

Nuxeo Platform 5.8 Release Notes

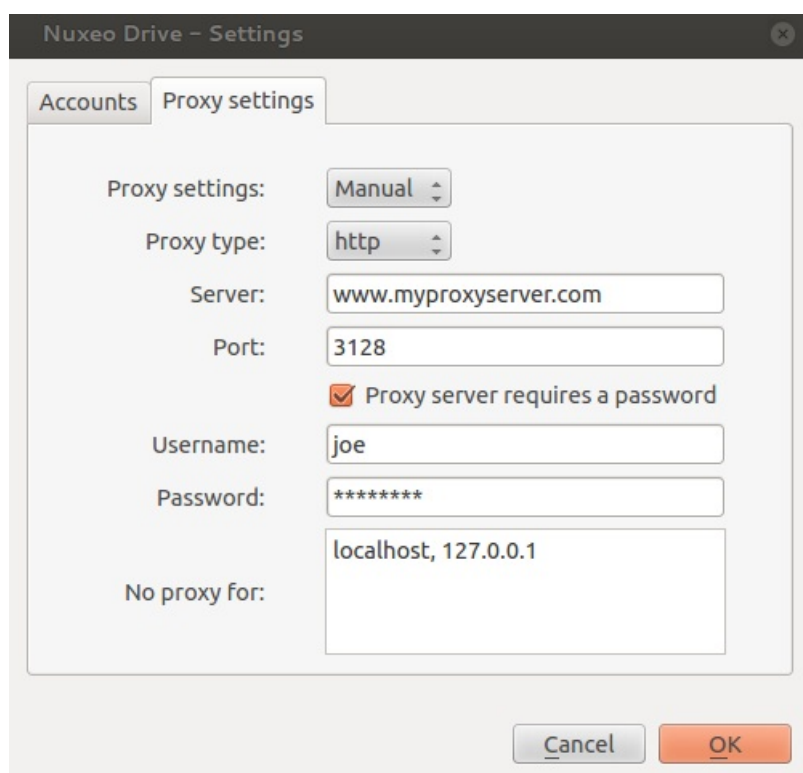
These release notes are a merge of the previous Fast Track release notes available [here](#).

For Users

Nuxeo Drive

Nuxeo Drive enables bidirectional synchronization of content between the local desktop and the Nuxeo content repository, on premise or in the Cloud. It works with all applications built on the Nuxeo Platform, including Document Management, Digital Asset Management, Case Management, or a custom content-centric application.

Nuxeo Drive can be used behind a proxy.

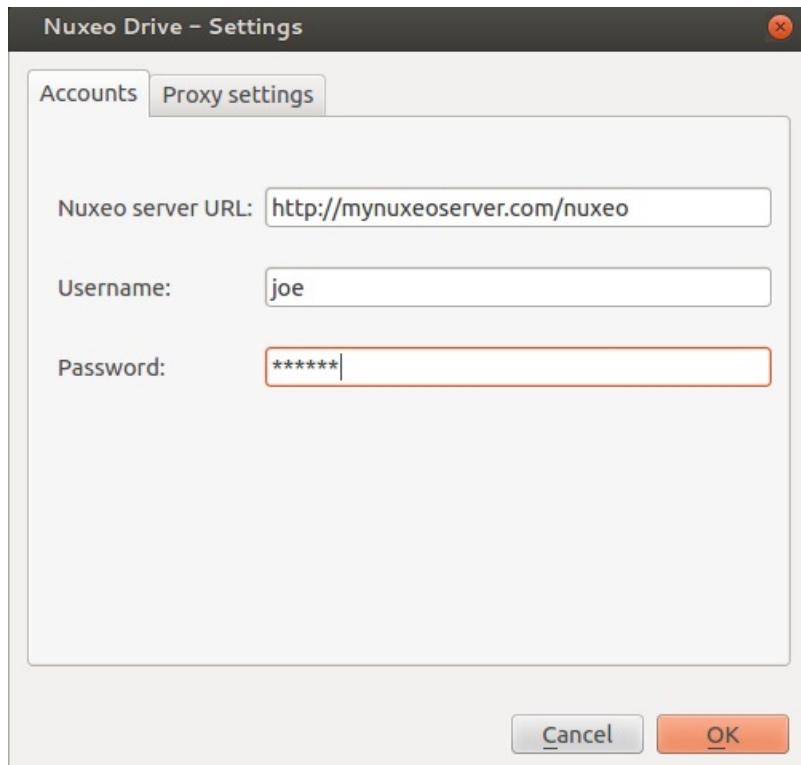


The screenshot shows the 'Nuxeo Drive - Settings' dialog box with the 'Proxy settings' tab selected. The dialog has two tabs: 'Accounts' and 'Proxy settings'. The 'Proxy settings' tab contains the following fields and controls:

- Proxy settings:** A dropdown menu set to 'Manual'.
- Proxy type:** A dropdown menu set to 'http'.
- Server:** A text input field containing 'www.myproxyserver.com'.
- Port:** A text input field containing '3128'.
- Proxy server requires a password:** A checked checkbox.
- Username:** A text input field containing 'joe'.
- Password:** A text input field containing '*****'.
- No proxy for:** A text input field containing 'localhost, 127.0.0.1'.

At the bottom of the dialog are two buttons: 'Cancel' and 'OK'.

We also added the unified settings panel:



The image shows a 'Nuxeo Drive - Settings' dialog box. It has two tabs: 'Accounts' and 'Proxy settings'. The 'Accounts' tab is active. It contains three input fields: 'Nuxeo server URL' with the value 'http://mynuxeo.com/nuxeo', 'Username' with the value 'joe', and 'Password' with masked characters '*****'. At the bottom right, there are 'Cancel' and 'OK' buttons.

DAM

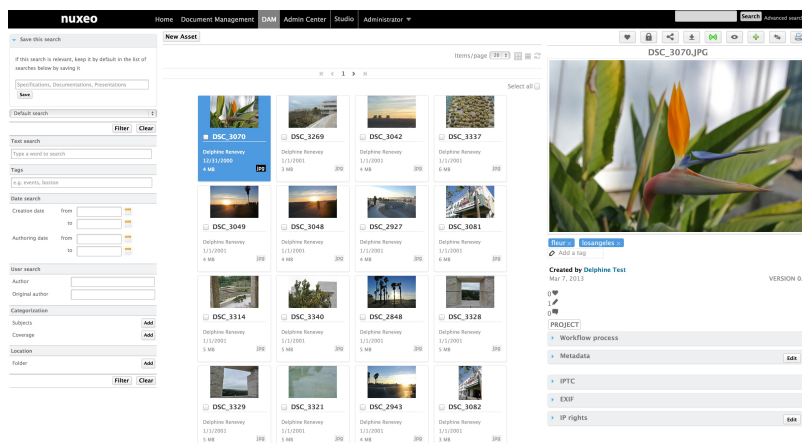
Customizable with Nuxeo Studio

Redesign from the ground up using ContentView, Layouts and widgets. It makes every part of DAM screen (search panel, result panel, asset view) easy to customize using Nuxeo Studio.

- Define the Left Hand Side with your own search layout
- Define the layout for the fields displayed in the thumbnail
- Define the Right Hand Side data for the selected asset (view/edit, as toggable layout)
- Define the Bulk-edit layout
- Define toolbar buttons, add your own
- Define the document types DAM can handle

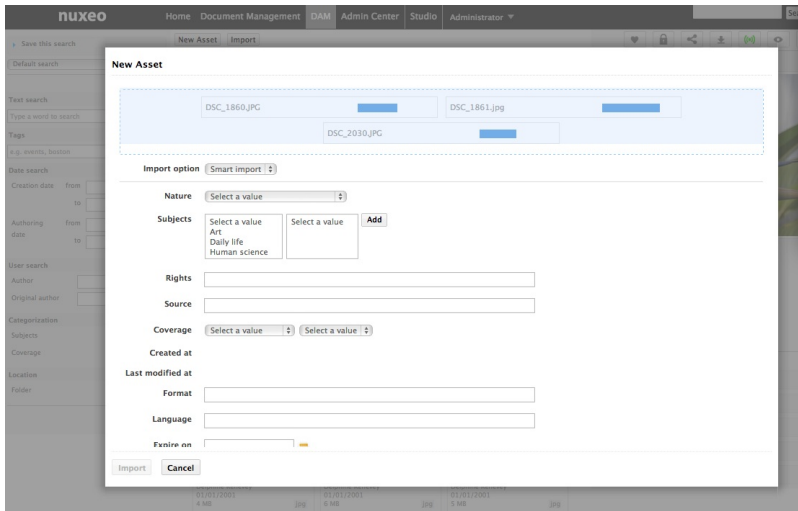
Browse all Assets from the Repository

All the assets of the repository are displayed (not just the one under "Media Library")



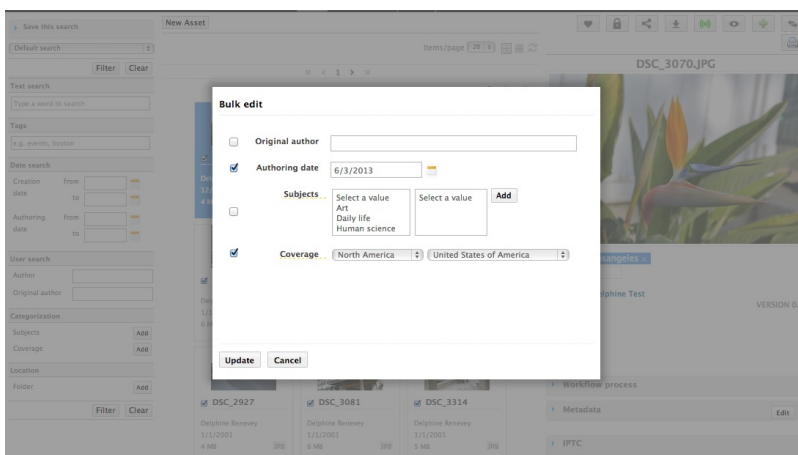
DAM Drag and Drop

You can now drag and drop as many assets as you want in the New Asset window:



Assets Browsing

- "New Asset" button:
 - Choose among appropriate document types with different metadata
- Import with D&D
 - Also in "List" view
- New "Items/Page" dropdown
- New "Refresh" button
- New "Select all" button
- New "Copy" button
- New "Add to worklist" button



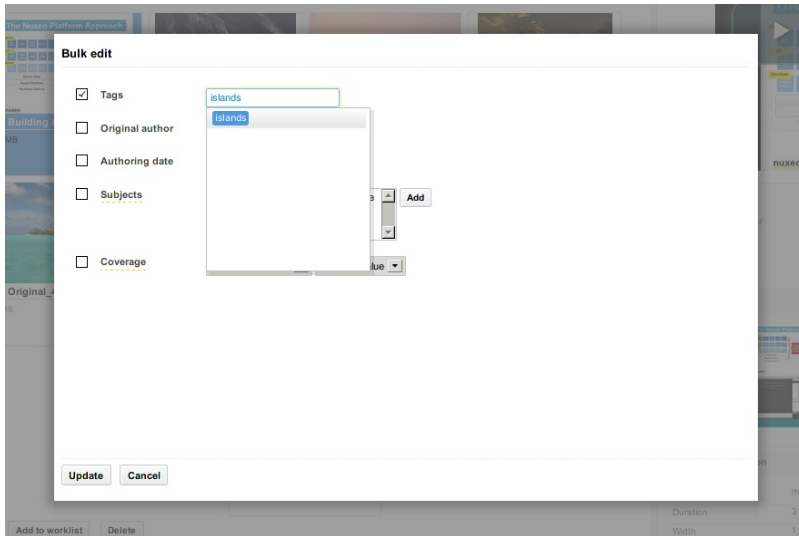
Asset View

- New toolbar, with the following buttons:
 - Like
 - Lock
 - Permanent link
 - Follow
 - Preview
 - Add to worklist
 - Go to DM view

- Print
- New "Tags" feature
 - Multiple tags
 - Click on a tag to display the list of assets using it
- New "Creator/LifeCycle/Version" display
- New count of people who liked the document
- Smart "Comments" group: Displayed only if there is at least one comment
- Access to workflow actions directly from the DAM UI

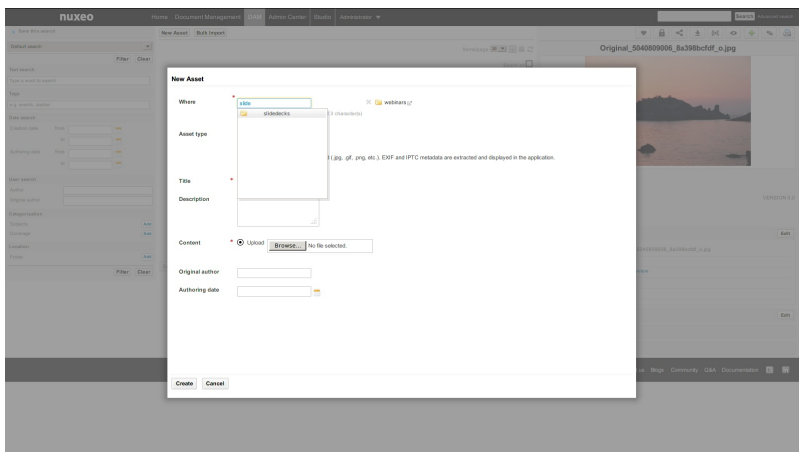
Bulk Tagging

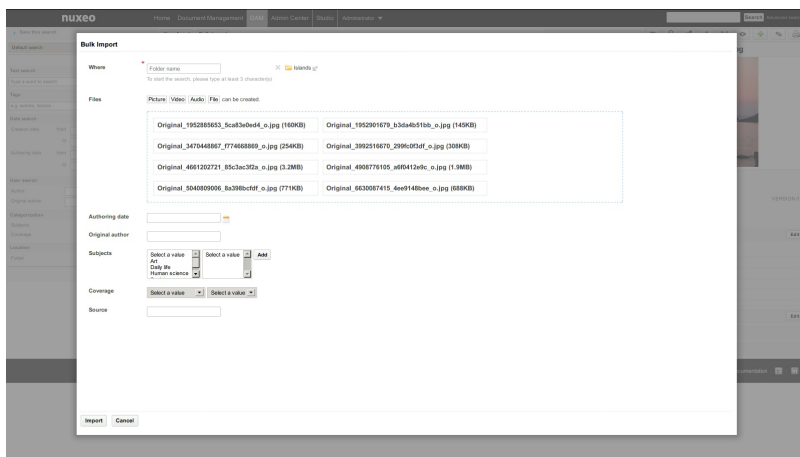
In the bulk edit form, you can now add tags to the selected list of documents.



Asset Creation

When creating a new asset or bulk importing assets, you can choose the location where to import them. Only the folders where the user has the create permission are selectable.



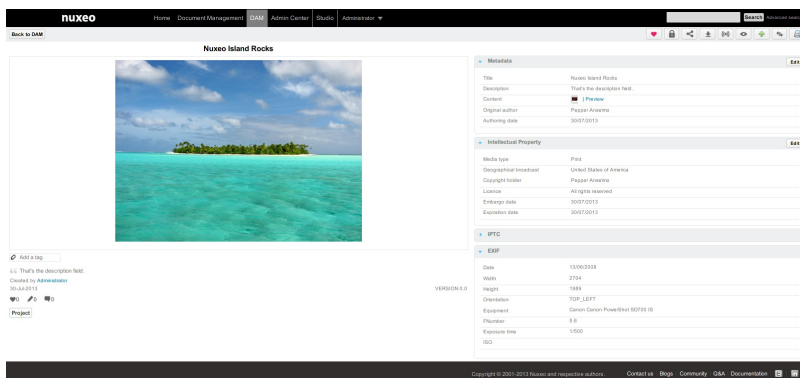


Assets View

The view or edit layouts configured on the Document types (the ones you can configure on your own document type on Studio for instance) are displayed in the Metadata section.

Single Asset View

Improve the single asset view of an asset (new layout accessible through the permalink).



Misc Improvements

Now you can use the HTML5 DnD on the Bulk import form.

If your import fails for some reason (for instance, if you forgot to fill a required field), you get an appropriate error message and you don't have to select DnD files to import again.

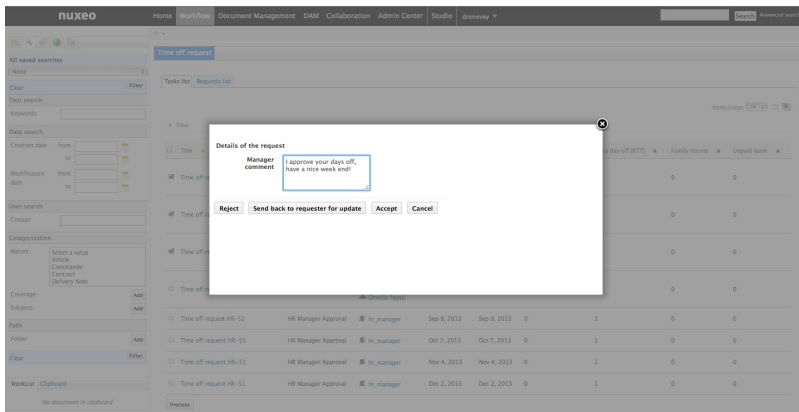
Workflow

A main workflow tab can be enabled to list running workflow tasks. This way you can select several tasks, and hence do bulk validation.

[Content routing prefetch](#) has been made more efficient, which improves performances.

[Redirect user](#) when she has no more rights to read the document after changing the state of the workflow.

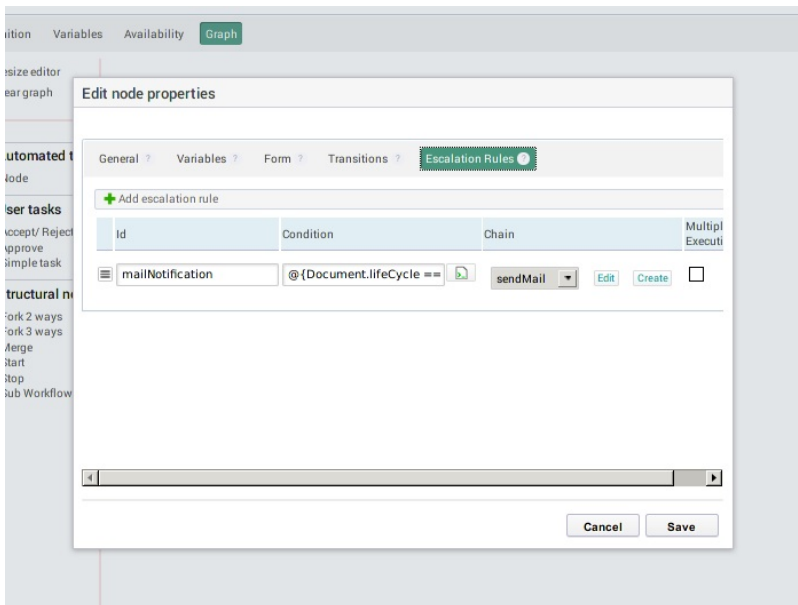
[Filter system workflow](#) related documents from search.



Issues List

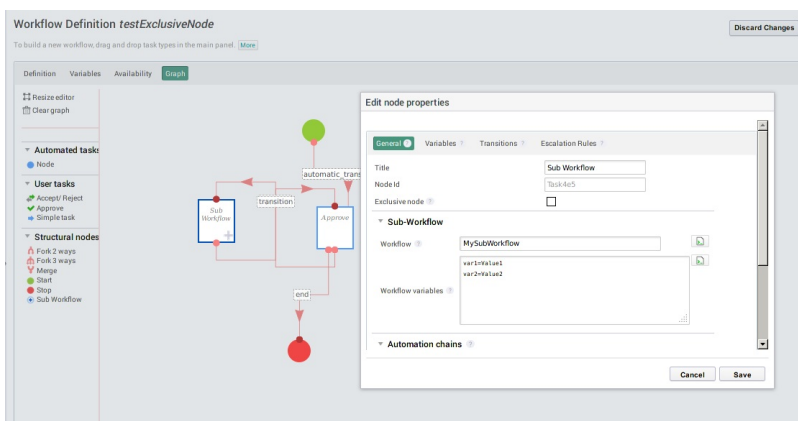
Escalation Rules

Escalation rules let you run an operation chain when a condition is true. These conditions are evaluated every 5 minutes by a configurable scheduler. They can be executed one time (default behaviour) or multiple times.



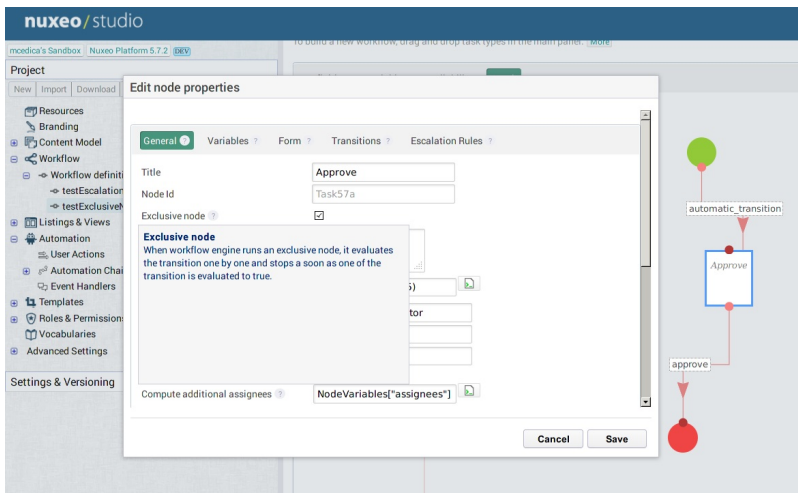
Sub-Workflow

We have added a new node called SubWorkflow. As you have already guessed it, it lets you embed sub-workflow into existing one. You can also give them parameters, escalation rules, input and output chains, and transitions, like a regular node.



Exclusive Node

When the workflow engine runs an exclusive node, it evaluates the transitions one by one and stops at the first one evaluated to true.



New Workflow Operations

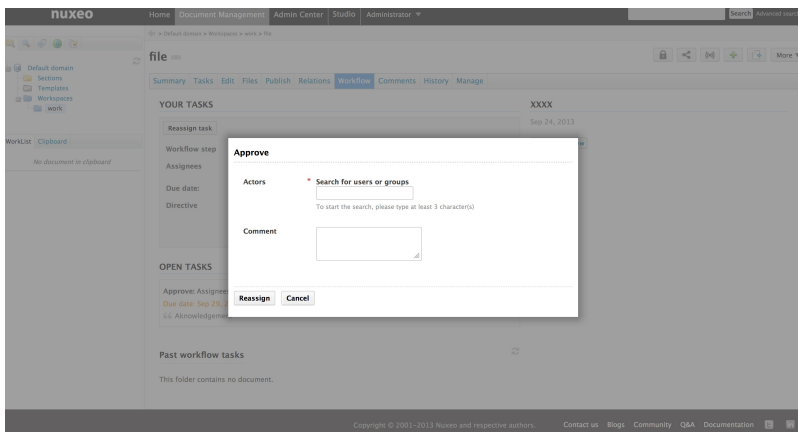
We've added new operations to cancel a workflow, resume a node, get the opened tasks, and complete a task and also improved the *Start Workflow* and *Set Workflow variable* operations.

New Workflow Tab

We have a new workflow tab that lets the user see all his assigned tasks at once and filter them.

Task Reassignment

Workflow tasks can now be reassigned to someone else.

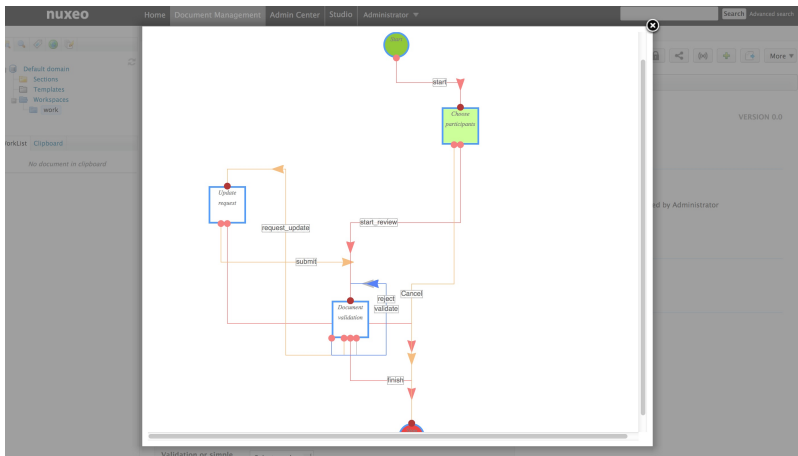


MultiTask Workflow Node

We have introduced a new option that creates a different task per assignee.

Workflow Graph Visualization

The workflow graph can be visualized as it has been created in Studio.



We now have a task resolution screen. It's a dedicated screen to solve one of your task and stay focus on the workflow.

Title	Modified	Last contributor	Version	State
Press release	10/23/2013	Laurent Doguin	0.0	Project

A workflow can be applied on multiple documents.

We added a parallel review workflow:

- First step, choose the participants
- Second step, every participants give their feedback.
- Third step, we present a summary of the 2nd step and the assignee can validate or reject the document.

Name	Actors	Directive	Comments	Start date	Due date
Consultation	Julie Allouch	Please give your opinion. OK for me		Oct 23, 2013	Oct 25, 2013

Mobile

A new application based on [Apache Cordova](#), bringing multi-OS support. It can be used as a framework to build your own mobile application based on Nuxeo. Using Nuxeo Webengine and JQuery Mobile, it proposes the following features: Saved search, browsing, upload and download of documents, integration with native phone API.

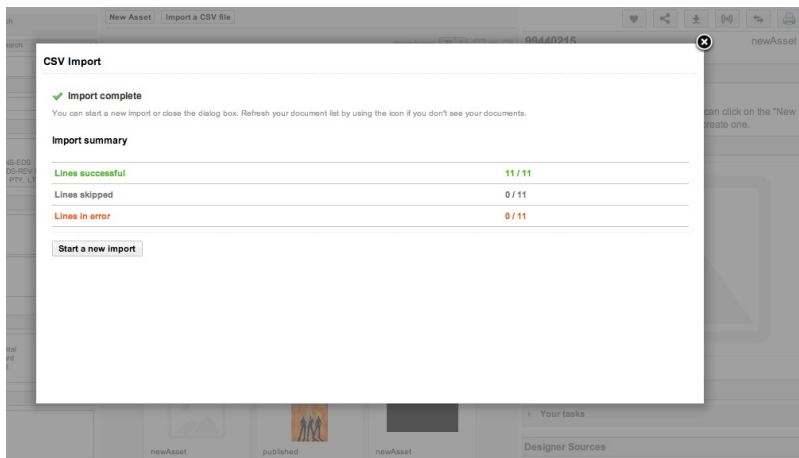
CSV Importer

We have a new module allowing import of documents from a CSV file ⁽¹⁾. Here's how it works: click on Import a CSV,

choose the CSV file, click on import and documents will be created using properties from the CSV (only scalar metadata).

You can specify a folder in Nuxeo.conf where the importer will look for binaries. Just make sure you put their relative path in the CSV.

(1) CSV import sample



User Experience

The platform screens have been redesigned and improved. More than eighty issues about the user experience, user interface and styles have been resolved.

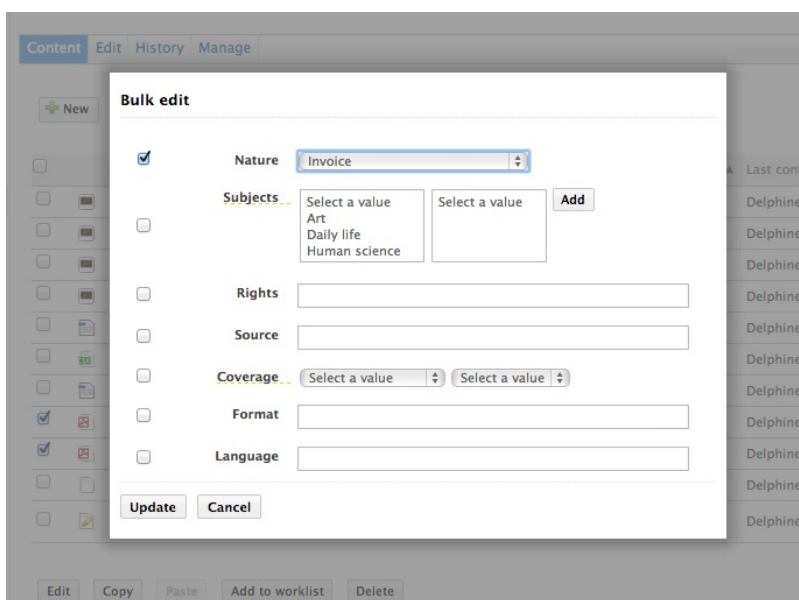
Safe Edit: Never loose your Forms

When editing a document, if the user tries to exit a page with unsaved changes then a popup window asks him if he really wants to leave the page (and if so discard the changes) or if he wants to stay on the page.

Where supported (i.e. localStorage available in the browser), a modification in the edited form triggers the autosave mechanism which records the new input values of the form. If the editing is accidentally lost (e.g. system / browser crash, network failure, etc ...), then the next time the user edits the page, the previous modifications will be detected from the local storage and the user will be asked whether to restore them or not.

Bulk Edit in listing

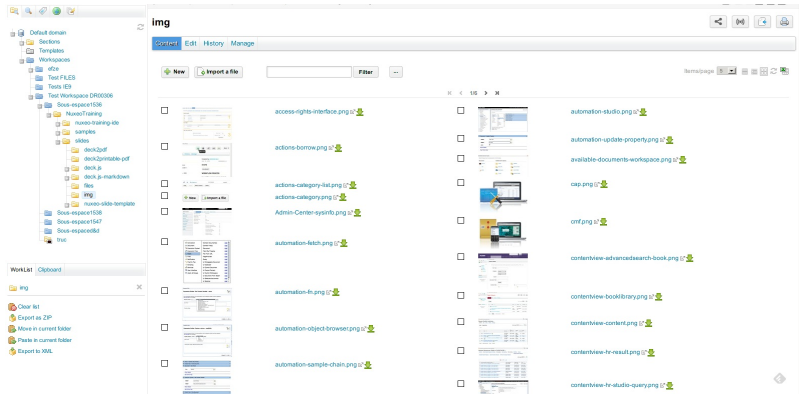
We have enabled **Bulk Editing** for document listing. It's now easy to show a form in a popup, displaying any layout you want, and that applies to documents you've selected.



Content Thumbnails

Thumbnail can be automatically generated for pdf or image documents using the main file of a document. It's also possible to contribute a thumbnail adapter for a specific document type or Facet. This is useful mainly for files other than pictures.

The generation of these thumbs is asynchronous, thanks to the use of workers. There is also a default placeholder showed when the generation isn't finished, or if Nuxeo could not generate the thumb.



Back Navigation

We've enhanced the back navigation of your browser using HTML5 pushState.

Permlink available for any Asset

Every Asset now has an accessible and sharable permlink for you to share.

Select2 Integration

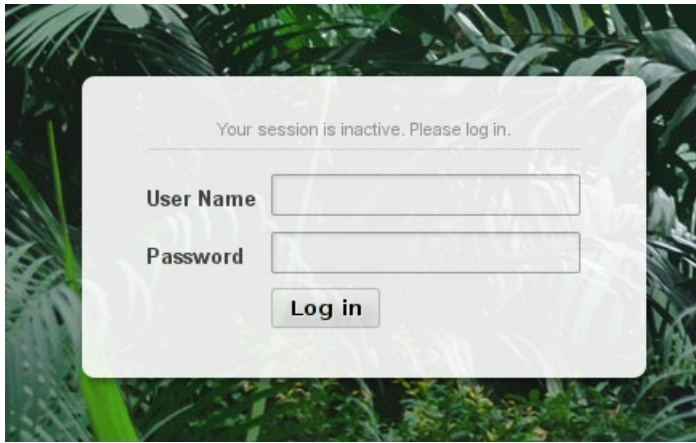
We have replaced the old selection/suggestion JSF widgets by [select2](#) based widgets. Not only does it look better but it's also easier to customize. You can easily setup a user suggestion widget that filter search query with a group parameter.

Nature	<input type="text" value="Select a value"/>
Subjects	<div><div>✖ Architecture a</div><div>Art</div><div>Art history</div><div>Science</div><div>Astronomy</div><div>Technology</div><div>Astronautic</div></div>
Rights	
Source	
Coverage	
Created at	9/23/2013 6:59 PM
Last modified at	9/23/2013 6:59 PM

We also introduced new suggestion widgets on directories. They support n-level chained vocabularies out of the box.

SessionTimeout Feedback

Users see a *'session expired'* message when their session expires.



Form Submit Shield

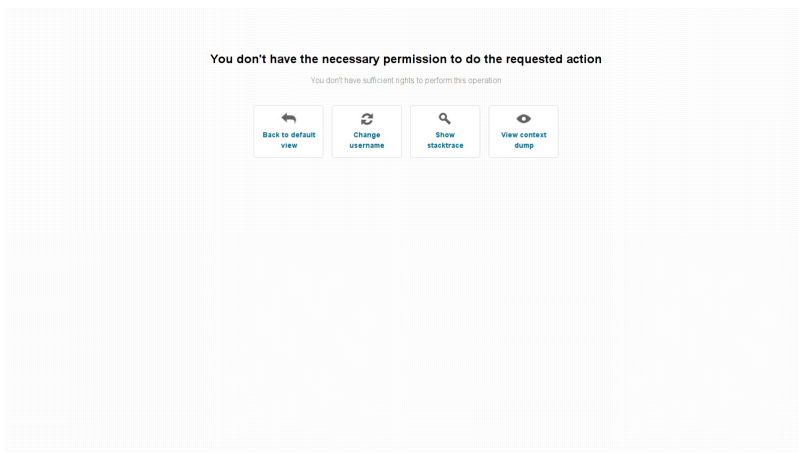
This will protect the heavy handed crazy clickers to have errors because they have clicked on the same button fifty times in two seconds. The user will receive a feedback message telling them the request is being processed. It's better for his mouse and the server. You can deactivate it through the `nuxeo.jsf.enableDoubleClickShield` property in `nuxeo.conf`.

Tag to Version

When a version of a document is created, the tags on the master document are copied to the new version.

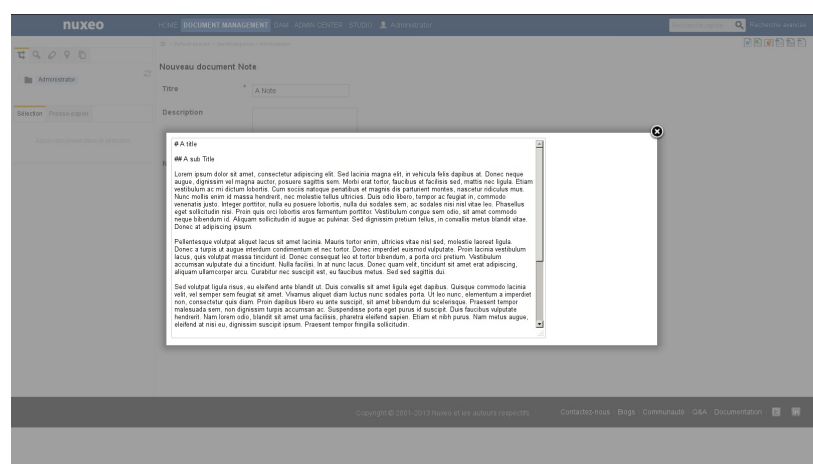
Redesigned Error Page

We have redesigned the legacy error page to have something cleaner, with a user-understandable error message.



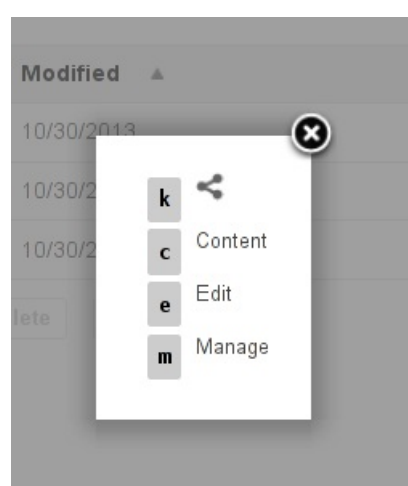
Rich Text Widget

When using the rich editor and text, xml or markdown, we've added a Toggle Fullscreen button that displays the editing window in a fancybox. It makes it easier to see what you're doing.



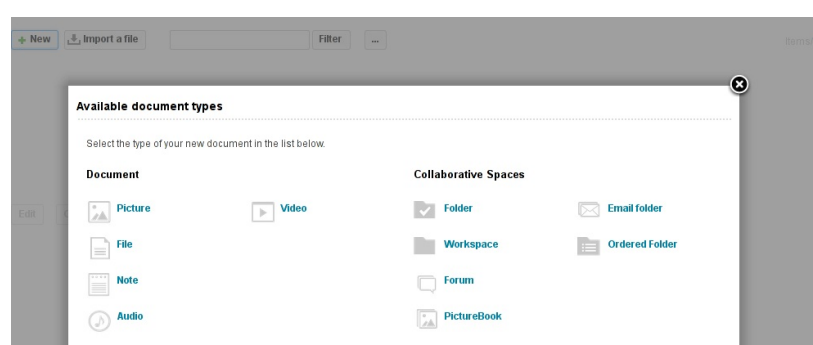
Access Keys

We have added new access key binded to actions. Try maj-h, you'll see the different access keys available. Use ctrl-maj E to go to the Edit tab for example.



New Icons

We have updated all the icons throughout the platform.



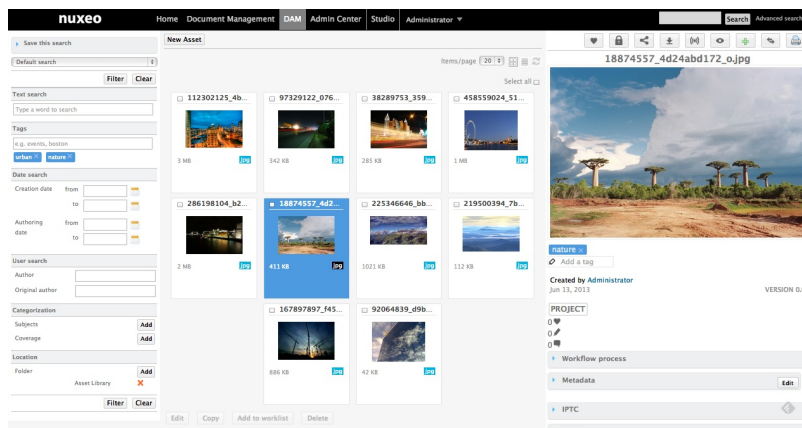
Style Update

Tables, forms and layout have been reviewed and simplified. A focus has been added to simplify the fill of inputs and make the document edition smoother.

Search

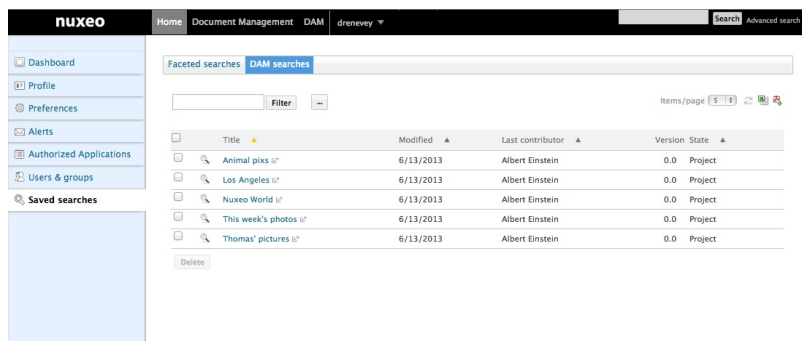
Use Tags in Searches

You can now add tags in your search form. Whether you're using the advanced search form, any faceted search, a content view query or a query operation. It works because we've added tag support to NXQL.



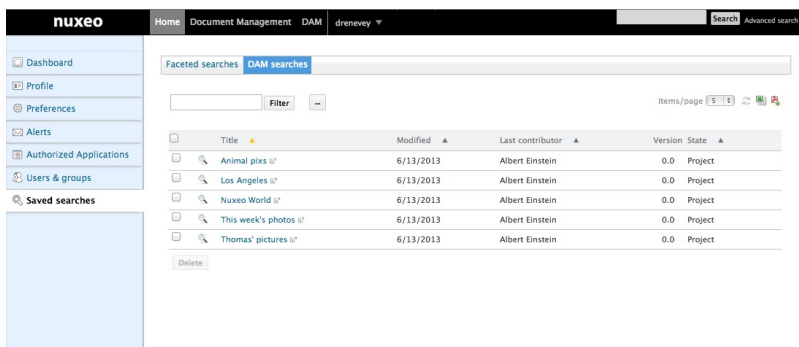
Saved Search Management Tab

Every saved search is listed on a home tab, a subtab for dam, a subtab for faceted search. You can share those saved searches with other users using the manage/rights tab.



Search Panel

- Search criteria can be saved
- Search layout re-organized for better user friendly interface
- Search is triggered when user clicks "Filter" or "Clear"
- Rename "Keywords" with "Text Search"
- Search by location renamed
- New search by tags
- New search by creation date
- New search by author
- New search by original author
- Search by subjects with a new, better, interface
- Search by coverage with a new, better, interface

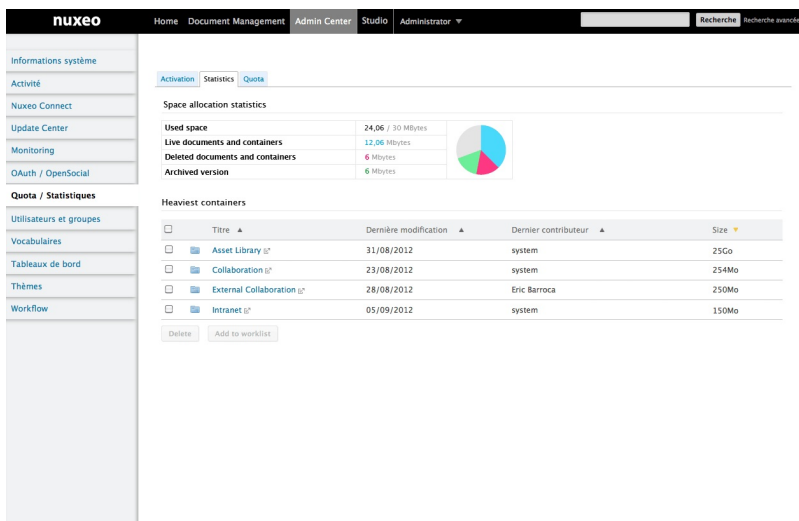


Quota

We've made many improvements in the Quota [Marketplace package](#).

[Global Quotas Activation](#), Quotas can be activated in the admin center

Once it's activated, everything statistical are automatically computed.



Administrators can see (in Manage > Quotas > Stats) the following statistics :

- Total volume used (deleted, live, versions) + "/Available space"
- Volume used by deleted files (trash)
- Volume used by archived versions

A graphical representation of those 3 statistics.

A content view with the title "Sub Folders" with the listing of subfolders

- Columns Title, do:Modified, Last Contributor, and Size
- Buttons Delete and Add to cart (to cut/paste)

Return meaningful error message when encountering a Quota error

User Workspace Quota Management Activation

As the Administrator of the system, you can enable Quota on user workspaces (radio button) with a default value (slider). The value is the same for all the user workspaces.

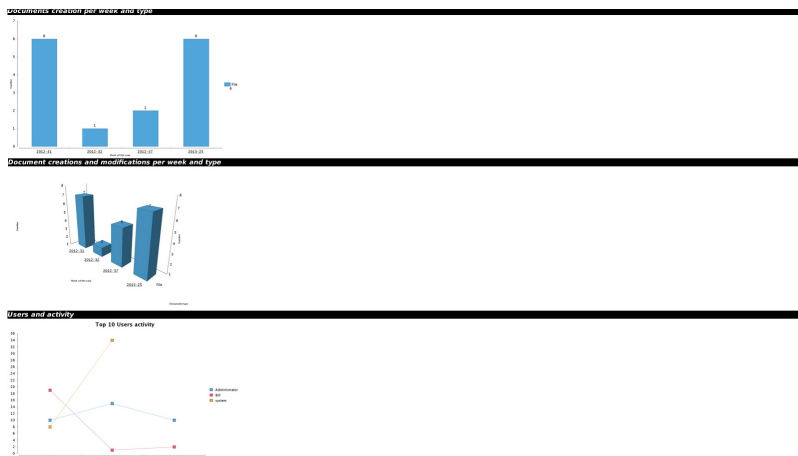
Quota processing now uses its own work queue.

Issues List

BIRT

Marketplace Package

The BIRT Connector has been aligned on BIRT 4.2.2. It means that you can create your report in BIRT Designer 4.2.2. We also made some significant ergonomity enhancements on the admin center UI and we've added parameter validation when generating a report.



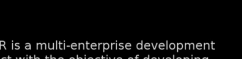
Jenkins Report Module

We have a [new module](#) available in the marketplace. It's an integration between Jenkins and Nuxeo that we use internally. It helps us keep track of our CI, see it's evolution and easily send a report of the state of the CI to our developers. Here's an example of the email we receive:

Hello devs! Here's a status of our CI main jobs on Jenkins. Current number of failing jobs: 3. Current number of unclaimed jobs: 1. Duty comments: To see the complete report click here. Cheers!

Video

The video.js player has been updated to version 4.2.2. This will enable us to benefit from their new plugin architecture. The first plugin we've integrated shows thumbnails of the video while hovering on the progress bar.



SAMAR is a multi-enterprise development project with the objective of developing a platform to manage multimedia content in Arabic.

Created by Administrator

30-Jan-2012

0 0

STATE


Project

No workflow process can be started on this document.

CONTRIBUTORS

Administrator

STORYBOARD



CONTENT

Main File

SAMAR - semantic.mp4 (4 MB)

CONVERSIONS

Format	Conversion...
Ogg 480p	
WebM 480p	
MP4 480p	

Some contributions have been integrated to Nuxeo. Thanks to Patrick Turcotte for the French (Canada) translation, to CapGemini NL for Dutch and to Jordi Mallach for the Catalan translation.

Content Automation

Automation client is fully OSGI, which means you can use it from any OSGI container (Felix, Equinox etc..)

You can fetch dirty properties of a modified document to send only modified properties with automation client

JSON Marshalling enhancement. You can send POJO in input or parameters.

We made a lot of progress on the Content Automation API.

It's possible to contribute a chain into another chain -> Composite Automation Chain.

It's possible to add parameters to a chain.

Chains contributed are displayed into the Automation documentation with operations listing, parameters and input/output.

An Automation Client Test Suite in Java (or Java Test Compatibility Kit -> TCK) is provided for helping people understanding how Automation is working remotely. You can read our [documentation](#) for the details.

[Managing business objects](#) (Plain Old Java Object client side for mapping Nuxeo Document Model Adapter server side) is now available.

The goal is to manipulate business object and avoid direct DocumentModel manipulation on client side:

```
pull : BusinessObject (POJO) <----JSON---- DocumentModelAdapter <---- DocumentModel
push : BusinessObject (POJO) -----JSON----> DocumentModelAdapter -----> DocumentModel
```

Trace

You can now enable [tracing](#) of Content Automation call. Operation chains and operations calls information are collected and logged during their execution by the Automation Trace feature.

Error management

Automation [exception chain](#) can be added to be executed when an error occurs during an Automation execution. After contributing your custom chains, you can declare your exception chains:

```
<extension point="chainException"
target="org.nuxeo.ecm.core.operation.OperationServiceComponent">

  <catchChain id="catchChainA" onChainId="contributedchain">
    <run chainId="chainExceptionA" priority="0" rollBack="true" filterId="filterA"/>
    <run chainId="chainExceptionA" priority="0" rollBack="false" filterId="filterA"/>
    <run chainId="chainExceptionB" priority="10" rollBack="true" filterId="filterB"/>
  </catchChain>

  <catchChain id="catchChainB" onChainId="anothercontributedchain">
    <run chainId="chainExceptionA" rollBack="false"/>
  </catchChain>

</extension>
```

New Description Parameter for Operation

When you write a new operation, you can add a description to your declared parameters. This comes in handy for the API documentation and Nuxeo Studio.

Document Oriented REST API

We have a new [document oriented API](#) for CRUD operation.

Create

```
POST /nuxeo/site/api/v1/id/{idOfTheDoc}
{
  "entity-type": "document",
  "name": "newName",
  "type": "File",
  "properties": {
    "common:icon": "/icons/domain.gif",
    "common:icon-expanded": null,
    "common:size": null
  }
}
```

Read

```
GET /nuxeo/api/v1/path/{pathOfTheDoc}
GET /nuxeo/api/v1/id/{idOfTheDoc}
```

Update

```
PUT /nuxeo/api/v1/id/{idOfTheDoc}
{
  "entity-type": "document",
  "repository": "default",
  "uid": "37b1502b-26ff-430f-9f20-4bd0d803191e",
  "properties": {
    "common:icon": "/icons/domain.gif",
    "common:icon-expanded": null,
    "common:size": null
  }
}
```

Delete

```
DELETE /nuxeo/api/v1/id/{idOfTheDoc}
```

Domain model object and Automation

You can use [Business objects with Automation](#). They are also binded on the REST API and you just have to use the same semantics than for document resources.

Updating a business object

```
PUT /nuxeo/api/v1/path/{pathOfTheDoc}/@bo/BusinessBeanAdapter
{
  "entity-type": "BusinessBeanAdapter"
  "value": {
    id: "37b1502b-26ff-430f-9f20-4bd0d803191e",
    "type": "Domain",
    "title": "Default domain"
    "description": "My new description"
  }
}
```

New Endpoints

Here's the list of endpoints available at the moment:

- Documents (/nuxeo/api/v1/id/{docId}): to do CRUD on documents (including paginated search)
- Users (/nuxeo/api/v1/user/{userId}): to do CRUD on users
- Groups (/nuxeo/api/v1/group/{groupId}): to do CRUD on groups
- Directories (/nuxeo/api/v1/directory/{directoryId}): to do CRUD on directories

- Automation (/nuxeo/api/v1/automation/{Operation id}): to call a "command", i.e. an operation or operation chain deployed on the server

New Adapters

You can also use several adapters for ACLs, blob, business objects, document children, operation, page provider, search etc... An adapter is a URL segment that starts with "@" and that transforms the input resource so as to return another resource. For example, using *@blob* will return the file of a document (the one store on the property given by the next URL segment), and chaining it to *@op* will call an operation (that takes in input a blob):

```
/nuxeo/api/v1/id/{docId}/@blob/file:content/@op/Blob.ToPDF
```

Pluggable Context

Because it is sometimes useful to optimize the number of requests you send to the server, we provide a mechanism to request more information on the answer, simply by specifying the context you want in the request header. You can activate contributor simply by adding the right header on the request. Those contributors can also be activated using action filter.

The document oriented REST API has been moved into a separate package to ease maintenance and upgrades.

We released an [AngularJS sample application](#) to showcase the API.

We've added pagination support for user and group endpoints.

We've added support for different version of the API:

```
/api/v1/id/{docId}...
```

Using the version segment is mandatory. The version is used mostly for giving guarantees to the API user that the JSON entities won't change when migrating to newer version of Nuxeo.

CMIS

We've upgraded OpenCMIS to 0.9.0 . This gives us the new CMIS 1.1 browser bindings but we haven't finished implementing all the new features yet.

New operations

Add Message - [Seam.AddMessage](#)

Add a feedback message to be displayed. The message will be internationalized. You can specify the severity of the message using INFO, WARN and ERROR, INFO being the default. Message parameters are extracted from the context using the 'AddMessage.messageParams'.

Add entry into multi-valued metadata - [AddEntryToMultivaluedProperty](#)

Add value to the field expressed by the xpath parameter. This field must be a multivalued metadata.

Convert to given mime-type - [Blob.Convert](#)

Convert the input file to a file of the given mime-type and return the new file.

Create Asset(s) from file(s) - [Dam.Import](#)

Create Asset(s) from Blob(s) using the FileManagerService.

Create Proxy Live - [CreateProxyLive](#)

This operation will create a proxy that points the given document as input. This is like a symbolic link for File System.

The proxy will be created into the destination specified as parameter.

Creates directory entries - [Directory.CreateEntries](#)

Creates directory entries. Entries are sent as a JSON array. Returns the created entry ids as a JSON array.

Deletes directory entries - [Directory.DeleteEntries](#)

Deletes directory entries. Entry ids to be delete are sent through a JSON array. Returns the deleted entry ids as a JSON array.

Get Live Document - [GetLiveDocument](#)

Get the live document even if this is a Proxy or Version Document.

Get Nuxeo Principal - [NuxeoPrincipal.Get](#)

Retrieve Nuxeo principal and export it as a DocumentModel. If login parameter is not set, the operation will return information about the current user, otherwise Directory Administration rights are required.

QueryAndFetch - [Resultset.PageProvider](#)

Perform a query or a named provider query on the repository. The result is paginated. The result is returned as a RecordSet (QueryAndFetch) rather than as a List of Document. The query result will become the input for the next operation. If no query or provider name is given, a query returning all the documents that the user has access to will be executed.

Raise Seam events - [Seam.RaiseEvents](#)

Raise Seam events without parameters. This is a void operation - the input object is returned back as the output

Reads directory entries - [Directory.ReadEntries](#)

Reads directory entries. Entry ids to be read are sent as a JSON array. Returns the entries as a JSON array of JSON objects containing all fields.

Remove Entry Of Multivalued Property - [RemoveEntryOfMultivaluedProperty](#)

Removes the first entry of the giving value in the multivalued xpath, does nothing if does not exist. If 'is Remove All' is checked, all entry instances in the list will be removed. if not then just the first one found will be removed.

UI Selected documents form list - [Seam.GetDocumentsFromSelectionList](#)

Fetch the documents selected in the selection list passed as a parameter. If the list name is empty, the current folder selection list is used.

Updates directory entries

Category: Services

Operation Id: [Directory.UpdateEntries](#)

Updates directory entries. Entries to be updated are sent as a JSON array. Returns the updated entry ids as a JSON array of JSON objects containing all fields.

Get a Directory Projection - [Directory.Projection](#)

Executes a query using given filter and return only the column *columnName*. The result is assigned to the context variable *variableName*. The filters are specified as key=value pairs separated by a new line. The key used for a filter is the column name of the directory. To specify multi-line values you can use a \\ character followed by a new line.

Example:

```
firstName=John  
lastName=doe
```

By default, the search filters use exact match. You can do a fulltext search on some specific columns using the `fulltextFields`. It's specified as comma separated `columnName`, for instance:

Example:

```
firstName,lastName
```

Traces.Get - [Traces.Get](#)

Retrieve trace associated to a Chain or an Operation.

Traces.ToggleRecording - [Traces.ToggleRecording](#)

Toggle Automation call tracing (you can set the 'enableTrace' parameter if you want to explicitly set the traceEnable value).

Remove Permission - [Document.RemovePermission](#)

Remove permissions for a given user on the input document(s). Returns the document(s).

Add Permission - [Document.AddPermission](#)

Add Permission on the input document(s). Returns the document(s).

Concatenate PDFs - [Blob.ConcatenatePDFs](#)

Given a File document holding a PDF on the `file:content` property and 2 PDFs on the `files:files` property, the following operation will provide a PDF that is the result of the merge of all the PDFs, with the content of the one in `file:content` property first.

Run For Each in new TX - [Context.RunOperationOnListInNewTx](#)

Run an operation in a new Transaction for each element from the list defined by the 'list' parameter. The 'list' parameter is pointing to context variable that represent the list which will be iterated. The 'item' parameter represent the name of the context variable which will point to the current element in the list at each iteration. You can use the 'isolate' parameter to specify whether or not the evaluation context is the same as the parent context or a copy of it. If the isolate is 'true' then a copy of the current context is used and so that modifications in this context will not affect the parent context. Any input is accepted. The input is returned back as output when operation terminate.

CSS 3 Support in Theme Engine

The [flute CSS parser](#) has now been replaced by [phloc-css](#), allowing CSS3 to be used in the Theme engine.

UI Style Guide

The [Nuxeo UI Style Guide](#) is an online help to design your content, explore all the available CSS classes for the different elements, and to check the list of icons embedded in the platform.

UI Framework

Widget Label properties now accept an EL expression. This allows you to have [dynamic labels](#).

Every richfaces ajax call use a [global default ajax event queue](#). You can of course still specify your own queue. This

results in more stable pages.

HTML5 Placeholder

You can use the HTML5 placeholder attribute on inputtext and textarea. It's available as a widget property. You can put text(that will be translated) or an EL.

Text search

Tags

New drop zone Widget

[This widget](#) allows you to execute an operation with the files you've dropped as input.

Widget Actions

One can use automatic JS confirmation popup. If you add a confirmMessage property in the actions, it generates the appropriate js method. It accepts any EL as parameter. Here's an example:

```
<action id="JenkinsReportSendMail">
  <properties>
    <property name="confirmMessage">label.jenkins.sendMail.confirm</property>
    <propertyList name="confirmMessageArgs">
      <value>#{docSuggestionActions.getDocumentWithId(currentSuperSpace.id)
        .getPropertyValue('jenkinsreports:report_email')}</value>
    </propertyList>
  </properties>
</action>
messages:
label.jenkins.sendMail.confirm=This will send an email to {0},
are you sure that you would like to continue?
```

You can now include widgets that render actions. This is particularly useful in Nuxeo Studio. This allows us to easily blend different action types like a submit button, an outgoing link and an action that [opens a fancybox](#). You can also [merge all of them](#) in any order you want.

We have a better support of [form inside fancybox](#) opened by actions. You can submit the associated form without submitting every other form on the page. No need to do nested form anymore.

You can mix any type of action. If one action does not support ajax (like a download), it can be in the same tool bar as an ajax action.

You can associate selected documents of a content view to actions outside the content view. That's how we made bulk edit.

[We allow resetting of JSF components state when using ajax](#). JSF action listener `#{jsfResetActions.resetComponents}` can be used in actions that need to reset all components in the same JSF naming container.

Action Context

Until now, action filters were using JEXL for resolution instead of the engine used by JSF, and Seam components could

be resolved in this context, but using a hack, hence only actions *starting with the seam component name* could be resolved correctly, for instance `#{clipboardActions.canCopy}`. Others were not resolved correctly, for instance `#{!clipboardActions.canCopy}`, `#{empty clipboardActions.selectedDocuments}`, `#{clipboardActions.canCopy}` and `clipboardActions.canPaste`, etc...

Now the action context has been changed into an interface, and is responsible for evaluating expressions held by filters (instead of the filter itself) => by implementing this interface, *any kind of expression with any kind of variables* in context can be resolved using the filters/actions service API. Seam JSF EL are resolved in a native Seam/JSF context.

Action Widget Types

We've made a pluggable action types library. It means that now you can define your own action type and associated rendering. You can use it within new action widget types (like ToolBar actions, Form actions, Tabs actions etc...).

Runtime Service Seam Injection

You can now inject Nuxeo runtime services directly in a Seam bean. Where you use to call `Framework.getLocalService(SchemaManager.class)` in a method, you can simply inject the service.

```
@In
SchemaManager theSchemaManagerService;
```

Richfaces Update

Nuxeo uses a custom version of Richfaces based on 3.3.1 GA, This custom version of Richfaces is now available on [GitHub](#) with all Nuxeo's patches.

New Header

Nuxeo's header has been rewritten using widgets displaying actions. Basically it means you can override each part of the header easily. Like the links, search box etc...

Ajax Tabs

You can Ajaxify any tab you want using the property `nuxeo.jsf.useAjaxTabs`. It means that when you click on a tab, only its content will be reloaded instead of the whole page. And the URL is updated automatically so you can have permalink with the selected tab, use the back button to go to a previously selected tab. This option can also be chosen when creating a new tab in Nuxeo Studio.

Debugging JavaScript

When you are in dev mode, the JavaScript files are not minified. This is much better when you need to debug JavaScript.

Content Views

On Nuxeo side, filter can be shown unfolded by default, and some content view templates have been redefined to ease the customization.

On Studio side, the "filter unfolded" option is presented for 5.7.2, and events as well as the page provider class and properties can now be configured also.

CORS support

If you do cross-domain requests from any JavaScript client to access WebEngine resources or Automation APIs, there's a chance that your browser forbids it. [CORS](#) allows you to communicate with Nuxeo from another domain using XMLHttpRequests.

We've added [a new extension point](#) based on Vladimir Dzhuvinov's universal CORS filter, which allows you to configure on which URL cross-origin headers are needed.

Scan Importer

The scan importer has been improved. You can choose if you want to create the import document container each time you launch an import. You can also choose to update an existing document instead of creating a new one.

Login Page

The login page can now be configured via an extension point, making it easier to upgrade when the login page is customized, and benefit from new features, like [OpenID authentication](#) provided by Nelson Silva.

Here is a sample configuration for the login page:

```
<extension target="org.nuxeo.ecm.platform.ui.web.auth.service.PluggableAuthenticationService" point="loginScreen">
  <loginScreenConfig>
    <bodyBackgroundStyle>url("${org.nuxeo.ecm.contextPath}/img/my_login_background_img.png") 100px 100px no-repeat black</bodyBackgroundStyle>
    <headerStyle></headerStyle>
    <footerStyle>display: none;</footerStyle>
    <loginBoxBackgroundStyle>url("${org.nuxeo.ecm.contextPath}/img/my_background_img.png") 0 0 no-repeat green</loginBoxBackgroundStyle>
    <loginBoxWidth>100px</loginBoxWidth>
    <logoUrl>${org.nuxeo.ecm.contextPath}/img/my_logo.png</logoUrl>
    <logoAlt>My logo Alt</logoAlt>
    <logoWidth>100px</logoWidth>
    <logoHeight>100px</logoHeight>
    <newsiframeUrl>https://www.nuxeo.com/embedded/dm-login</newsiframeUrl>
  </loginScreenConfig>
</extension>
```

Long Running Listener

You can now extend a new abstract class called `AbstractLongRunningListener`. Its purpose is to help building an Asynchronous listeners that will handle a long running process.

By default, `PostCommitEventListener` is executed in a Worker that will take care of starting/committing the transaction.

If the listener requires a long processing, this will create long transactions which should be avoided. To avoid this behavior, this base class split the processing in 3 steps:

- Pre processing: transactional first step
- Long running: long running processing that should not require transactional resources
- Post processing: transactional final step

To manage sharing between the 3 steps, a simple Map is provided.

Move To Tomcat 7

Tomcat has been upgraded to version 7.0.42. The main improvement is the upgrade to the Servlet 3.0 API. Take a look at the [details](#).

Nuxeo Visible Content Store (VCS)

Issues List

Enhancements

[Single datasource mode](#), removes the extra cost of XA transactions when all datasources points to the same database (which is the most common setup). It simplifies configuration and enhances performance.

[Soft delete](#), more efficient deletion using delayed batching.

Optimized id generation, using database-native sequences ([PostgreSQL](#), [SQL Server](#)) or UUIDs ([PostgreSQL](#)). This

improves performance and database size.

[Flag to disable proxies](#), if you don't use proxies, then this removes the extra work at the SQL level and improves performance.

[Allow attaching schemas to proxies](#)

Database Support

[Microsoft SQL Server 2012 support](#), including Azure.

[PostgreSQL 9.2 support](#) for cluster mode (non-clustered mode already worked in Nuxeo 5.6).

[IBM DB2 9.5 preliminary support](#) (non-optimized, therefore only for small databases).

Misc

Various performance enhancements for PostgreSQL, Oracle and SQL Server

[Added "NotFulltextIndexable" facet](#) to avoid indexing documents with this facet.

[Allow clustering](#) for Oracle on Amazon RDS.

[Optimize complex list property update](#) (use UPDATE instead of DELETE + INSERT).

[Support Microsoft's JDBC driver for SQL Server.](#)

Redis

Several components in Nuxeo will make use of Redis as an optional feature. To configure Redis, the following extension point can be used:

```
<extension target="org.nuxeo.ecm.core.redis.RedisService" point="configuration">
  <redis disabled="false">
    <prefix>nuxeo:work:</prefix>
    <host>localhost</host>
    <port>6379</port>
    <password>secret</password>
    <database>0</database>
    <timeout>2000</timeout>
  </redis>
</extension>
```

Redis Service

To use the Redis service, do:

```
RedisService redisService = Framework.getLocalService(RedisService.class);
```

This service currently provides two methods:

```
redisService.getJedisPool();
redisService.getPrefix(); // should be used as prefix for any key
```

Standard usage pattern is:

```
protected Jedis getJedis() {
    return redisService.getJedisPool().getResource();
}

protected void closeJedis(Jedis jedis) {
```

```

    redisService.getJedisPool().returnResource(jedis);
}

protected void myMethod() throws IOException {
    Jedis jedis = getJedis();
    try {
        // ... use jedis here ...
        // ... all keys should be prefixed with redisService.getPrefix()
    } finally {
        closeJedis(jedis);
    }
}
}

```

Redis WorkManager

We added a Redis based WorkManager. This is helpful for:

- High-Activity work queueing, to avoid saturating memory or blocking,
- Queue persistence, to avoid lost jobs on shutdown,
- New feature: node-aware job execution.

PDF Signing

Now you can apply more than one signature to a PDF and configure the position of the signatures following a predefined matrix. Here's a sample contribution:

```

<extension target="org.nuxeo.ecm.platform.signature.api.sign.SignatureService"
point="signature">
  <configuration>
    <reason>This document signed as an example.
  </reason>
    <layout lines="2" columns="5" startLine="2" startColumn="1" textSize="10"/>
  </configuration>
</extension>

```

Mockito Integration

We've added Mockito support to our test framework. Here's an example:

```

@RunWith(FeaturesRunner.class)
@Features(RuntimeFeature.class)
public class CanMockServicesTest {

    @RuntimeService
    @Mock
    AFakeService myService;

    @Before
    public void doBefore() {
        when(myService.getSomething()).thenReturn("Hello !");
    }

    @Test
    public void itShouldBindMocktoAService() throws Exception {
        AFakeService service = Framework.getService(AFakeService.class);
        assertNotNull(service);
        assertEquals("Hello !", service.getSomething());
    }
    ...
}






















```

Localization

Crowdin

Nuxeo has an account on [Crowdin](#). It makes it easier for everyone to help translate Nuxeo in different languages. You can read [this blog](#) for more information.

Needs Translation:

 Arabic translated: 73%	 Basque translated: 29%	 Catalan translated: 69%	 Chinese Simplified translated: 42%	 Czech translated: 70%	 Dutch translated: 92%
 French translated: 88%	 French, Canada translated: 0%	 Galician translated: 68%	 German translated: 83%	 Greek translated: 39%	 Italian translated: 64%
 Japanese translated: 73%	 Macedonian translated: 0%	 Polish translated: 67%	 Portuguese translated: 70%	 Portuguese, Brazilian translated: 32%	 Russian translated: 72%
 Serbian (Cyrillic) translated: 0%	 Spanish translated: 66%	 Vietnamese translated: 43%			

Dutch Translation

Our partner Capgemini Netherlands has contributed the Dutch translation for Nuxeo.

Miscellaneous

Simple Computed Group

The idea is to simplify computed groups management by a simple contribution. Here are two examples. The first one will create a virtual group *grade_companyValue* where *companyValue* is the company value for the user. The second one will create a *creator_documentId* for each document where the user is the creator.

```
<extension point="computer" target="org.nuxeo.ecm.platform.computedgroups.ComputedGroupsServiceImpl">
  <!-- userAttribute -->
  <userMetadataGroupComputer xpath="company" groupPattern="grade_%s" name="grade_cg"/>
  <!-- documentMetadata -->
  <documentMetadataGroupComuter xpath="dc:title" whereClause="dc:creator = '%s'" groupPattern="creator_%s" name="creator_" />
</extension>
```

ScanImporter Enhancement

We're still improving the scan importer. One of the latest enhancement is the support of [MVEL](#) in the XML mapping code. This will allow you to setup some business logic during import without writing custom Java code. Here's an example:

```
<extension target="org.nuxeo.ecm.platform.importer.xml.parser.XMLImporterComponent"
  point="documentMapping">

  <docConfig tagName="html">
    <docType>Instruction</docType>
    <parent><![CDATA[ #{
      nodes = xml.selectNodes('//meta[@name=\'RCDirection\'']/@content');
      String parent = nodes.get(0).getText();
      return Fn.mkdir(root, '/', parent, 'StructureFolder');
    }]]></parent>
    <name><![CDATA[ #{
      String valueFound = xml.selectNodes("//meta[@name='RCIdentifiant']/@content")[0].getText();
      String name = valueFound.replace(' ', '').replace('/', '-');
      return name;
    }]]></name>
    <postCreationAutomationChain>testBJA</postCreationAutomationChain>
  </docConfig>

</extension>
```

With the following xml fragment:

```
<html>
  <head>
    <meta name="RCDirection" content="Dir1/Sec1.1" />
    <meta name="RCIdentifiant" content="DGAL/C98 - 8010" />
  </head>
  ...
</html>
```

Will be equivalent to the following code snippet:

```
// This computation is because the parent evaluation with the Fn.mkdir part.  
// This happens if the Dir1/Sec1.1 documents doesn't exist in root  
// => see Fn.mkdir description above  
String path = root.pathAsString;  
DocumentModel doc = session.createDocumentModel(path, "Dir1", "StructureFolder");  
doc = session.createDocument(doc);  
  
path = doc.getPathAsString();  
doc = session.createDocumentModel(path, "Sec1.1", "StructureFolder");  
doc = session.createDocument(doc);  
  
// Here is because the document creation activation  
// Parent MVEL expression return the result of the mkdir  
path = doc.getPathAsString();  
// node selected into the xml is "DGAL/C98 - 8010"  
// MVEL expression remove space and replace slash by minus  
String name = "DGA-C98-8010";  
doc = session.createDocumentModel(path, name, "Instruction");
```

NXQL

See the [NXQL Documentation](#) for the current state of NXQL.

ecm:isCheckedIn: 1 if the document is checked in and 0 if not (the opposite of DocumentModel.isCheckedOut()). This can only be compared to 1 or 0.

ecm:isVersion or ecm:isCheckedInVersion: 1 for versions and 0 for non-version (DocumentModel.isVersion()). This can only be compared to 1 or 0. (The name ecm:isVersion is available since Nuxeo 5.7.3)

ecm:versionVersionableId: the id of the versionable document of a version (the versionable document is the one from which the version was created).

ecm:isLatestVersion: 1 if this is the latest version of a document, 0 if not. This can only be compared to 1 or 0.

ecm:isLatestMajorVersion: 1 if this is the latest major version of a document, 0 if not. This can only be compared to 1 or 0.

ecm:tag: allows NXQL queries on a document's tags.

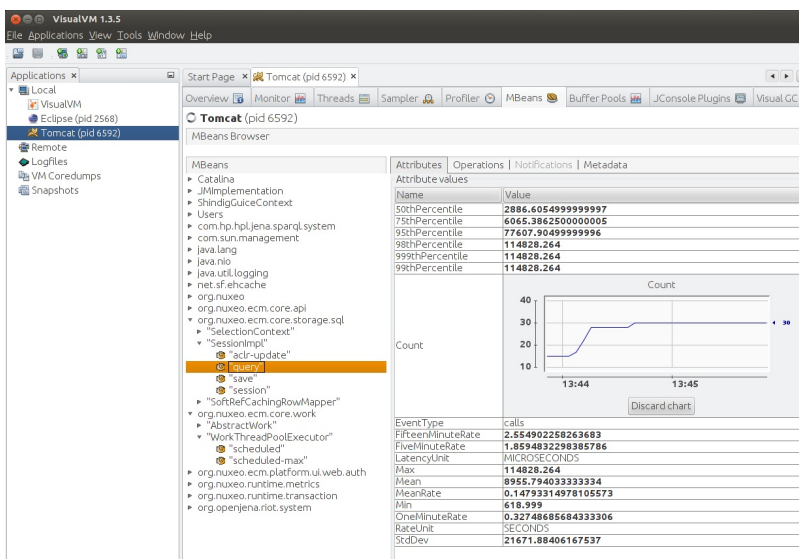
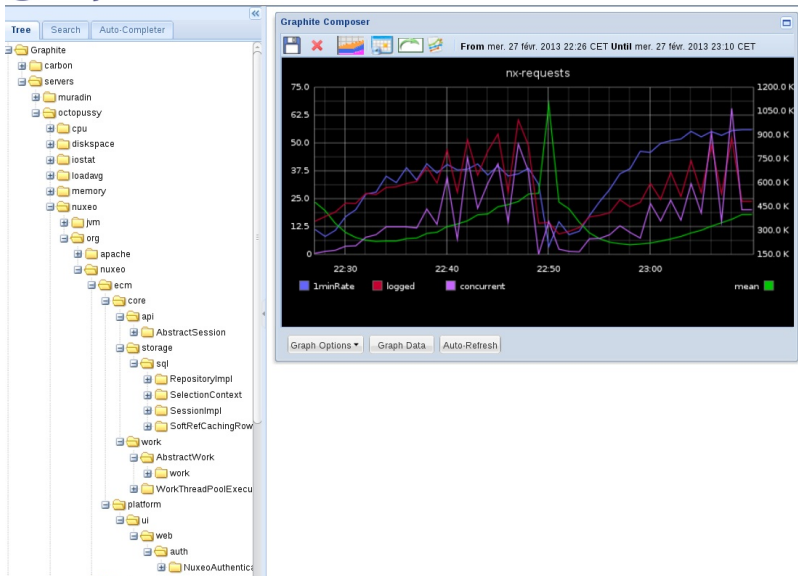
ecm:proxyTargetId: allows NXQL queries on the id of the target of a proxy.

ecm:proxyVersionableId: allows NXQL queries on the id of the versionable document of a proxy (the versionable document is the one from which the version to which the proxy is pointing was created).

Monitoring

Use [Coda Hale Yammer Metrics](#) to instrument Nuxeo, providing lots of internal metrics, accessible from JMX or Graphite. Read more in our [documentation](#).

graphite



Multi-Tabs Navigation

Nuxeo now handles multiple-tabs navigation without having a "cannot restore view" error. A new StateManager has been introduced in order to store ViewStates per conversation. In other words, when a user navigates to a new tab, a new conversation is started and its view states are saved separately. If the user comes back to an old tab, he will not have "view expired" exceptions anymore.

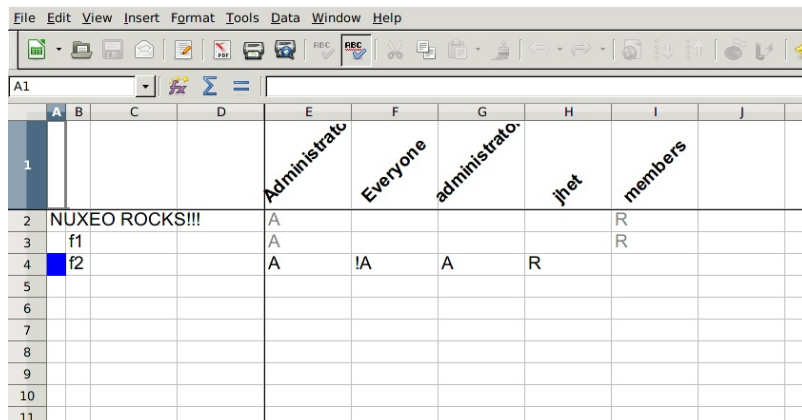
- By default, 4 conversation states are saved following a LRU policy (Least Recently Used).
- Each conversation state still has max 4 saved view states.
- This means that potentially, a maximum of 16 view states are saved in session.

You can adapt these settings using the appropriate parameters in `nuxeo.conf`.

Right Audit

This [Marketplace package](#) allows to run an audit on all permissions for all users of a document tree. The result is an XLS matrix for every exported document showing permission for each user, including blocking permissions if any.

You can also do an excel export of all the users and groups.



	A	B	C	D	E	F	G	H	I	J
1					Administrato	Everyone	administrato	jhet	members	
2		NUXEO ROCKS!!!			A				R	
3		f1			A				R	
4		f2			A	IA	A	R		
5										
6										
7										
8										
9										
10										
11										

Kerberos Login Module

A [Kerberos module](#) has been contributed by OpenWide, allowing authentication throught the Kerberos protocol.

You can also [login to Nuxeo using your OpenID](#), thanks to Nelson Silva.

MarketPlace/Update Center

nuxectl commands

mp-show: Show Marketplace package(s) information. You must provide the package file(s), name(s) or ID(s) as parameter.

Blogs and Websites Packaging

Blogs and Website have been removed from the DM module and are now independant packages.

Database Support

We have updated the [MS SQL Server documentation](#) with new recommendations.

PostgreSQL 9.2-1002 JDBC4 driver supports simple connection fail-over useful for high-availability configurations.

PostgreSQL 9.3 is officialy supported.

Nuxeo.conf New Parameters

Parameter	Default Value	Description
nuxeo.jsf.numberOfConversationsInSession	4	Since 5.7.2, Parameter to control the number of conversation states that are saved in session. Each conversation holds a number of view states that is defined by nuxeo.jsf.numberOfViewsInSession
nuxeo.jsf.numberOfViewsInSession	4	Since 5.7.2 (5.6-HF20) Parameter to control the JSF init parameter com.sun.faces.numberOfViewsInSession
nuxeo.jsf.numberOfLogic	4	Since 5.7.2 (5.6-HF20) Parameter to control the JSF init parameter com.sun.faces.numberOfLogicalViews

alViews		
nuxeo.notification.eMailSigner	The Nuxeo team	Signer of the sent e-mail.
org.nuxeo.automation.trace	false	<p>Since Nuxeo 5.7.3, you can enable this mode if you'd like to display automation traces during runtime:</p> <ul style="list-style-type: none"> You'll benefit from exhaustive logs to debug all automation chain and/or operation execution. The automation trace mode is disabled by default (not suitable for production). It can be activated through JMX via org.nuxeo:TracerFactory MBean during runtime.
org.nuxeo.automation.trace.printable	*	<p>Since Nuxeo 5.7.3, by default, all automation executions are 'printable' (appear in logs) when automation trace mode is on.</p> <ul style="list-style-type: none"> You can filter chain and/or operation execution trace printing by setting this property to chain name and/or operation separated by comma. Comment this property to get all automation chains/operations back in printing (by default set to * (star)
nuxeo.jsf.enableDoubleClickShield	false (unset)	Enable a shield on forms to prevent users from submitting twice the same form (accidental double-click)
org.nuxeo.dnd.upload.timeout	30000	Maximum time for uploading a file via Drag & Drop to the server.
org.nuxeo.dnd.exec.timeout	30000	Maximum time for executing import of files uploaded via Drag & Drop
org.nuxeo.dnd.extended.mode.timeout	2000	Mouseover time to switch to extended mode UI (setting to -1 disables the extended mode)
nuxeo.jsf.useAjaxTabs	false (unset)	Enable ajaxified tabs on document views.
org.nuxeo.prod	false	Setting this property to "true" will display a quite visible warning message in the Admin Center, stating that this is a production instance. This is useful for administrators who are sometimes confusing their Nuxeo production server with their test server (not to rat anyone out).
nuxeo.vcs.blocking-timeout-millis	100	Database maximum wait time to get a connection from the pool when all connections are in use, for Nuxeo repository (VCS).
nuxeo.vcs.idle-timeout-minutes	10	Database timeout after which connections not in use are removed from the pool, for Nuxeo repository (VCS).
nuxeo.vcs.fulltext.disabled	false	
nuxeo.vcs.noddl	false	
nuxeo.vcs.idtype	varchar	