

Id structure is

VCCRDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation cycle id.

```
public static final String VALIDCYCLE_STAGE_TEMPLATE = "VCST"
```

validation cycle template stage .

Id structure is

VCSTDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation cycle template id, YYYYYYYY is the hexadecimal representation of the validation cycle template stage id.

```
public static final String VALIDCYCLE_STAGE = "VCSR"
```

validation cycle stage .

Id structure is

VCSRDDDDXXXXXXXXYYYYYYYY0000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation cycle id, YYYYYYYY is the hexadecimal representation of the validation cycle stage id

```
public static final String VALIDCYCLE_STAGE_DATA_TEMPLATE = "VCDT"
```

validation cycle template stage data.

Id structure is

VCDTDDDDXXXXXXXXYYYYYYYYZZZZZZZZCCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation cycle template id, YYYYYYYY is the hexadecimal representation of the validation cycle template stage id, ZZZZZZZZ is the hexadecimal representation of the actor id, CCCCCCCC is the hexadecimal representation of the actor type.

```
public static final String VALIDCYCLE_STAGE_DATA = "VCDR"
```

validation cycle stage data.

Id structure is

VCDRDDDDXXXXXXXXYYYYYYYYZZZZZZZZCCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation cycle template id, YYYYYYYY is the hexadecimal representation of the validation cycle template stage id, ZZZZZZZZ is the hexadecimal representation of the actor id, CCCCCCCC is the hexadecimal representation of the actor type.

```
public static final String
VALIDCYCLE_STAGE_TEMPLATE_NOTIFICATION = "VCNT"
```

validation cycle template stage notification.

Id structure is

VCNTDDDDXXXXXXXXYYYYYYYYZZZZZZZZCCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation cycle template id, YYYYYYYY is the hexadecimal representation of the validation cycle template stage id, ZZZZZZZZ is the hexadecimal representation of the event, CCCCCCCC is the hexadecimal representation of the destination type .

```
public static final String VALIDCYCLE_STAGE_NOTIFICATION = "VCNR"
```

validation cycle stage notification.

Id structure is

VCNRDDDDXXXXXXXXYYYYYYYYZZZZZZZZCCCCCCCC

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the validation

cycle id, YYYYYYYY is the hexadecimal representation of the validation cycle stage id, ZZZZZZZZ is the hexadecimal representation of the event, CCCCCCCC is the hexadecimal representation of the destination type .

```
public static final String EXPIRABLE_DOWNLOAD_LINK = "EXDL"
```

Expirable download link.

Id structure is

EXDLDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the expirable download link id.

```
public static final String NOTIFICATION_DEFINITION = "NODF"
```

Notification definition

Id structure is

NODFDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the notification definition id.

```
public static final String NOTIFICATION = "NOTF"
```

Notification event

Id structure is

NOTFDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the notification event id.

```
public static final String FAVORITE = "FAVO"
```

Favorite

Id structure is

FAVODDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the notification id.

```
public static final String OBJECT_LINK = "OLNK"
```

Object link

Id structure is

OLNKDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the link id.

```
public static final String PACKAGE = "PCKG"
```

Package link

Id structure is

PCKGDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the package id.

```
public static final String JUKE = "JUKE"
```

Juke link

Id structure is

JUKEDDDDXXXXXXXX00000000000000000000000000

where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the jukeid id.

```
public static final String POSITION = "POSN"
```

Position

Id structure is

POSN0000PPPPPPPPRRRRRRRRGGGGGGGGUUUUUUUU

where PPPPPPPP is a unique position-sequence generated integer,

RRRRRRRR is the role numeric id, GGGGGGGG is the group numeric id, UUUUUUUU is the user numeric id.

```
public static final String PROCESS_TESTING_DEFINITION = "PTDF"
```

Process testing definition.

Id structure is

PTDFDDDDXXXXXXXXYYYYYYYY0000000000000000

where where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the testing definition numeric id, YYYYYYYY is the hexadecimal representation of the process definition numeric id,

```
public static final String PROCESS_TESTING_INSTANCE = "PTIN"
```

Process testing instance.

Id structure is

PTINDDDDXXXXXXXXYYYYYYYYZZZZZZZZTTTTTTTT

where where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the testing instance numeric id, YYYYYYYY is the hexadecimal representation of the testing definition numeric id, ZZZZZZZZ is the hexadecimal representation of the process definition numeric id, TTTTTTTT is the hexadecimal representation of the process numeric id

```
public static final String PROCESS_TESTING_MESSAGE = "PTMS"
```

Process testing message.

Id structure is

PTMSDDDDXXXXXXXXYYYYYYYYZZZZZZZZ00000000

where where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the testing message numeric id, YYYYYYYY is the hexadecimal representation of the testing instance numeric id, ZZZZZZZZ is the hexadecimal representation of the process batch id

```
public static final String PROCESS_TESTING_BATCH = "PTBT"
```

Process testing batch.

Id structure is

PTBTDDDDXXXXXXXX000000000000000000000000

where where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the batch message numeric id

```
public static final String PROCESS_TESTING_DEFINITION_BATCH = "ASTB"
```

Role-to-user assignment object prefix.

Id structure is

ASTBDDDDXXXXXXXXYYYYYYYYZZZZZZZZTTTTTTTT

, where DDDD is the database numeric id, XXXXXXXX is the hexadecimal representation of the assignment numeric id, YYYYYYYY is the hexadecimal representation of process testing definition id, ZZZZZZZZ is the hexadecimal representation of the process testing batch id, TTTTTTTT is the hexadecimal representation of the process testing instance id

SIATEL_0

```
public abstract class Siatel_0
```

Inheritance root abstrract class for all Identified implementations.

All Superinterfaces :

Serializable , Cloneable , Replicable , Identified

See also :

Identified , Prefix , IdFormat

Constructor summary :

Constructor
`Siatel_0 ()`
`Siatel_0 (com.siatel.domain.Siatel_0 source)`
`Siatel_0 (java.lang.String id)`

Constructor detail :

```
public Siatel_0 ()
```

Default constructor.

```
public Siatel_0 (Siatel_0 source)
```

Copy constructor.

Parameters :

`source` - the copy source

```
public Siatel_0 (String id)
```

Id initializer constructor.

Parameters :

`id` - the object id

SIATEL_1

```
public abstract class Siatel_1
```

All Superinterfaces :

Labeled , Owned , Flagged , Dated

Author :

stefanu

Constructor summary :

Constructor
Siatel_1 ()
Siatel_1 (java.lang.String id)
Siatel_1 (java.lang.String id, java.lang.String label, java.lang.String ownerId, long flags, java.util.Date dateC, java.util.Date dateM)

Constructor detail :

```
public Siatel_1 ()
```

```
public Siatel_1 (String id)
```

```
public Siatel_1 (String id, String label, String ownerId, long flags,
Date dateC, Date dateM)
```

SIATEL_1_1

```
public abstract class Siatel_1_1
```

All Superinterfaces :

Described

Author :

stefanu

Constructor summary :

Constructor
Siatel_1_1 ()
Siatel_1_1 (java.lang.String id)
Siatel_1_1 (java.lang.String id, java.lang.String label, java.lang.String ownerId, long flags, java.util.Date dateC, java.util.Date dateM, java.lang.String description)

Constructor detail :

```
public Siatel_1_1 ()
```

```
public Siatel_1_1 (String id)
```

```
public Siatel_1_1 (String id, String label, String ownerId, long flags,
Date dateC, Date dateM, String description)
```

SIATEL_1_2

```
public abstract class Siatel_1_2
```

All Superinterfaces :

Deployable

Author :

stefanu

Constructor summary :

Constructor
Siatel_1_2 ()
Siatel_1_2 (java.lang.String id)
Siatel_1_2 (java.lang.String id, java.lang.String label, java.lang.String ownerId, long flags, java.util.Date dateC, java.util.Date dateM, java.lang.String description, boolean deployed)

Constructor detail :

```
public Siatel_1_2 ()
```

```
public Siatel_1_2 (String id)
```

```
public Siatel_1_2 (String id, String label, String ownerId, long flags,
Date dateC, Date dateM, String description, boolean deployed)
```

SIATEL_1_3

```
public abstract class Siatel_1_3
```

All Superinterfaces :

Realized

Author :

stefanu

Constructor summary :

Constructor
Siatel_1_3 ()
Siatel_1_3 (java.lang.String id)
Siatel_1_3 (java.lang.String id, java.lang.String label, java.lang.String ownerId, long flags, java.util.Date dateC, java.util.Date dateM, java.lang.String description, boolean deployed, java.lang.String fileId)

Constructor detail :

```
public Siatel_1_3 ()
```

```
public Siatel_1_3 (String id)
```

```
public Siatel_1_3 (String id, String label, String ownerId, long flags,
Date dateC, Date dateM, String description, boolean deployed, String
fileId)
```

SIATEL_1_4

```
public abstract class Siatel_1_4
```

All Superinterfaces :

DeployableEx

Since :

30/11/2007

Author :

Victor Citiriga

Constructor summary :

Constructor
Siatel_1_4 ()
Siatel_1_4 (java.lang.String id)
Siatel_1_4 (java.lang.String id, java.lang.String label, java.lang.String ownerId, long flags, java.util.Date dateC, java.util.Date dateM, java.lang.String description, boolean deployed, java.lang.String fileId, java.lang.String deploymentId, long deploymentTimestamp)

Constructor detail :

```
public Siatel_1_4 ()
```

```
public Siatel_1_4 (String id)
```

```
public Siatel_1_4 (String id, String label, String ownerId, long flags, Date dateC, Date dateM, String description, boolean deployed, String fileId, String deploymentId, long deploymentTimestamp)
```

SIATEL_2

```
public abstract class Siatel_2
```

New in 7.8: added support for UUIDEnabled. The uuid is not (de)serialized in this class, but by descending classes, since it was later added and descendants' token indexes have evolved differently

All Superinterfaces :

DatedEx , Viewable , Timed , UUIDEnabled

Author :

victorc

null :

7.8

Constructor summary :

Constructor
Siatel_2 ()
Siatel_2 (java.lang.String id)
Siatel_2 (java.lang.String id, java.lang.String label, java.lang.String ownerId, long flags, java.util.Date dateC, java.util.Date dateM, java.util.Date dateA, java.lang.String formId, java.util.List attributes, int icon, java.util.Date dateS, java.util.Date dateE)

Constructor detail :

```
public Siatel_2 ()
```

```
public Siatel_2 (String id)
```

```
public Siatel_2 (String id, String label, String ownerId, long flags, Date dateC, Date dateM, Date dateA, String formId, List attributes, int icon, Date dateS, Date dateE)
```

SIATEL_3

```
public abstract class Siatel_3
```

All Superinterfaces :

Lifecycled , Securable , Deletable

Author :

stefanu

Constructor summary :

Constructor
Siatel_3 ()
Siatel_3 (java.lang.String id)
Siatel_3 (java.lang.String id, java.lang.String label, java.lang.String ownerId, long flags, java.util.Date dateC, java.util.Date dateM, java.util.Date dateA, java.lang.String formId, java.util.List attributes, int icon, java.lang.String stateId, java.util.Date dateS, java.util.Date dateE, boolean deleted, java.util.Date dateDel, int deletedBy)

Constructor detail :

```
public Siatel_3 ()
```

```
public Siatel_3 (String id)
```

```
public Siatel_3 (String id, String label, String ownerId, long flags, Date dateC, Date dateM, Date dateA, String formId, List attributes, int icon, String stateId, Date dateS, Date dateE, boolean deleted, Date dateDel, int deletedBy)
```

TAG

```
public class Tag
```

Object tags.

Bits 0-63 match to different object tags

Author :

stefanu

Field summary :

Modifier and type	Field
public static final long	TAG_TEXT_1
public static final long	TAG_TEXT_2
public static final long	TAG_TEXT_3
public static final long	TAG_TEXT_4
public static final long	TAG_TEXT_5
public static final long	TAG_BACKGROUND_1
public static final long	TAG_BACKGROUND_2
public static final long	TAG_BACKGROUND_3
public static final long	TAG_BACKGROUND_4
public static final long	TAG_BACKGROUND_5

Field detail :

```
public static final long TAG_TEXT_1 = 1L
```

Bit 1. Tag #1.

```
public static final long TAG_TEXT_2 = 2L
```

Bit 2. Tag #2.

```
public static final long TAG_TEXT_3 = 4L
```

Bit 3. Tag #3.

```
public static final long TAG_TEXT_4 = 8L
```

Bit 4. Tag #1.

```
public static final long TAG_TEXT_5 = 16L
```

Bit 5. Tag #2.

```
public static final long TAG_BACKGROUND_1 = 4294967296L
```

Bit 32. Tag #1.

```
public static final long TAG_BACKGROUND_2 = 8589934592L
```

Bit 33. Tag #2.

```
public static final long TAG_BACKGROUND_3 = 17179869184L
```

Bit 34. Tag #1.

```
public static final long TAG_BACKGROUND_4 = 34359738368L
```

Bit 35. Tag #2.

```
public static final long TAG_BACKGROUND_5 = 68719476736L
```

Bit 36. Tag #2.

UNIQUECRITERION

```
public class UniqueCriterion
```

This class defines a unqiue criterion used by view elements design and runtime.

This class consists of:

1. the **id of the attribute** on which this criterion is evaluated.
2. a **modifier** to apply the attribute values when evaluating.
3. the **value** to which this criterion evaluates at view runtime. This member is ignored when managing view elements (view design).

All Superinterfaces :

Cloneable

Since :

12/07/2006

Author :

vic

See also :

ViewElement , View

Constructor summary :

Constructor
`UniqueCriterion ()`
`UniqueCriterion (java.lang.String attrId, int attrModifier)`
`UniqueCriterion (java.lang.String attrId, int attrModifier, java.lang.Object value)`
`UniqueCriterion (java.lang.String attrId, int attrModifier, java.lang.Object value, boolean ascending)`

Constructor detail :

```
public UniqueCriterion ()
```

No arguments constructor.

```
public UniqueCriterion (String attrId, int attrModifier)
```

Constructor with design time initialization parameters.

Parameters :

`attrId` - the attribute id

`attrModifier` - the attribute modifier

```
public UniqueCriterion (String attrId, int attrModifier, Object value)
```

Constructor with runtime initialization parameters.

Parameters :

`attrId` - the attribute id

`attrModifier` - the attribute modifier

`value` - the criterion evaluation

```
public UniqueCriterion (String attrId, int attrModifier, Object value,
boolean ascending)
```

Constructor with runtime initialization parameters.

Parameters :

`attrId` - the attribute id

`attrModifier` - the attribute modifier

`value` - the criterion evaluation

`ascending` - the sort order

VALUETOKEN

```
public class ValueToken
```

String tokens used for various purposes.

Since :

12/07/2007

Author :

vic

Constructor summary :

Constructor
ValueToken ()

Constructor detail :

```
public ValueToken ()
```

SHORTCUT AND HELPER FUNCTIONS

Several shortcut functions are available; these are usually wrappers for more complex operations that do type checking and conversions for convenience.

_ID

```
_id(String idOrPrefix, Object maybeDbId, Object maybeN0, Object maybeN1,
Object maybeN2, Object maybeN3)
```

Creates a full identifier from parts (prefix, database id and numerics). Please refer to specific objects' ID construction schemes; objects can use up to four numeric parts for an identifier.

Only one of them is the actual low-level identifier, while the others may be returned from the various API calls to help in processing and reduce the number of low-level API calls.

JS call:

```
_id(Prefix.FOLDER, 1, 1);
```

Parameters :

`idOrPrefix` - the prefix required to construct a full identifier; anything that can be cast to a

string, usually the corresponding part of another ID or one of the Prefix constants

`maybeDbId` - the database id part of the full identifier; it can be anything that can be cast to a number

`maybeN0` - the first numeric id part of the full identifier; it can be anything that can be cast to a number

`maybeN1` - the second numeric id part of the full identifier; it can be anything that can be cast to a number

`maybeN2` - the third numeric id part of the full identifier; it can be anything that can be cast to a number

`maybeN3` - the fourth numeric id part of the full identifier; it can be anything that can be cast to a number

Returns :

The identifier as a String value.

Throws :

Nothing

Example :

```
function execute() {
    var id = _id(Prefix.FILE_DOCF, 1, 10, 11, 12, 13);
    _info('  id = ' + _s(id));

    var dbId = _dbid(_s(id));
    _info('dbId = ' + dbId.toString());
}
```

Output :

```
    >    id = DOCF00010000000A0000000B0000000C0000000D
    >  dbId = DATA000100000000000000000000000000000000
```

_DBID

```
_dbid(Object unknown)
```

Creates a full database identifier for a given source; the source can be anything that can be interpreted

to get a database identifier such as another object identifier, a numeric (integer or long) or another object.

JS call:

```
_dbid(someDocument);
```

Parameters :

`unknown` - the source that will be used to extract the numeric part of the database identifier, then construct a database identifier

Returns :

The database identifier as a String.

Throws :

Nothing

Example :

```
function execute() {
    var id = _id(Prefix.FILE_DOCF, 1, 10, 11, 12, 13);
    _info('  id = ' + _s(id));

    var dbId = _dbid(_s(id));
    _info('dbId = ' + dbId.toString());
}
```

Output :

```
>    id = DOCF00010000000A0000000B0000000C0000000D
>  dbId = DATA0001000000000000000000000000000000000
```

_OBJ

```
_obj(Object objectSource)
```

Creates a GARGANTUA object from the source, which can be a wrapped Java object, an identifier or a key-value store

IMPORTANT : when creating objects, this helper will not load all properties from the database; for instance, when using an identifier as a parameter, it will crerate an object of the correct type, then set the identifier, but any other properties will have to be loaded from database; Always use the proper loading method for accessing the object's data.

JS call:

```
_obj(Object objectSource);
```

Parameters :

`objectSource` - the source info that will be used to create the proper object

Returns :

A GARGANTUA object.

Throws :

Nothing

_G

```
_g(String function, NativeObject args)
```

Helper to invoke a specific function passing a set of arguments.

JS call:

```
_g(String function, NativeObject args);
```

Parameters :

`function` - the function to call

`args` - arguments to pass to the function

Throws :

Nothing

_CACHED

```
_cached(String cacheKey, NativeFunction function, Object arg1, Object
arg2, Object arg3, Object arg4, Object arg5, Object arg6, Object arg7,
Object arg8, Object arg9)
```

Caches the given call using the given parameters; if the cached call is executed again, then the cached value is retrieved, allowing for faster response times.

JS call:

```
_cached(cacheKey, function, arg1, arg2, arg3, arg4, arg5, arg6, arg7,
arg8, arg9);
```

Parameters :

`cacheKey` - the cache key used to identify a call and return the proper result later

`function` - the function call to cache

`arg1 .. arg9` - arguments to pass to the call; optional, depending on the function call

Returns :

The result of the function applied to the given parameters; if the result is caches, then the function call is not execute4d and the cached result is returned.

Throws :

Nothing

_P, _PLUGIN

```
_p(String pluginId, String cmd, NativeObject args)
_plugin(String pluginId, String cmd, NativeObject nullableArgs)
```

Invokes a given command from a specific plugin.

JS call:

```
_dbid(pluginId, cmd, args);
```

```
_plugin(pluginId, cmd, args);
```

Parameters :

`pluginId` - the plugin to invoke; use the short name of the plugin (the part after 'siatel-plugin')

`cmd` - the command to execute within the plugin

`args` - the DTO parameters to pass to the command being invoked; these depend on the plugin command specifics

Returns :

The result of the plugin invoke, as a DTO.

Throws :

Nothing

_SLEEP

```
_sleep(Object millis)
```

Sleep for the given interval.

JS call:

```
_sleep(millis);
```

Parameters :

`millis` - an object that can be converted to a long representing the number of milliseconds to sleep.

Throws :

Nothing

_LOG

```
_log(String message, String level)
```

Logs the provided message at the givel log level.

JS call:

```
_log(message, level);
```

Parameters :

`message` - the message to log

`level` - the log level at which the message must be loggeg; corresponds to the standard Java logging levels (ERROR, WARN, INFO, DEBUG, etc.)

Throws :

Nothing

_DEBUG

```
_debug(String message)
```

Logs the provided message at DEBUG level.

JS call:

```
_debug(message);
```

Parameters :

`mesage` - the message to log

Throws :

Nothing

_INFO

```
_info(String message)
```

Logs the provided message at INFO level.

JS call:

```
_info(message);
```

Parameters :

`mesage` - the message to log

Throws :

Nothing

_WARN

```
_warn(String message)
```

Logs the provided message at WARN level.

JS call:

```
_warn(message);
```

Parameters :

`mesage` - the message to log

Throws :

Nothing

_ERROR

```
_error(String message)
```

Logs the provided message at ERROR level.

JS call:

```
_error(message);
```

Parameters :

`mesage` - the message to log

Throws :

Nothing

_ENTER

```
_enter(String _app, String _package, String _function, String level)
```

Helper used to log the entry point to a specific call.

JS call:

```
_enter(_app, _package, _function, level);
```

Parameters :

`_app` - the name of the application to which the function belongs

`_package` - the name of the package to which the function belongs

`_function` - the name of the function

`level` - the level at which to log the message

Throws :

Nothing

_EXIT

```
_exit(String _app, String _package, String _function, String level)
```

Helper used to log the exit point to a specific call.

JS call:

```
_exit(_app, _package, _function, level);
```

Parameters :

`_app` - the name of the application to which the function belongs

`_package` - the name of the package to which the function belongs

`_function` - the name of the function

`level` - the level at which to log the message

Throws :

Nothing

_L

```
_l(Object unknown)
```

Converts the given parameter to a long value, if possible.

JS call:

```
_l(object);
```

Parameters :

`unknown` - the object to attempt to convert to a long value

Returns :

A long value

Throws :

Nothing

_I

```
_i(Object unknown)
```

Converts the given parameter to an integer value, if possible.

JS call:

```
_i(object);
```

Parameters :

`unknown` - the object to attempt to convert to a integer value

Returns :

An integer value

Throws :

Nothing

_D

```
_d(Object unknown)
```

Converts the given parameter to a Date value, if possible.

JS call:

```
_d(object);
```

Parameters :

`unknown` - the object to attempt to convert to a Date value

Returns :

A Date object

Throws :

Nothing

_S

```
_s(Object unknown)
```

Converts the given parameter to a String value, if possible.

JS call:

```
_s(object);
```

Parameters :

`unknown` - the object to attempt to convert to a String value

Returns :

A String value

Throws :

Nothing

_B

```
_b(Object unknown)
```

Converts the given parameter to a boolean value, if possible.

JS call:

```
_b(object);
```

Parameters :

`unknown` - the object to attempt to convert to a boolean value

Returns :

A boolean value

Throws :

Nothing

_LIST

```
_list(Object unknown)
```

Converts the given parameter to a List value, if possible.

JS call:

```
_list(object);
```

Parameters :

`unknown` - the object to attempt to convert to a List value

Returns :

A List object

Throws :

Nothing

_ARRAYLIST

```
_arrayList(Object unknown)
```

Converts the given parameter to an ArrayList value, if possible.

JS call:

```
_arrayList(object);
```

Parameters :

`unknown` - the object to attempt to convert to an ArrayList value

Returns :

An ArrayList object

Throws :

Nothing

_DTO

```
_dto(Object unknown)
```

Converts the given parameter to a DTO object, if possible.

JS call:

```
_dto(object);
```

Parameters :

`unknown` - the object to attempt to convert to a DTO structure

Returns :

A DTO structure

Throws :

Nothing

_BOMIFIED

```
_bomified(String text)
```

Adds the UTF-8 BOM marker (0xEF 0xBB 0xBF) to the given string

JS call:

```
_bomified(text);
```

Parameters :

`text` - the text to add the BOM marker to

Returns :

The original string with the BOM marker

Throws :

Nothing

_DEBOMIFIED

```
_debomified(String text)
```

Removes the UTF-8 BOM marker (0xEF 0xBB 0xBF) from the given string

JS call:

```
_debomified(text);
```

Parameters :

`text` - the text to remove the BOM marker from

Returns :

The original string without the BOM marker

Throws :

Nothing

_OR

```
_or(NativeArray arrBits)
```

Performs a bitwise or on the given parameters.

JS call:

```
_or(arrBits);
```

Parameters :

`arrBits` - the values for which to perform the bitwise or operation.

Returns :

A long value representing the result of the operation

Throws :

Nothing

_AND

```
_and(NativeArray arrBits)
```

Performs a bitwise and on the given parameters.

JS call:

```
_and(arrBits);
```

Parameters :

`arrBits` - the values for which to perform the bitwise and operation.

Returns :

A long value representing the result of the operation

Throws :

Nothing

_DF

```
_df(Object maybeDate, String format)
```

Formats the given parameter using the provided format string.

JS call:

```
_df(maybeDate, format);
```

Parameters :

`maybeDate` - the Date object or an object that can be converted to a Date

`format` - the format

Returns :

A String value that holds the formatted Date.

Throws :

Nothing

_NOW

```
_now(Object format)
```

Returns the 'now' moment according to the requested format.

JS call:

```
_now(format);
```

Parameters :

`format` - either one of 'millis' or a formatting string; optional.

Returns :

If the format is not specified, the function returns a Date object initializes to the current moment.
If the format is 'millis', the function returns a Long value for the current moment (equivalent of `System.currentTimeMillis()`).

If the format a string value, the function tries to format the current moment using the provided format.

Throws :

Nothing

SPECIAL FUNCTIONS

MISC.SIGNREQUEST

The signRequest function will schedule or start a new sign envelope using the input data.

When starting a new sigtn envelope, two possibilities are available :

1. Start a new sign envelope by providing all data required for the signing process
2. Start a new sign envelope by using a previously saved template and providing only the specific data

that is either not present or not yet defined in the template.

Starting a new sign envelope by using an existing template is easier and faster, since not all required data must be provided from the script.

SIGN ENVELOPE

A sign envelope has several blocks of data, named stages :

1. start stage : provides general information about the sign envelope and start conditions
2. sign stage : provides all required information about the signatories
3. publish stage : provides information about final processing when saving back to the EDM database
4. broadcast stage : provides information about the recipients for the document broadcast

When triggering a new sign envelope by scripting, a JSON object (currently referred to as cycleData) is used. Please refer to the following sections for a description of this object.

The cycleData object

The cycle data object has the following object keys :

template

The name or the identifier of a sign envelope template

manager

The name or the identifier of the manager of this sign envelope; by default, unless otherwise specified, this manager is also the manager for all the stages of the signing envelope.

owner

The name or the identifier of the owner of this sign envelope.

overwrite

Boolean flag to define whether the data from the template is to be merged or overwritten; please see the Merge or overwrite sign envelope data section for more detailed information on this behavior

lockDocument

Boolean flag to define whether the document(s) are to be locked while the signing envelope is active

publicUsers

Boolean flag to define whether all users may access the sign envelope history; be default, only the manager may view the history data.

docValidity

The document validity once the sign envelope is finished (all signing parties have finished signing); must be used exclusively with the docExpiration data

docExpiration

The expiration date of the documentonce the sign envelope is finished (all signing parties have finished signing); must be used exclusively with the docValidity data

comment

A text that will be displayed and sent to the signing parties as part of the notification messages

stages

An object providing data for each stage of the sign envelope; see the Sign envelope stages section for more details.

docs

An array of documents to be handled by this sign envelope; each document must either be signed by at least one signatory or is an attachment that is displayed for user reference only. See the Documents section for more details on document types and signature positioning.

Below is an example of a cycle data object :

```
var cycleData = {
    template  : '<id-or-label>',
    manager   : ' <id-or-label>',
    overwrite : true, // false means 'merge', for 'overwrite' set to true
    stages    : {
          start     : {
              // start stage data
          },
          sign        : {
              // sign stage data
          },
          publish   : {
              // publish stage data
          },
          broadcast : {
              // broadcast stage data
          }
    },
```

```
    docs         : [
           {
               // documents to sign or attachments
           }
    ]
};
```

The above structure is just a brief skeleton of the cycleData object; more detailed information and a complete example are provided on the following sections

Merge or overwrite sign envelope data

Sign envelope stages

This section describes the sign envelope stages as well as provide an example.

The stages object provides information about the sign envelope stages as an object with keys for each stage, like in the structure below (it is not a full example)

```
stages     : {
       start      : {
           // start stage data
       },
       sign         : {
           // sign stage data
       },
       publish    : {
           // publish stage data
       },
       broadcast : {
           // broadcast stage data
       }
}
```

The following sections will detail each stage data.

Start stage

Available fields

overwrite

Boolean flag to indicate whether data is to be merged into the provided template or not

manager

The user name or identifier of the manager for the start stage

maxDuration

The overall maximum duration of the sign envelope

startDate

The start date of the sign envelope; this can be a date in the future, and all signing parties will be able to sign the document starting with this date.

The following code is an example of a start stage object :

```
start     : {
    overwrite   : true,
    manager     : '...',
    maxDuration : '...',
     startDate    : '...'
}
```

Sign stage

Available fields

docHasMarkers

Boolean to indicate that documents that are subject of the sign operation provide document markers for signature positioning

data

An array of objects providing information on all signing parties for the sign envelope

The following code is an example of a sign stage object

```
sign        : {
    // sign stage data
}
```

Publish stage

Available fields

manager

The user name or id of the manager for the start stage

The following code is an example of a publish stage object

```
publish   : {
    // publish stage data
}
```

Broadcast stage

Available fields

manager

The user name or id of the manager for the start stage

The following code is an example of a broadcast stage object

```
broadcast : {
    // broadcast stage data
}
```

Documents

A COMPLETE EXAMPLE

Scripts pour les modules complémentaires

Scripting

AddressBook

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

ADDCOLUMN

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```

addColumn

```
addColumn(props)
```

Adds a column specified by provided props to the grid.

Parameters

props

An object with the column properties

Return value

None

Example

```
var props = {
     'id': 'productName',
     'type': 'string',
     'label': 'Product',
     'tooltip': 'Product name',
     'align': 'left',
     'width': '250px'
};
grid.addColumn(props);
```