OpenAir User Scripting Reference Card

NSOA Functions

context

- + **getAllParameters() : Array**
- + **getAllTerms() : Array**
- + **getParameter(*name*) : *var***
- + **getTerm(*termid*) : *String***
- + **isTestMode() : Boolean**
- + **parseTerminology(*message*) : String**
- + **remainingTime() : Integer**
- + **remainingUnits() : Integer**

form

- + **confirmation(*message*) : Boolean**
- + **error(*field, message*) : Boolean**
- + **getAllValues() : *Array***
- + **getLabel(*field*) : *String***
- + **getName(*field*) : *String***
- + **getNewRecord() : oaBase**
- + **getOldRecord() : oaBase**
- + **getValue(*field*) : *var***
- + **get_value(*field*) : *String***
- + **warning(*message*) : Boolean**
- + **setValue(*field, value*) : *String***

https

- + **delete(*request*) : Object**
- + **get(*request*) : Object**
- + **patch(*request*) : Object**
- + **post(*request*) : Object**
- + **put(*request*) : Object**

listview

- + **data(*listviewId*) : Iterator**
- + **list() : Array**

meta

- + **alert(*message*) : Boolean**
- + **log(*severity, message*) : Boolean**
- + **sendMail(*message*) : Boolean**

NSConnector

- + **integrateAllNow () : Boolean**
- + **integrateRecord () : Boolean**
- + **integrateWorkflowGroup(*name*) : Boolean**

record

- + **< complexType > (*[id]*) > : oaBase**

report

- + **data(*reportId, optionalParameters*) : Iterator**
- + **list() : Array**

wsapi

- + **add(*objects*) : Array**
- + **approve(*approveRequests*) : Array**
- + **delete(*objects*) : Array**
- + **disableFilterSet(*[flag]*) : Boolean**
- + **enableLog(*[flag]*) : Boolean**
- + **modify(*attributes, objects*) : Array**
- + **read(*readRequest*) : Array**
- + **reject(*rejectRequests*) : Array**
- + **remainingTime() : Integer**
- + **submit(*submitRequests*) : Array**
- + **upsert(*attributes, objects*) : Array**
- + **unapprove(*unapproveRequests*) : Array**
- + **whoami() : oaUser**

NSOA.form.error(field, errorMessage)

```
// displays 'An error message' next to the budget_time field
function formError() {
    NSOA.form.error("name","An error message");
}
```

- Call this function to stop the form from saving
- Use this function to print an error message associated to the selected form field on the OpenAir form.
- The first argument is the field name on the form where you want the message to show up. Pass an empty string to display the message at the top of the form.
- The message that appears looks exactly like the OpenAir system generated errors.

NSOA.form.getValue(field)

```
// writes the field value to the log
function logFieldValue() {
    NSOA.meta.alert('The date is ' + NSOA.form.get_value('date'));
}
```

NSOA.form.getAllValues()

```
//  get all the values on the fields on the form and write to log
var allValues = NSOA.form.getAllValues();
for( var key in allValues ) {
    NSOA.meta.alert(key + ' has value ' + allValues.key);
}
```

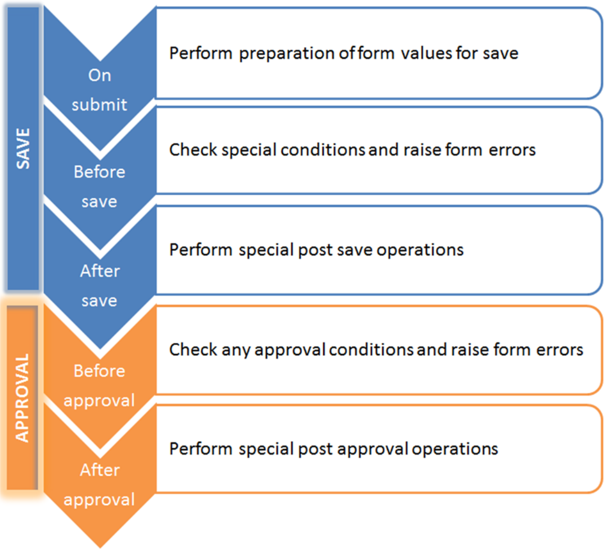
NSOA.meta.log(type,message)

```
// writes an error severity message to the log
function logMessage() {
    NSOA.meta.log('error', 'Travel date is after receipt date');
}
// Hint: see also the 'Form script deployment logs' detail report
```

NSOA.meta.alert(message)

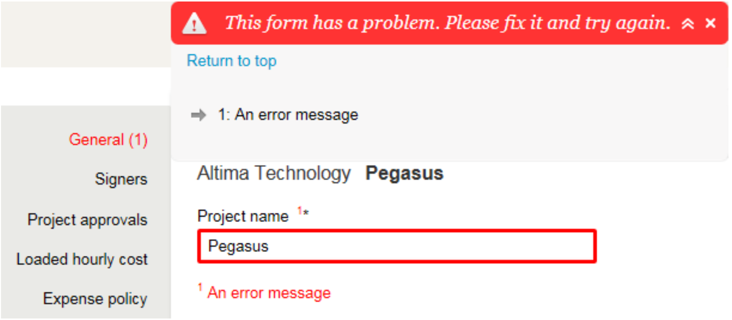
```
// writes an info severity message to the log
function logMessage() {
    NSOA.meta.alert('Travel date is after receipt date');
}
```

Events



- On submit: Perform preparation of form values for save
- Before save: Check special conditions and raise form errors
- After save: Perform special post save operations
- Before approval: Check any approval conditions and raise form errors
- After approval: Perform special post approval operations

Example



This form has a problem. Please fix it and try again.

Return to top

1: An error message

Altima Technology **Pegasus**

Project name [1]*

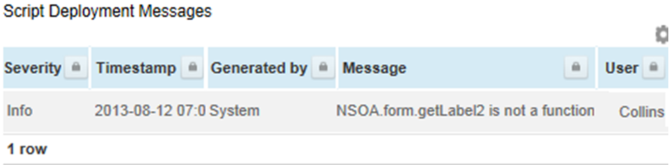
Pegasus

[1] An error message

Validation – Before save

```
// compare two date fields on a receipt
// NOTE: call on 'Before save' form event
function validateTravelDates() {
    var receiptDate = NSOA.form.getValue('date');
    var travelDate = NSOA.form.getValue('TravelDate__c');
    if ( receiptDate < travelDate ) {
    NSOA.form.error('TravelDate__c',
        'The travel date cannot be after the receipt date!');
    }
}
```

Log



Script Deployment Messages

Severity	Timestamp	Generated by	Message	User
Info	2013-08-12 07:0	System	NSOA.form.getLabel2 is not a function	Collins

1 row

Severity	Fatal, Error, Warning, Info, Debug, or Trace
Timestamp	time the message was logged
Generated by	your script or the system
Message	full message text
User	who was running the script

OpenAir User Scripting Reference Card

NSOA.record.<complex type>([id])

```
function testCreatingObjects {
    // create category object populated with data for id = 5
    var populatedCategory = NSOA.record.oaCategory(5);
    // create a blank category object
    var blankCategory = NSOA.record.oaCategory();
}
```

NSOA.wsapi.add(objects)

- See the example on the right
- The function takes an array of objects
- [category] // This is an array with one category object
- You can pass up to 1000 objects
- The ID is created by the system (returned in the result)

NSOA.form.getNewRecord()

- See the example on the right
- Gets the entity record for a form with the newly saved values
- This function should be called on the **After save event**
- The function returns **null** if called before the form has been saved
- **NSOA.form.getOldRecord()** returns the current (unsaved) values

NSOA.wsapi.disableFilterSet([flag])

```
function testDisableFilterSet {
    // disables user filter sets on .wsapi requests
    NSOA.wsapi.disableFilterSet(true);
    // enables user filterset on .wsapi requests (default)
    NSOA.wsapi.disableFilterSet(false);
    // returns the Boolean state (without changing setting)
    var filtersDisabled = NSOA.wsapi.disableFilterSet();
}
// Hint: set NSOA.wsapi.enableLog(true) to show SOAP messages
```

Scripting Governance

To prevent scripts from consuming excessive resources or running out of control, two limitations are placed on scripts:

Time Limit — If a form script runs for more than 5 seconds (not including wsapi call time) it is automatically terminated with a form error.

If a request (all scripts triggered by the same form save) uses more than 60 seconds of wsapi time it will also be automatically terminated.

Scheduled scripts are allowed 1 hour of JS runtime and 1 hour of wsapi.

Units Limit — NSOA functions are assigned a unit value. Each time an NSOA function is called its unit value is consumed. A form script is allowed to consume a maximum of 1000 units with each run before it is automatically terminated with a form error.

Scheduled scripts are allowed 1, 000, 000 units.

SendMail Limit — Forms scripts are allowed to send a maximum of 3 emails on a single event.

Scheduled scripts are allowed to send a maximum of 100 emails on a single scheduled event.

NSOA.context.remainingUnits()

```
// displays remaining units as a form error message
function displayRemainingUnits () {
    NSOA.form.error('','Units: ' + NSOA.context.remainingUnits());
 }
```

Editor Shortcuts

<Ctrl> + <SPACE>	**Show Auto List (after “NSOA.”)**
<Ctrl> + <F11>	**Full-screen editor (<Esc> to exit)**
<Ctrl> + F	**Start a search**
<Ctrl> + G	**Find next**
<Shift> + <Ctrl> + G	**Find previous**
<Shift> + <Ctrl> + F	**Replace**
<Shift> + <Ctrl> + R	**Replace all**
<Alt> + G	**Activate Jump to Line dialog**

Creating Records

```
// create a new category
function createCategory {
    // create an empty category object
    var category = new NSOA.record.oaCategory();
    // fill out the details – the ID is created by OpenAir
    category.name = "New Category";
    category.cost_centerid = "123";
    category.currency = "USD";
    // invoke the add call
    var results = NSOA.wsapi.add( [category] );
    // log the ID of the new category
    NSOA.meta.alert(‘The category id is  ’ + results[0].id);
}
```

Modifying Records

```
// This is called on the "After save" event for the Project form
function append_to_project_notes() {
    // get the saved project values
    var newr = NSOA.form.getNewRecord();
    // Invoke the add call
    var project = new NSOA.record.oaProject();
    project.id = newr.id;    // use the saved project ID
    project.notes = newr.notes + "₩nAppended: " +  (new Date().toString());
    // disable user filters to make sure it runs for all users
    NSOA.wsapi.disableFilterSet(true);
    // invoke the modify call
    var arrayOfupdateResult = NSOA.wsapi.modify([], [project]);
    // log the modify status
    NSOA.meta.alert("Got modify status: " + arrayOfupdateResult[0].status);
}
```

Governance Units

NSOA.context.getTerm(termid)
NSOA.context.isTestMode()
NSOA.context.remainingTime()
NSOA.context.remainingUnits()
NSOA.record.<complex type>([id])
NSOA.wsapi.remainingTime()
	0

NSOA.context.getParameter(name)
NSOA.form.confirmation(message)
NSOA.form.error(field, message)
NSOA.form.getLabel(field)
NSOA.form.getName(field)
NSOA.form.getNewRecord()
NSOA.form.getOldRecord()
NSOA.form.getValue(field)
NSOA.form.get_value(field)
NSOA.form.warning(message)
NSOA.listview.list(), NSOA.report.list()
NSOA.wsapi.disableFilterSet([flag])
NSOA.wsapi.enableLog([flag])
NSOA.wsapi.whoami()
	1

NSOA.context.parseTerminology(message)
NSOA.meta.alert(message)
NSOA.meta.log(severity, message)
	4

NSOA.context.getAllParameters()
NSOA.context.getAllTerms()
NSOA.form.getAllValues()
NSOA.https.delete(request)
NSOA.https.get(request)
NSOA.https.patch(request)
NSOA.https.post(request)
NSOA.https.put(request)
NSOA.meta.sendMail(msg)
NSOA.NSConnector.integrateRecord()
	10

NSOA.listview.data(listviewId)
NSOA.report.data(reportId, optionalParameters)
	10 for each page loaded into iterator on demand (max 1000 items per page)

NSOA.wsapi.read(readRequest)
NSOA.wsapi.add(objects)
NSOA.wsapi.delete(objects)
	20 (+10 per additional object)

NSOA.wsapi.modify(attributes, objects)
NSOA.wsapi.upsert(attributes,objects)
	40 (+20 per additional object)

NSOA.NSConnector.integrateAllNow()
NSOA.NSConnector.integrateWorkflowGroup(name)
	1000

Scripting Documentation

OpenAir User Scripting Guide
OpenAir SOAP API Reference Guide
OpenAir Business Intelligence Connector Guide