

Nuxeo Server 10.3 Fast Track Release Notes

VERSION: FT

Updated: December 14, 2018

This page relates to the release notes of Nuxeo Server and related addons for the 10.10 cycle, a.k.a LTS 2018 cycle. It will list the improvements and features that are successively shipped with the 10.1, 10.2, 10.3 and LTS 2018 releases. Evolutions are grouped by components.

You can also find detailed JIRA release notes:

- [10.1 JIRA release notes](#)
- [10.2 JIRA release notes](#)
- [10.3 JIRA release notes](#)

We also provide [instructions for upgrading](#) to the latest release.

Nuxeo Server

Runtime

New Descriptor Interface

A Descriptor interface has been added, such that all descriptors implementing it will be automatically registered and unregistered in the targeted component, saving lots of boilerplate code and pushing towards a standardization of the contribution registration behaviours. As an example, here is the AvroComponent written [before](#) and [after](#). 214 locs became 131, 4 inner classes removed, 2 full methods removed, 83 less useless locs, and a standardized and reliable code.

More on JIRA ticket [NXP-25186](#)

Duration type in descriptors

You can now use Duration type into your descriptors. Nuxeo handles the JDK Duration format (PT10M, PT200S, PT0.500S) or a specific format for days, hours, minutes, seconds and milliseconds (10m, 200s, 500ms).

More on JIRA ticket [NXP-25402](#)

Core Repository

Trash, Untrash and EmptyTrash Operations

Two new

operations [TrashDocument](#) and [UntrashDocument](#) have been added.

More on JIRA ticket [NXP-24282](#) and [NXP-24281](#).

New Trash Service Enabled

The new trash service has been enabled by default on the repository. It uses a system property `ecm:isTrashed` for labelling a document as being trashed. It also fires dedicated events `documentTrashed` and `documentUntrashed` (hold by TrashService interface). A migrator has been implemented for migrating content from the old trash system (relying on lifecycle state) to the new one (relying on the system property `ecm:isTrashed`). See upgrade notes for more information on migration. Further more, the implementation now makes use of the Bulk Action Framework for more resilience and ability to send to trash millions of documents at once.

More on JIRA tickets [NXP-24850](#) and [NXP-24035](#) and [NXP-25259](#)

ecm:isTrashed in the JSON

`ecm:isTrashed` is now in the JSON representation of a Nuxeo Document

More on JIRA ticket [NXP-24741](#)

New firstAccessibleAncestor REST API Enricher

It is now possible to get the closest document's ancestor of a document using the `firstAccessibleAncestor` JSON Enricher.

More on JIRA ticket [NXP-24282](#)

New hasContent Enricher

The enricher hasContent adds a boolean property "hasContent" so as to let the client know if there are children for the given node saving one call when trying to build hierarchical navigation.

More on JIRA ticket [NXP-24298](#)

No Mention of the Repository on a Document Reference

When referencing a document in a property, we don't need anymore to store the repository id. If the repository id is not there, the same as the referencing document is chosen. The following

two `documentResolver` restrictions with `idOnly` and `pathOnly` can be used for this:

```
<xs:restriction base="xs:string"
ref:resolver="documentResolver" ref:store="idOnly" />
```

```
<xs:restriction base="xs:string"
ref:resolver="documentResolver"
ref:store="pathOnly" />
```

Their semantics is to store only the id or only the path, without any prefixed repository. When fetching the constraint, the document of the given id or path is resolved in the same repository as the current document.

More on JIRA ticket [NXP-22450](#).

GetProxies Method on CoreSession

`getProxies(DocumentRef)` has been added to the CoreSession to efficiently get proxies, without having to write and run a query.

More on JIRA ticket [NXP-24922](#).

KeyValueStoreUIDSequencer

A new `KeyValueStoreUIDSequencer` is available, to store sequences in a key/value store. The store is the same for all sequencers, but they are using different keys, prefixed by the sequencer name.

It can be used by doing for example:

```
<extension
target="org.nuxeo.ecm.core.uidgen.UIDGeneratorService
" point="sequencers">
  <sequencer name="uidgen"
class="org.nuxeo.ecm.core.uidgen.KeyValueStoreUIDSequencer" default="true" />
</extension>
```

```
<extension
target="org.nuxeo.runtime.ConfigurationService"
point="configuration">
  <!-- this is an example, "sequence" is already the
default -->
  <property
name="nuxeo.uidseq.keyvaluestore.name">sequence</
property>
</extension>
```

Assuming the following configuration to define a suitable key/value store, for example:

```
<extension
target="org.nuxeo.runtime.kv.KeyValueService"
point="configuration">
  <store name="sequence"
class="org.nuxeo.ecm.core.mongodb.kv.MongoDBKeyValues
tore">
    <property name="collection">sequence</property>
  </store>
</extension>
```

More on JIRA ticket [NXP-23744](#).

Core Storage

Compatibility with MongoDB 4.X

Nuxeo DBS MongoDB implementation is now compatible and continuously tested with MongoDB 4.0.

More on JIRA ticket [NXP-25620](#).

MongoDB Client TrustStore for in-flight encryption

The following `nuxeo.conf` properties can be set to define appropriate TLS/SSL configuration for

```
MongoDB: nuxeo.mongodb.ssl=true
nuxeo.mongodb.truststore.path
nuxeo.mongodb.truststore.password
nuxeo.mongodb.truststore.type
nuxeo.mongodb.keystore.path
nuxeo.mongodb.keystore.password
nuxeo.mongodb.keystore.type
```

More on JIRA ticket [NXP-26072](#).

More Secured AES Binary manager

The encryption used to be based on [AES/CBC/PKCS5Padding](#) which has been found to be insecure (susceptible to padding Oracle attacks). Now a more secure encryption algorithm, [AES/GCM/NoPadding](#), is used.

More on JIRA ticket [NXP-25540](#).

KMS Keys Support on S3 Binary Store

The support for [KMS keys](#) for S3 Server-Side Encryption is added.

More on JIRA ticket [NXP-22949](#).

Compatibility of the S3 Blob Provider with Dell Ecs and "Pathstyleaccess" S3 Option

Some binary managers S3 compatible require a specific URL which format is now handled by our S3 Blob Provider.

More on JIRA ticket [NXP-25525](#).

MySQL fulltext table with InnoDB

Fulltext tables stored in MySQL/MariaDB now use InnoDB engine instead of MyISAM. InnoDB engine is more performant and allows to add foreign constraints on table's properties. You can change your current engine by following this [documentation](#)

More on JIRA ticket [NXP-17479](#).

Directory

Improved Query API for Directories

New directory query APIs using a QueryBuilder:

```
Session.query(QueryBuilder, fetchReferences)
```

```
Session.queryIds(QueryBuilder)
```

It is now possible to use predicate style queries for fetching directory entries.

More on JIRA ticket [NXP-19262](#)

Tenant-isolated directories with MongoDB

Multi-tenant addon now supports tenant-isolated directories with MongoDB

More on JIRA ticket [NXP-22682](#)

Same Directory Entry ID on Different Tenants

Unicity check on directory entry has been moved post tenant-specific computation, so that same end user id can be used in two different tenants.

More on JIRA ticket [NXP-25264](#)

Ldap Connection Timeout Reduced to 1 Min

The pooling connection timeout of the LDAP connections has been reduced from 30 min to 1 min to avoid some performance issues when making a lot of connections with multiple users.

More on JIRA ticket [NXP-25085](#)

Filters parameter on Directory.SuggestEntries

It is now possible to filter directories values on a given column value, so as to implement chain select behaviors for instance. You can use the "filters" parameter, with a serialized property,

like: `{"parent": "europe"}`

More on JIRA ticket [NXP-25299](#)

Workflow

More Properties on the Task Object

When using the REST API, the JSON structure of a Task object now also includes:

- the workflow initiator
- the workflow title
- the workflow life cycle state
- the graph route URL

More on JIRA ticket [NXP-24476](#).

Nuxeo Streams

Avro serialization for Nuxeo Log/Stream

Avro (<http://avro.apache.org>) has been integrated to the platform: Avro schema store, Nuxeo Document <> Avro format converter so as to be able to use this format for Nuxeo Stream messages. It makes communication with outer world easier (no Nuxeo dependency in the message), makes messages more compact and give backward / forward compatibility in messages format. As a consequence, Nuxeo Stream can now encode record with different codec:

- `legacy`: the original format based on java Externalizable
- `avro`: avro message with a schema fingerprint header (Nuxeo has an avro SchemaStore service to retrieve schemas).
- `avroBinary`: avro message without schema header so more compact
- `avroJson`: avro in JSON for debug purpose only

You can choose the encoding for the different service using `nuxeo.conf` options:

```
nuxeo.stream.work.log.codec=legacy
```

```
nuxeo.stream.audit.log.codec=legacy
```

```
nuxeo.stream.pubsub.log.codec=avroBinary
```

Note that you should not change the codec of an existing stream (Kafka Topic or Chronicle file), this should be done only on new stream.

More on JIRA ticket [NXP-22597](#) and [NXP-24324](#)

Avro Confluent Support

You can now use Avro Confluent format which is slightly different than the Avro format. It allows to use tools from Confluence, and to query the messages using the [KSQL](#).

More on JIRA ticket [NXP-25538](#).

Kafka 2.0

The Nuxeo Platform now relies on Kafka 2.0

More on JIRA ticket [NXP-25600](#).

SASL and TLS Authentication Against Kafka

```
# SASL
```

```
kafka.sasl.enabled=false
```

```
kafka.security.protocol=SASL_PLAINTEXT
```

```
kafka.sasl.mechanism=SCRAM-SHA-256
```

```
kafka.sasl.jaas.config=org.apache.kafka.common.security.scram.ScramLoginModule required
```

```
username="kafkaclient1" password="kafkaclient1-secret";
```

SSL

```
kafka.ssl=false  
kafka.truststore.type=JKS  
kafka.truststore.path=  
kafka.truststore.password=  
kafka.keystore.type=JKS  
kafka.keystore.path=  
kafka.keystore.password=
```

More on JIRA ticket [NXP-25956](#).

Stream Computations Use Watermark

Computations that re-append records in other streams now add a watermark (time and flag information) in the record. That allows always benefiting from advanced features of the streams such as latency computation.

More on JIRA ticket [NXP-24641](#)

Using The Key as the Kafka Record Key

When using the Kafka implementation of Nuxeo Stream, the key in the log is now the Kafka key itself, not the hash of it as it was previously (only option for the other backend, Chronicle queue).

More on JIRA ticket [NXP-24640](#).

WorkManager

Error Event After Successive Failures on a Work

An event `workFailed` is now fired when a work fails several times.

More on JIRA ticket [NXP-24126](#).

State Information for the StreamWorkManager

By using `nuxeo.stream.work.storestate.enabled=true`, it is now possible to get information of a given work when using the StreamWorkManager.

More on JIRA ticket [NXP-24397](#).

Ability to Cancel a Long-running Work with the StreamWorkManager

StreamWorkManager is now able to cancel a long-running work, provided the work calls regularly `isSuspending`.

More on JIRA ticket [NXP-24400](#).

Scheduler

Customisable Job

It is now possible to specify what Job should be created when using the scheduler thanks to a new attribute `jobFactoryClass`:

```
<schedule id="testSchedulerSingleExecution"
```

```
jobFactoryClass="org.nuxeo.ecm.core.scheduler.SingleE
xecutionEventJobFactory">
  <event>testSchedulerSingleExecution</event>
  <cronExpression>* * * * * ?</cronExpression>
</schedule>
```

More on JIRA ticket [NXP-24540](#).

Timezone Sensitive Cron Service

The scheduler service (Cron) now allows using a timezone on expressions to avoid relying on the time of the server for firing events.

More on JIRA ticket [NXP-24955](#).

PubSub

PubSub Service on Nuxeo Stream

An implementation of the PubSub service has been provided using Nuxeo Stream. This allows to not rely on Redis for this service that is notably used for cache syncing on the repository in a cluster, as well as for acquiring locks on documents. To use it you can apply the following configuration in [nuxeo.conf](#):

```
nuxeo.pubsub.provider=stream
```

More on JIRA ticket [NXP-23799](#).

Audit

Batch Retry

AuditLogWriter uses a computation with a batching and retry mechanism that allows to tolerate up to 2 minutes of unavailability of the audit backend.

More on JIRA ticket [NXP-25341](#).

STARTSWITH Operator Available for All Audit Backend (SQL, MongoDB and Elasticsearch)

Following introduction

of `AuditBackend#queryLogs(AuditQueryBuilder)`, we now have an easy way to query audit. We introduced in 10.1 the STARTSWITH operator, we could use it as below:

```
auditBackend.queryLogs(new
AuditQueryBuilder().predicates( //
    Predicates.eq(LOG_EVENT_ID,
"something" ),
    Predicates.startsWith(LOG_DOC_PATH,
"/myFolder" ))) ;
```

More on JIRA ticket [NXP-24396](#).

Query

NXQL `ecm:isTrashed` Support

Following evolutions on the trash service, the NXQL property `ecm:isTrashed` has been added to be able to filter queries on trashed or not trashed documents.

NOT IN Support in NXQL

You can now use the `NOT IN` syntax for querying content.

More on JIRA ticket [NXP-25699](#).

Generic Aggregate support

One can now leverage any Elasticsearch aggregate thanks to a new "generic" aggregate type.

These can be used as aggregates in Elasticsearch pageprovider contributions by specifying the relevant `type=` parameter, for example:

```
<aggregate id="cardinality_title"
parameter="dc:title" type="cardinality">
  <field name="cardinality_title_agg"
schema="advanced_search" />
</aggregate>
<aggregate id="missing_description"
parameter="dc:description" type="missing">
  <field name="missing_description_agg"
schema="advanced_search" />
</aggregate>
```


The aggregations are explained in more detail on <https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-metrics.html>

More on JIRA ticket [NXP-25827](#).

Some Built-In Page Providers Moved to Elasticsearch

`REST_API_SEARCH_ADAPTER` and `all_collections` page providers have been added to the default list of page providers provided by Elasticsearch. If you have defined your own `elasticsearch.override.pageproviders` then it is recommended to add those two to your list.

More on JIRA ticket [NXP-24346](#).

Fulltext Extraction Can Be Leveraged More Easily

Fulltext maximum size is now 128 KB by default. To change this, the repository configuration can be updated to use another `fieldSizeLimit`, see the documentation.

The `binaryTextUpdated` event now contains two properties of interest to know what was update exactly:

- `systemProperty` contains the name of the property updated
- `systemPropertyValue` contains the value

More on JIRA ticket [NXP-25716](#) and [NXP-25279](#).

Conversion

Better Thumbnail Quality

Thumbnails size has been set to 1000x1000 pixels (previously 350x350).

More on JIRA ticket [NXP-24717](#).

ConversionException

When calling a converter directly through its name, a check is done on its source MIME type to see if it can handle the input blob. A `ConversionException` is thrown if the converter can't handle it. This typically avoid having a converter open a format that it doesn't know how to handle.

More on JIRA ticket [NXP-25840](#).

Rendition

Renditions on proxy

Renditions are now returned by Rendition Service on proxy objects. Renditions from the source document are returned.

More on JIRA ticket [NXP-24636](#).

Suggestion Service

The suggestion bar now uses `match_phrase_prefix` and is based on the following query:

```
SELECT * FROM Document WHERE /*+ES:
INDEX(dc:title.fulltext)
OPERATOR(match_phrase_prefix) */
ecm:fulltext.dc:title LIKE '?' AND ecm:mixinType !=
'HiddenInNavigation' AND ecm:isVersion = 0 AND
ecm:isTrashed = 0 AND ecm:parentId IS NOT NULL
```

It is made more robust to plurals, etc. More on JIRA ticket [NXP-18198](#).

Elasticsearch

Reindex Operation Based on Bulk Action Framework

[Elasticsearch.BulkIndex](#) is available as an operation to perform a full reindex of the repository based on using BAF and Nuxeo Streams. This is really a great news when dealing with big repositories where re-indexing content would take days: you can resume on any system availability problem, without having to restart again.

More on JIRA ticket [NXP-26032](#).

Elasticsearch 6.3

Elasticsearch 6.3+ is now required. See upgrade instructions.

More on JIRA ticket [NXP-24102](#).

Support of X-Pack

The use of Elasticsearch X-Pack is now allowed, [see documentation](#).

More on JIRA ticket [NXP-23048](#) and [NXP-26074](#).

More Like This Hint

A new hint is available that allows to leverage the More Like This query of Elasticsearch (<https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-mlt-query.html>)

Ex:

```
SELECT * FROM Document WHERE /*+ES:  
INDEX(dc:title.fulltext,dc:description.fulltext)  
OPERATOR(more_like_this) */ ecm:uuid = '1234'
```

will take the most frequent terms of the title and description of document 1234 and find documents that also match those terms.

More on JIRA ticket [NXP-25315](#).

Audit and UID indexes configured in translog async

By default translog are committed per request, configuring them with the async option relieves a lot the stress on the hard drives. This behaviour is now applied to all indexes: repository, audit and uid, when

option `elasticsearch.index.translog.durability=async` is set in nuxeo.conf.

More on JIRA ticket [NXP-25587](#).

Tag Service

Sanitization

A new configuration

property `nuxeo.tag.sanitization.enabled` has been added to enable/disable tag sanitization (true by default).

More on JIRA ticket [NXP-25035](#).

Bulk Service (aka "Bulk Action Framework")

The BulkService is a new Nuxeo Platform service that allows to persist a document set homogenous to an NXQL query (and in the future to a page provider) so as to process an "action" on each of the documents. The action is processed in small batches. It is possible to remotely start a bulk using

the `Bulk.RunAction` operation that accepts as a parameter the name of the action and an NXQL query for specifying the list of documents on which to run the bulk. The service allows to get a status on a given "Bulk" via the get status endpoint: `/nuxeo/api/v1/bulk/{commandId}`

Actions can be contributed via an extension point. [A few actions](#) are already available such as:

- `setPropertyies` that allows to bulk set a some properties values on a set of documents.

- [automation](#) that allows to execute an automation chain or script on the set of documents
- [csvExport](#) that allows to export in CSV the content of the repository.

It is also possible to run a bulk command on a page provider using the REST API:

HTTP Method	Path	Request Body	Response
POST	/nuxeo/api/v1/search/bulk/{actionId}	Bulk action parameters	Bulk action status
POST	/nuxeo/api/v1/search/pp/{pageProviderName}/bulk/{actionId}	Bulk action parameters	Bulk action status
POST	/nuxeo/api/v1/search/saved/{savedSearchId}/bulk/{actionId}	Bulk action parameters	Bulk action status

The Bulk Service uses Avro for encoding all its messages in Nuxeo Streams. Computations are used for performing the actions which means that all Bulk service commands benefit from a batching and retry policy, configurable.

More on JIRA

tickets [NXP-24837](#), [NXP-25060](#) and [NXP-25097](#), [NXP-25249](#), [NXP-25250](#), [NXP-25301](#), [NXP-25391](#).

Annotations Service

Annotation Java Service

A new annotation service has been added, it stores annotations in the repository. Furthermore new facet ExternalEntity has been created to handle Annotations or Comments created from an external service like ARender. This facet allows to store the serialized entity.

More on JIRA ticket [NXP-24096](#) and [NXP-24725](#).

Annotation REST API Adapter

The web adapter "annotation" has been added on the document resource so as to retrieve and set annotations on documents using the REST API.

More on JIRA ticket [NXP-24364](#).

Configuration Option for Annotations Location

A new configuration

property [nuxeo.annotations.placeless.storage](#) is available to change how the annotations are stored. If set to [true](#) (default), the annotations are placeless documents. If set to [false](#), the annotations are stored in a hidden folder. This folder is created under the domain of the annotated document, or under the root if no domain.

More on JIRA ticket [NXP-24933](#).

Comment Service

New implementation

The comment service no more makes use of the relation service. Comments now own:

- highest commented document id
- parent comment id (if exists)

Comments are stored under (First folder in the hierarchy of documents of the commented document / comments). A migrator has been implemented, see migration notes.

More on JIRA ticket [NXP-25425](#).

REST API for the comment services

New REST endpoints have been added for the comment service allowing on a document resource to post and retrieve comments. A comment has the following JSON representation:

```
{
  "id": "00000000-0000-0000-0000-000000000000", //
comment id
  "documentId":
"00000000-0000-0000-0000-000000000000", // commented
document model id
  "author": "Joe",
  "text": "A comment example",
  "creationDate": "1970-01-01T00:00:00Z",
  "modificationDate": "1970-01-01T00:00:00Z",
```



```
"entityId": "...", // the entity id in external
system (present when comment is created from another
system)
```

```
"origin": "...", // the entity origin (present when
comment is created from another system)
```

```
"entity": "..." // the serialized external entity
(present when comment is created from another system)
}
```

See details about the endpoints on the linked ticket. More on JIRA ticket [NXP-24746](#).

New Method for Creating an Answer at a Specific Place

- API in [CommentManager](#) is able to create sub-comments in a specific location
- The [CommentableDocument](#) adapter is enriched to provide the above service
- The [CommentableDocument](#) adapter is enriched to provide the service of creating comments in a specific location using the existing

```
API org.nuxeo.ecm.platform.comment.api.CommentM
anager#createLocatedComment\(org.nuxeo.ecm.core
.api.DocumentModel,
org.nuxeo.ecm.core.api.DocumentModel,
java.lang.String\)
```

More on JIRA ticket [NXP-24863](#).

External Comment entity facet

In order to integrate with external collaboration tools, a new generic external comment facet has been added to the Comments addon, that allows to deal with comments that are managed by external systems: reference them and even serialise their content. This has been designed for now within the scope of the ARender integration.

More on JIRA ticket [NXP-25070](#).

Automation

More Java Objects in Automation Scripting

It's now possible to allow specific Java classes to be used via Automation Scripting, by default we add:

- `java.util.ArrayList`
- `java.util.Arrays`
- `java.util.Collections`
- `java.util.UUID`
- `org.nuxeo.runtime.transaction.TransactionHelper`
- `org.nuxeo.ecm.core.api.Blobs`

- `org.nuxeo.ecm.core.api.impl.blob.StringBlob`
- `org.nuxeo.ecm.core.api.impl.blob.JSONBlob`

Other classes can be added and previously-allowed ones denied, through:

```
<require>org.nuxeo.automation.scripting.classfilter</require>
  <extension
target="org.nuxeo.automation.scripting.internals.AutomationScriptingComponent" point="classFilter">
  <classFilter>
    <allow>com.example.MyClass</allow>
    <allow>com.example.MyOtherClass</allow>
  </classFilter>
</extension>
```

```
<deny>org.nuxeo.runtime.transaction.TransactionHelper
</deny>
  </classFilter>
</extension>
```

One can use `<deny>*</deny>` to disallow all previously-allowed classes.

The default contribution now allows scripting code like:

```
function run(input, params) {
  var uuid =
java.util.UUID.randomUUID().toString();
```

```
    return  
org.nuxeo.ecm.core.api.Blobs.createJSONBlob("{\"uuid':  
\"\" + uuid + \"\"}");  
}
```

More on JIRA ticket [NXP-25020](#).

Async adapter for Automation

An `@async` adapter has been added so as to easily call any operation asynchronously and to have means to get a status on the execution. See the documentation on the ticket below.

More on JIRA ticket [NXP-26172](#).

User Manager

UserGroup.Suggestion Operation Handles Fullname Search

The `SuggestUserEntries` operation performs a full name user search, e.g. typing "John Do" returns the user with first name "John" and last name "Doe". Typing "John" still returns the "John Doe" user and possibly other users such as "John Foo". Respectively, typing "Do" returns the "John Doe" user and possibly other users such as "Jack Donald".

More on JIRA ticket [NXP-24583](#).

New APIs with Query Builder support for more complex queries

In order to remove any post filtering actions, the usermanager component now makes use of the new directory filtering capabilities. Two new apis have been added that takes into account the QueryBuilder object that can be passed for defining the search criteria.

More on JIRA ticket [NXP-19264](#).

Batch upload

Optimised Multipart/Form Upload

When uploading content to Nuxeo using the multi-part/form-data way, no useless copy is made on the way, optimising drastically the upload performance with large videos when using this upload method.

More on JIRA ticket [NXP-24384](#).

User Registration

Stronger Enforcement on Groups Validation for Newly Created Users

Non-administrator users can only invite members from their own group(s).

More on JIRA ticket [NXP-24653](#).

Key Value Store

KeyValue Store and Lock Manager on MongoDB for MongoDB template

The MongoDB template now makes use by default of the MongoDB implementations for the KeyValue store and the Lock Manager.

More on JIRA ticket [NXP-25617](#).

OAuth

Rest API for OAuth 2 Tokens

Some new endpoints have been added to handle CRUD operations on provider and client OAuth tokens.

`GET/PUT/DELETE /oauth2/token/provider/<providerId>/user/<username>` -> perform CRUD on provider tokens

`GET /oauth2/token/provider` -> retrieve all provider tokens for current user

`GET/PUT/DELETE /oauth2/token/client/<clientId>/user/<username>` -> perform CRUD on client tokens

`GET /oauth2/token/client` -> retrieve all client tokens for current user

`GET /oauth2/client` -> retrieve all oauth2 clients

`GET /oauth2/client/<clientId>` -> retrieve an oauth2 client

More on JIRA ticket [NXP-24578](#).

Transient Store

Batch Handler

The platform now provides a way to plug custom logics to upload content to a transient store, by contributing a Batch Handler.

```
<extension
target="org.nuxeo.ecm.automation.server.BatchManager"
point="handlers">
<batchHandler>
<name>foo</name>
<class>org.someorg.somepackage.SomeClassThatImplementsBatchHandler</class>
<property name="transientStore">${backingTransientStore}</property>
<property name="key1">value1</property>
<property name="key2">value2</property>
...
<property name="keyN">valueN</property>
</batchHandler>
</extension>
```

An S3 implementation of this batch handler has been added, so as to be able to upload to S3 directly and to benefit from S3 accelerated upload infrastructure (See the new addon here after).

More on JIRA ticket [NXP-24208](#).

Configurable blobProvider for KeyValueBlob
transientStore

A `KeyValueBlobTransientStore` can now specify the ids of the key/value store and blob provider to use, instead of defaulting to the name of the transient store itself:

```
<extension
target="org.nuxeo.ecm.core.transientstore.TransientSt
orageComponent" point="store">
<store name="mytransientstore"
class="org.nuxeo.ecm.core.transientstore.keyvalueblob
.KeyValueBlobTransientStore">
<property name="keyValueStore">mykeyvaluestore</
property>
<property name="blobProvider">myblobprovider</
property>
...
</store>
</extension>
```

More on JIRA ticket [NXP-24847](#).

Authentication

Stateless authentication with JWT tokens

There is a new Java API to acquire a JWT token to authorize a user:

```
JWTService service =
Framework.getService(JWTService.class);
String token = service.newBuilder().build();
```


The builder can also be used to add specific claims in the token (only CLAIM_SUBJECT is meaningful to the authenticator for now) and a TTL.

The token should then be propagated and passed by a third-party service wishing to connect to Nuxeo using the `Authorization: Bearer <token>` request header. As a compatibility fallback, the request query parameter `access_token` can also be used.

This solution is compatible with cluster installations.

More on JIRA ticket [NXP-24734](#).

SAML Authentication working with LDAP directories

Two new parameters are now available when configuring SAML authentication plugin:

- `userResolverCreateIfNeeded` to create the user if it does not exist in the repository (default value is true)
- `userResolverUpdate` to update the user if present in the repository (default is value true) When set to true, both parameters require a user directory that is not read-only. They should be set to false with readonly directories.

More on JIRA ticket [NXP-25062](#).

SAML: time skew support

A new `nuxeo.saml2.skewTimeMs` configuration property to control the clock skew in milliseconds has been introduced. Default value is $60 * 1000$ (1 minute).

More on JIRA ticket [NXP-24766](#).

Stronger digest for PORTAL_AUTH plugin

It is now possible to configure the algorithm used for the digest among the list given [here](#). Previously, MD5 was systematically used>.

```
<parameter name="digestAlgorithm">SHA-512<parameter>
```

More on JIRA ticket [NXP-25887](#).

Callback URL on logout

A new `callbackUrl` parameter has been added to the logout URL so that if the authentication plugin doesn't provide a redirect, that URL is the one used for the redirect.

More on JIRA ticket [NXP-26002](#).

REST API

Nuxeo Rest API Adapters are Now Overridable

`Nuxeo-AllowOverride: true` has been added to the REST module of Nuxeo Platform so that it becomes possible to override an adapter.

More on JIRA ticket [NXP-24532](#).

Nuxeo REST API now consumes application/json

The REST API now produces and consumes only application/json as content type. `application/json+nxentity` is now never returned as content type response; the server does not expect it as content type request.

- `application/json+nxrequest` content type should not be used anymore. It still works but it's deprecated server side. `application/json` should be used instead when POSTing to automation. `application/json+esentity` which was never used has also been removed.

More on JIRA ticket [NXP-25036](#).

Enrichers can now apply to the `blob` type.

For instance, to get the links to open a LiveConnect blob in all applications associated to its MIME type, you can use the `enrichers-blob: appLinks` enricher. Any blob property, e.g. `file:content`, will then be enriched in the following way:

```
{
  "file:content": {
    "name": "...",
    "mime-type": "...",
    ...,
    "appLinks": [
      {
        "appName": "...",
```

```
        "icon": "...",
        "link": "...",
    },
    ...
]
}
```

More on JIRA ticket [NXP-26126](#).

More properties on the document object

[proxyTargetId](#) and [versionableId](#) are now available on the JSON of a document object.

More on JIRA ticket [NXP-25909](#).

ManagedBlob Decoder

A generic JSON decoder has been implemented for managed blobs.

More on JIRA ticket [NXP-24925](#).

Key/Value Store: Increment API

New APIs are available on [KeyValueStore](#).

Optimized storage of [Long](#) values:

- `put(String key, Long value)`
- `put(String key, Long value, long ttl)`

- `getLong(String key)`
- `getLongs(Collection keys)`

Atomic increment:

- `addAndGet(String key, long delta)`

More on JIRA ticket [NXP-23745](#).

CSV Export Service

A new service has been added for performing CSV Exports. It is exposed via the Bulk Action Framework as an action (`csvExport`). It provides ability to select schemas and/or XPath properties that should be exported, as well as a few of options, zip, sort and the language in which the vocabularies should be translated at export time. Since it relies on the Bulk Action Framework, it is robust enough to export asynchronously hundreds of thousands of lines without any difficulties.

More on JIRA ticket [NXP-25571](#).

AWS Service

A new service has been added to retrieve credentials and other configurations information. A template `aws` is available, to define the AWS configuration. When activated (which is automatically done by the `marketplace-amazon-s3` package), the following `nuxeo.conf` properties are available:

nuxeo.aws.accessKeyId nuxeo.aws.secretKey nuxeo.aws.region
Two new methods are then available to retrieve the information in the code:

- `NuxeoAWSCredentialsProvider.getInstance()`
- `NuxeoAWSRegionProvider.getInstance().getRegion()`

More on JIRA ticket [NXP-25075](<https://jira.nuxeo.com/browse/NXP-25075>).

Packaging / Distribution / Miscellaneous

Java 11 Support

Nuxeo LTS 2019 is supported on Java 8 and Java 11.

More on JIRA ticket [NXP-24210](#).

CSRF

The platform now supports CSRF token that can be retrieved and then added to any requests to the server, with a mode where their presence is systematically checked. To enable it:

```
<extension  
target="org.nuxeo.runtime.ConfigurationService"  
point="configuration">
```

```
<property name="nuxeo.csrf.token.enabled">true</  
property>  
</extension>
```

You will find more information on the required flow on the linked ticket below. CSRF tokens allow to have deep protection against attacks of the system.

More on JIRA ticket [NXP-25903](#).

nuxeoctl register --offline

It is now possible to register offline a new instance from the terminal with the following command, that will then start a command line wizard:

```
nuxeoctl register --offline
```

More on JIRA ticket [NXP-23815](#).

Reporting metrics to StatsD

It is possible to send Nuxeo metrics to StatsD [More about StatsD](#) just by configuration.

```
metrics.statsd.enabled=true  
# report every 15s  
metrics.statsd.period=15  
metrics.statsd.host=my-statd-server  
metrics.statsd.port=8125
```

Enabling Dev mode from the wizard

It is now possible to enable the dev (necessary to perform a hotreload with the browser extension) from the wizard.

More on JIRA ticket [NXP-25508](#).

Disabling Studio package dependency validation

A runtime configuration property has been added that allows to disable the dependency validation on the Studio package: `studio.snapshot.disablePkgValidation`. Until we manage to optimize performances on the server side, this allows to gain up to 30 seconds on the hot-reload action, highly recommended!

More on JIRA ticket [NXP-25719](#).

HSTS Policy

The HSTS header is enabled by default when HTTPS is in use. It forces only HTTPS requests.

More on JIRA ticket [NXP-24254](#).

AWS Client Upgrade

The Nuxeo Platform now uses version 1.11.323 of the Amazon SDK. This notably allows using AWS Comprehend service.

More on JIRA ticket [NXP-24981](#).

Log4J 2

Nuxeo now uses Log4j 2 as its logging backend instead of Log4j. Two logging APIs are generally available in Nuxeo:

- Commons Logging
- Log4j 2 API SLF4J is still available, but Log4j 2 API are preferred by reason of its lambda support.

More on JIRA ticket [NXP-23863](#).

JSON output for Log4J

Required dependencies have been added so that it is possible to configure the logs to be serialized as valid JSON. See sample log4j configuration on the linked ticket.

More on JIRA ticket [NXP-25096](#).

JDK Check Enforced in nuxeoctl

A check on the presence of a JDK per is enforced at startup in nuxeoctl

More on JIRA ticket [NXP-21200](#).

Target Platform filtering on mp-listall command

mp-listall command has been optimized by listing only relevant packages for the version of the Nuxeo server it is run on. More on JIRA ticket [NXP-22520](#).

Tomcat upload time configurable

Default Tomcat `connectionUploadTimeout` has been set to 1 min and is now configurable in `nuxeo.conf`. It is the time during which tomcat accepts to not receive any byte of information, making the upload experience more resilient to bad network connections for instance.

More on JIRA ticket [NXP-25037](#).

Making Use of Tomcat 8 Rewrite Valve

It is now possible to contribute rules to Tomcat Rewrite Valve by leveraging our deployment preprocessor.

Ex (deployment-fragment.xml):

```
<?xml version="1.0"?>
<fragment version="1">
  <!-- JSF permalink redirect -->
  <extension target="rewrite#RULE">
    RewriteRule ^/nxdoc/default/(.*)/view_documents /
ui/#!/doc/$1 [NE,R]
  </extension>

  <!-- ES6 / ES5 code -->
  <extension target="rewrite#RULE">
    RewriteCond %{HTTP_USER_AGENT} .*Chrome.*
    RewriteRule ^/shop/(.*) /shop/es6-bundled/$1 [L]
    RewriteRule ^/shop/(.*) /shop/es5-bundled/$1 [L]
  </extension>
```

</fragment>

More on JIRA ticket [NXP-25040](#).

Multi-Region Replication

A set of addons and scripts have been produced for the ability to replicate all the Nuxeo data in near real time into another hosting region, using Kafka streams for the replication. The intent is to be able to recover from a disaster in less than a minute. Contact Nuxeo for more information about it.

More on JIRA ticket [NXP-24189](#).

Servlet API 3.1

Servlet API 3.1 is now used in Nuxeo code.

More on JIRA ticket [NXP-24386](#).

Nginx File Upload / Download Acceleration

This feature allows an Nginx proxy in front of Nuxeo to accelerate uploads and downloads.

In order to enable this feature you must add to

your `nuxeo.conf:nuxeo.nginx.accel.enabled=true`

IMPORTANT: when this feature is enabled, you **MUST** front the Nuxeo server with a Nginx proxy that correctly deals with the X-Accel-Location header. Not doing so would be a **SECURITY HOLE**.

More on JIRA ticket [NXP-25831](#).

Addons

Nuxeo Web UI

Trash

Add trash functionality and management to document deletion. Document deletion moves it to trash. In order to manage trash:

- Documents with Folderish facet added a trash pill to manage deleted documents.
- New trash search on the main menu. Has a faceted search on path, size, authors, and text. Trashed documents can be restore or permanently deleted by users with Manage Everything permission. A new `EmptyTrash` operation allows to permanently delete a Folderish trash content which is available on the Folderish trash pill UI. Finally, a set of functional tests for new trash features.

More on JIRA ticket [NXP-23798](#).

Document Metadata Differences

Documents can be compared to other documents or within its versions. Common schemas are compared. UI is fully responsive and can show only different data or all of it. The comparison UI is fully configurable.

- [NXP-24780](#) Allows documents to be selected to compare.
- [NXP-24781](#) Permits user to pick a different version to be compared.
- [NXP-24786](#) Enables switching comparing documents position.
- [NXP-24784](#) Visualizes metadata differences on two documents.
- [NXP-24784](#) Allows custom element to be added to compare different metadata properties. Defines custom elements to be used on blobs, document and user references.
- [NXP-25941](#) Leverage ARender difference capabilities on document comparison (ARender addon).

Default Comparison Elements

Allows custom element to be added to compare different metadata properties. Defines custom elements to be used on blobs, document and user references.

Blobs: In case of blobs, the name should be the content for comparison. An action should also be present so that if the MD5 of those blobs is different, the action allows the user to launch a blob content comparison (using ARender comparison).

Document references: should compare the document title

User references: should compare with the user tag widget

A custom element was introduced to compare documents: `nuxeo-diff`. This element now dynamically loads the file `diff/imports.html`, which imports custom elements from the `diff/elements` folder and registers them in a global registry using a set of rules. These rules define for what property names or property types should this elements be used to represent the differences (or the raw value in case there are no differences). A sample of a valid `diff/imports.html` follows:

```
<link rel="import" href="elements/nuxeo-default-diff.html">
<link rel="import" href="elements/nuxeo-blob-diff.html">
<link rel="import" href="elements/nuxeo-dcdescription-diff.html">
<script>
  Nuxeo.Diff.registerElement('nuxeo-blob-diff',
    {type: 'blob'});
  Nuxeo.Diff.registerElement('nuxeo-dcdescription-diff',
    {property: 'dc:description'});
</script>
```

These elements are used as children by `nuxeo-object-diff`, which is the element responsible for representing a diff for a given property, loading those elements based on the property name or property type. In case no custom element is registered for the given criteria, the default diff element (named `nuxeo-default-diff`) is used.

In order to allow the current custom elements to be overridden and new elements to be added, the following files are not vulcanized:

- [nuxeo-diff-page.html](#)
- [diff/imports.html](#)
- [diff/elements/.html](#)

The [nuxeo-diff-page](#) stamps [nuxeo-diff](#), which can be parameterized. [nuxeo-diff](#) exposes several customization properties, including the headers, enrichers and schemas used to fetch the documents. The headers property can be used to determine whether documents are fetched with or without resolved entities.

More on JIRA ticket [NXP-24785](#)

Document Publishing

Added document publishing capabilities:

- [NXP-24426](#) Publish document on a publishable space.
- [NXP-25687](#) New layout to publications with information on original document.
- [NXP-24434](#) Allows default rendition to be configured as a contribution.
- [NXP-24428](#) List document own publishing items.

- [NXP-24435](#) Republish the newer and latest version.
- [NXP-24432](#) Unpublish documents from original one.
- [NXP-24427](#) RAbility to publish a set of documents in bulk.

Comments on Document

Allows comments on commentable document:

- [NXP-25535](#) Added comments and replies on commentable document layout.
- [NXP-25536](#) Allows creation of new comment or reply.
- [NXP-25537](#) Deletion and edition of comments and replies.

Machine Learning Suggestions on Document Creation Forms

Integrates Machine Learning custom models suggestion in Web UI to provide predictions thanks to the [Nuxeo-AI-Core plugin](#).

- [NXP-NXP-25565](#) Suggestion integration on form inputs.
- [NXP-26070](#) Suggestions working on multivalue input fields.

CSV Export UI

It exposes CSV export in the UI, allowing users to download a listing in CSV format. [NXP-25934](#) Ability to export results and folderish content in CSV.

Direct Upload to 3rd-Party Service

Batch upload refactored to support third party providers. It is possible to integrate providers for feature rich and performance upload. To this end, the upload behaviour now supports external providers and allows features like progress and multipart.

More on JIRA ticket [NXP-24269](#).

Delegate and Reassign Tasks

Migrated missing features from workflow to allow delegation and reassignment on workflow tasks:

- [NXP-24997](#) Migrated task reassign on workflows.
- [NXP-24998](#) Migrated task delegation on workflows.

User Actions Scalability

Provides a solution for large number of user actions and small screens. We added a drop menu for secondary user actions.

- Three slots, to which actions are contributed, were wrapped in a responsive menu: [DOCUMENT_ACTIONS](#), [BLOB_ACTIONS](#) and [RESULTS_](#)

SELECTION_ACTIONS.

- Three css variables were added to control the width of these menus, respectively: `--nuxeo-browser-actions-menu-max-width` `--nuxeo-document-blob-actions-menu-max-width` `--nuxeo-results-selection-actions-menu-max-width`
- These variables can be overridden in the themes.

More on JIRA ticket [NXP-25146](#).

Orderable Folders

Orderable Folders are now available in Nuxeo Platform with up and down actions. It works with multiple selected documents. For this purpose a new operation is available to order child documents. Navigation tree now takes into account order on Orderable Folders.

More on JIRA ticket [NXP-24254](#).

User Cloud Settings

Adds an interface to allow users to manage their OAuth2 tokens. Managing inbound and outbound authorisations. [NXP-20773](#) The user has a way to manage all the cloud services permission tokens granted to the Nuxeo account. [NXP-22588](#) Users can now manage external applications authorisations with Nuxeo. [NXP-24841](#) Improves performance on Cloud Service area to lazy load pages. [NXP-24578](#) New Rest API for OAuth 2 tokens.

Some new endpoints have been added to handle CRUD operations on provider and client OAuth tokens:

GET/PUT/DELETE /oauth2/token/provider/<providerId>/user/<username> -> perform CRUD on provider tokens

GET /oauth2/token/provider -> retrieve all provider tokens for current user

GET/PUT/DELETE /oauth2/token/client/<clientId>/user/<username> -> perform CRUD on client tokens

GET /oauth2/token/client -> retrieve all client tokens for current user

GET oauth2/client -> retrieve all oauth2 clients

GET oauth2/client/<clientId> -> retrieve an oauth2 client

More on JIRA ticket [NXP-24252](#)

Drag and drop

It is now possible to use drag and drop user interaction to move and copy documents to folders and collections on a listing.

[NXP-24351](#) Enables user to add documents to sibling collections by drag and drop. [NXP-24350](#) Enables user to move documents to sibling folders by drag and drop.

More on JIRA ticket [NXP-22807](#)

ARender annotation pill

Introduces new file-based document pill to allow annotations with ARender.

More on JIRA ticket [NXP-25107](#)

Configuration service on Web UI

Allowing a namespaced properties to configure Web UI. Properties can be marked as a list and if defined many times, values will be appended as comma separated values. You can override existing list property with the `override` attribute:

```
<property name="nuxeo.list.value" list="true">foo</property>
<property name="nuxeo.list.value">bar</property>
<!-- nuxeo.list.value is now "foo,bar" -->
<property name="nuxeo.list.value"
override="true">newValue</property>
<!-- nuxeo.list.value is now "newValue" -->
```

We then put properties namespaced with `org.nuxeo.web.ui` into the `Nuxeo.UI.config` javascript namespace that Web UI (and even standalone elements) can access and use them.

- [NXP-25678](#) Added a configuration service to allow Web UI properties definition with a namespace.
- [NXP-25512](#) Allows to override/extend fetch properties and enrichers used to browse a document.

UX Improvements

Several UX improvements were added to Web UI:

- [NXP-24535](#). Fixed link on user invitation e-mail that led to "page not found".
- [NXP-24347](#). The "Remove from Collection" action is now displayed on all document types with the Collection facet.
- [NXP-24082](#). A new close action button has been added to let you hide the drawer. It appears on the middle right side of the drawer.
- [NXP-25021](#) A login banner is displayed when the WebUI session has expired.
- [NXP-25202](#) Adds an initial progress page for fast feedback on loading.
- [NXP-24648](#) Improves UX by notifying user when connection is offline making Web UI not reacting.
- [NXP-24752](#) Introduces mixin that allows selection toolbar appearance and position customisation. Adds theme variable for selection color and uses this by default.
- [NXP-24692](#) To assure the best legibility, the color of the "shortcut" icons was changed to have a high contrast to the background color.

- [NXP-22062](#) Document UIDs on audit log now have a link back to document.
- [NXP-24894](#) Fixes drawer queue keyboard navigation.
- [NXP-24466](#) Document information included in administration audit logs.
- [NXP-25104](#) New improved document layout resolving selection actions ambiguity and improving space usage.
- [NXP-22453](#) Brings direct links to document tabs and subpages.
- [NXP-25638](#) Added replace file action to improve preview space.
- [NXP-25630](#) Improved drop zone ambiguity with live connect.
- [NXP-25352](#) Several improvements on the drop file element.
- [NXP-20604](#) On granting permission is now possible to add multiple users or groups.
- [NXP-25325](#) With auto-search off, pressing enter allows to quickly submit the search.
- [NXP-25740](#) Picture subtype now inherit Picture view layout.

- [NXP-25637](#) Improved listing element layout and composition.
- [NXP-25636](#) Improved document details area to optimize content preview.

Performance Improvements

Several actions to provide better performance on Web UI.

- [NXP-25158](#) Enables a way to skip aggregates computation, making the request potentially faster. This is used on a range or pagination queries as there is no need to update aggregates. This can be used by adding `skipAggregates=true` as a HTTP header when invoking the search rest endpoint.
- [NXP-25202](#) Adds an initial progress page for fast feedback on loading.
- [NXP-25139](#) Updates cache headers for Web UI static resources.
- [NXP-24820](#) The memory footprint of the Web UI has been reduced.
- [NXP-24422](#) Edge performance was analysed and new improvement actions were planned.

- [NXP-25385](#) Largely improved caching strategy.
- [NXP-24880](#) Allows search initialisation to skip aggregation for large repositories.
- [NXP-25320](#) Aggregation navigation on listing scroll removed by default.
- [NXP-25302](#) Performance improvements on grid element.

More on JIRA ticket [NXP-25038](#)

Performance Audit System

Added a performance audit system to measure key metrics. [NXP-25303](#) Added performance [marks](#) to key elements that can be displayed on the UI on demand. [NXP-25414](#) As relevant metrics we identified those already collected by the devtools of most browsers:

- dom content loaded
- first contentful paint
- first pain
- on load

Additionally, we're including an experimental set of network metrics, which take all requests performed by the browser into account to compute:

- finish time: time taken to complete all requests since `PerformanceTiming.fetchStart`
- request count: the number of requests issued
- transfer size
- decoded body size

However, some requests seem to be ignored and not included in the performance entries (<https://developer.mozilla.org/en-US/docs/Web/API/Performance/getEntries>) on Chrome, which renders this data incomplete in this browser. Furthermore, transfer size and decoded body size cannot be retrieved on Safari and Edge.

We are also storing a set of custom performance marks and measurements, which are aligned with what we want to measure:

- [mark] `nuxeo-app.ready`: marks the moment when the `nuxeo-app` element is ready
- [mark] `nuxeo-app.page-changed`: marks the last moment where there was a page switch (not set when the first page is loaded, since there is not transition)
- [mark] `nuxeo-app.page-loaded`: marks the moment where the last dom-change event was fired from within the current

page, .i.e, the last moment in which there were changes in the page

- [measurement] `<page-name>.dom-changed`: a measurement between the moment in which the last `dom-change` event is fired from within the current page and the last `page-change` mark (or `PerformanceTiming.fetchStart` if `undefined`) - this is mostly helpful for development and debugging purposes We added these metrics to the performance analyzer added in NXP-25303. You can try these out by running:

```
Nuxeo.Performance.report({networkStats: true});
```

New Languages

[NXP-24451](#) and [NXP-24445](#) Italian, Dutch and Swedish languages have been added to Web UI and Nuxeo Elements. [NXP-25112](#). Adds Chinese simplified and Hebrew locales for Web UI.

Nuxeo Elements

Migration to ES6

Migrates all elements to ES6 class-based syntax which allows usage of Polymer 2 and move away from Polymer legacy APIs:

- [ELEMENTS-506](#) Migrates nuxeo-dataviz-elements to ES6.

- [ELEMENTS-505](#) Migrates nuxeo-elements to ES6.
- [ELEMENTS-640](#) Migrates Widgets to ES6 and Polymer 2.
- [ELEMENTS-635](#) Migrates UI Elements behaviors, slots and filters to ES6 and Polymer 2.
- [ELEMENTS-637](#) Migrates Actions and Viewers to ES6 and Polymer 2.
- [ELEMENTS-639](#) Migrates Listing elements to ES6 and Polymer 2.
- [ELEMENTS-636](#) Migrates Document Permissions and User Management to ES6 and Polymer 2.
- [ELEMENTS-638](#) Migrates Demos and Tests to ES6 and Polymer 2.

More on JIRA ticket [ELEMENTS-504](#).

Performance improvements

Several actions to provide better performance on Nuxeo Elements.

- [ELEMENTS-673](#) Improves performance on lazy loading to not fetch already loaded documents
- [ELEMENTS-701](#) Adds option to skip aggregation computation in nuxeo-page-provider for better performance. More on JIRA

ticket [NXP-25038](#)

- [ELEMENTS-726](#) Improved document preview performance.

UX Improvements

Several UX improvements were added to Nuxeo Elements that impact Web UI 10.2:

- [ELEMENTS-685](#) Removes background on image previewer and exposes it as an attribute for better visual.
- [ELEMENTS-707](#) Improves requirement symbol on form's labels.
- [ELEMENTS-666](#) Improves nuxeo-user-suggestion UI to make it non-ambiguous.
- [ELEMENTS-622](#) Selection UI improved to be consistent in all view mode.
- [ELEMENTS-687](#) Changes image previewer dragging behaviour to contain image on bounds.
- [ELEMENTS-677](#) Changes blob delete action to distinguish from document trash.
- [ELEMENTS-736](#) The date format is customizable in WebUI.

- [ELEMENTS-740](#) Allow share link copied on click.
- [ELEMENTS-703](#) Added a 100% zoom action on image preview.

Update Nuxeo JS client dependency

Adapts Element's tests for nuxeo js client 3.6.1

More on JIRA ticket [ELEMENTS-650](#)

Sort aggregates elements

Aggregation widgets can now be sorted by the label with the *sort-by-label* boolean property. i.e. `<nuxeo-checkbox-aggregation sort-by-label .../>`

More on JIRA ticket [ELEMENTS-621](#)

Elements improvements and additions

- [ELEMENTS-669](#) Introduces new element .
- [ELEMENTS-662](#) Changes `nuxeo-date-picker` element to user vaadin date picker for better browser support.
- [ELEMENTS-662](#) Two new elements, `<nuxeo-directory-checkbox>` (multiple) and `<nuxeo-directory-radio-group>` (single) added to present and select directory entry(ies).

- [ELEMENTS-624](#) Exposes `rows` attribute in nuxeo-textarea element.
- [ELEMENTS-610](#) Replaces video javascript library with browser native video element for to better support browsers.

Nuxeo AI

Service to Configure and Serve ML Custom Models

Current AI/Machine Learning services for labelling and classification behave mostly in a very generic manner. They do not take into account specific business logic and data, but only broader and more abstract content. As an example, on a picture with cars, they are very able to recognize cars, but for a carmaker, it would be more relevant to detect what would be the car model and year of release. Even more valuable would be maybe to identify the correct car color in the brand's catalogue and detect any car extras.

To enable business specific classification and recognition, models need to be build and trained with this information. This new service allows customers to build ML custom models that after defined are trained and publish for usage. A ML custom model is defined with:

- NXQL query - to provide the correct dataset of documents to use during training
- Input fields - the metadata that the model should use to do classifications. Ie. The document image, a set of text fields.

- Output field - this is the field that the model will try to predict. These could be a field linked to a vocabulary, a document reference or just a `string` metadata.

After defined, the custom model has two other main stages:

- Training - a ML model will be produced by training the repository model with the definition provided
- Publishing - The result ML model can then be published to be available as service that does Machine Learning classification on the context of a particular business.

Both these stages are performed on Nuxeo AI Cloud Services, but Nuxeo Server provides the means to export the relevant repository content in a way that Nuxeo AI Services can use. Nuxeo Server also allows the means to configure and use published ML models, query and getting predictions to use in UI suggestions.

- [NXP-25477](#) Adds operation to use all enabled Custom Models.
- [NXP-25476](#) Adds a service to register Custom Models to be served.
- [NXP-26001](#) Adds new schema for suggestions metadata.

Documents Export for Training

In order to train a ML Custom Model with proper business data, there is the need to gather and process all the repository

documents on the scope of the ML Custom Model. This provides a documents dataset export to AI cloud in TFRecord format.

- [NXP-25286](#) Document reference to represent Corpus objects.
- [NXP-25295](#) Brings a new Stream processor to transform documents to TFRecord.
- [NXP-25758](#) Made a writer to produce a TFRecord from a collection of documents.
- [NXP-25294](#) Enables a batch processing from an NXQL query and a list of properties to process.

Integrate with AI Cloud

Provides integration between the Nuxeo Server and AI Cloud services to allow dataset export to AI Cloud, pushing defined document sets in an reliable and efficient way to be use in ML Custom Model Training.

- [NXP-25920](#) Integrates a java client to enable Nuxeo to interact with AI Cloud Services.
- [NXP-26189](#) Provides an Admin page to export dataset to AI Cloud services.
- [NXP-25827](#) Added support for numeric ES aggregates : cardinality, missing, count, sum, avg, max, min. Initially, these are used to provide data sanity statistics on document

datasets.

Create the Framework for Document Enrichment Service Integration

In order to take advantage of any type of Machine Learning service, third-party or in-house, for content augmentation we need an infrastructure. This infrastructure needs to be resilient and to scale indefinitely in order to handle a repository with millions of ingested documents.

We call this content augmentation **Enrichments**. They provide any data produced by AI that is related to a document's metadata.

Examples are:

- Identification of objects in pictures;
- Translate metadata fields;
- Classify text content sentiment
- ..

Enrichments are produced from different services and providers, therefore, they come in different shapes and formats. The new **enrichment** schema is divided into three parts:

- Raw - Information that comes directly from the service

- Normalized - Normalized information per type of content. This allows to process information from different providers
- Facet - canonical information that is index for search. ie. in a video with cars races, [normalized](#) might have all the shots/ times that cars appears, while [facet](#) will only indicate the video has cars.

The enrichment pipeline user [Nuxeo Stream](#) and is fully configurable making it flexible to use in any metadata for any type of provider.

- [NXP-25191](#) Adds a new event listener on binary text conversion from files that redirects text to correct stream.
- [NXP-24749](#) Provides a listener for images then redirect their info to the correct image Stream.

More on JIRA ticket [NXP-24605](#).

Event Stream for Data Annotation

Refactor events and integrates them with Streams that will process ML enrichments. Also takes care of results to correct processors to store metadata.

- [NXP-25191](#) Adds a new event listener on binary text conversion from files that redirects text to correct stream.

- [NXP-24749](#) Provides a listener for images than redirect their info to the correct image Stream.
- [NXP-25279](#) Makes the raw binary text available different processes like ML enrichment ones.

More on JIRA ticket [NXP-24722](#).

Turn Raw Data into the Internal Enrichment Metadata Format and Store It

Ingests raw ML services data, normalizes it and exposes as a facet on documents.

- [NXP-25124](#) Adds invalidation process to ML enrichments.
- [NXP-25122](#) Dispatches an event on new ML enrichments.
- [NXP-24771](#) Converts raw data into the internal enrichment schema format.
- [NXP-25651](#) Provides a generic caching system to enrichment process so that paid services are not called more than once without need.

More on JIRA ticket [NXP-24770](#).

Nuxeo AI AWS

Using the new ML Enrichment service on Nuxeo AI, some Amazon machine learning services were integrated:

- Amazon Rekognition - Object tagging and scene detection; facial and celebrity recognition tagging; Text in image tagging.
- Amazon Comprehend - Sentiment Analysis.
- Amazon Translate - Translation. When activated, these new enrichers provide new content what can be used from the `enrichment` schema on the `Enrichable` facet in a normalized and searchable way; or can be configured to be transformed and inserted into a document native metadata.

This addon brings some default configuration templates that can be enable from `nuxeo.conf`:

```
nuxeo.ai.images.enabled=true
nuxeo.ai.text.enabled=true // process text content
main content by default
nuxeo.enrichment.aws.images=true // process image
main file by default
nuxeo.enrichment.aws.text=true // enable sentiment
analyses
nuxeo.enrichment.save.tags=true // transform label
enrichments into the document tags
nuxeo.enrichment.save.facets=true // save enrichment
facet content on documents
nuxeo.enrichment.raiseEvent=true // fire an event
when a new enrichment is obtained
```

- [NXP-24750](#) Adds a Stream processor that integrates AWS Rekognition for image labelling and tagging.
- [NXP-24823](#) Provides a new extension to integrate AWS Comprehend to predict sentiment on main file text.
- [NXP-25053](#) Adds ability to tag images text with OCR.
- [NXP-25192](#) Provides addition of face and celebrity face detection from AWS Rekognition as image tagging.
- [NXP-25599](#) Adds ML enrichment to translate metadata using AWS Translate.

More on JIRA ticket [NXP-24770](#).

Nuxeo AI Image Quality

Introduces new enrichments from [SightEngine](#) as a ML enrichment addon with content warning and image quality Services integrated:

- weapon - the probability that the image contains weapons
- alcohol - the probability that the image contains alcohol
- drugs - the probability that the image contains drugs
- nudity - containing information on the nudity content of the image

- type - containing information on the type of the image
- faces - containing information on the presence of faces / celebrities in the image
- sharpness - a value between 0 (very blurry) and 1 (very sharp)
- contrast - a value between 0 (low contrast) and 1 (high contrast)
- brightness - a value between 0 (very dark) and 1 (very bright)
- scam - the probability that the image contains scammers
- text - containing information on the presence of text in the image
- offensive - containing information on the presence of offensive content in the image
- celebrity - the probability that the image contains a celebrity
- colors - describing the colors of the image received
- media - describing the image received

It is possible to configure this addon from the [nuxeo.conf](#):

```
nuxeo.ai.images.enabled=true // process images wich
will enable them on this addon
nuxeo.enrichment.save.tags=true // convert
enrichments into labels
nuxeo.enrichment.save.facets=true // save enrichment
facet content on documents
nuxeo.enrichment.raiseEvent=true // fire an event
when a new enrichment is obtained
nuxeo.ai.sightengine.apiKey=YOUR_API_KEY // API key
from sightengine to autenticate
nuxeo.ai.sightengine.apiSecret=YOUR_API_SECRET/ / API
secret from sightengine to autenticate
More on JIRA ticket NXP-25725.
```

Nuxeo JSF UI

Back to GWT annotations

The legacy annotations module that had been withdrawn is actually back and can be enabled by setting the `nuxeo.old.jsf.preview` property to true.

```
<extension
target="org.nuxeo.runtime.ConfigurationService"
point="configuration">
  <property name="nuxeo.old.jsf.preview">true</
property>
</extension>
```

More on JIRA ticket [NXP-25110](#)

Amazon S3 Direct Upload for Web UI

New add-on to upload using AWS S3 infrastructure with support for multipart. Allows future integration of other providers. Integrated with Web UI upload with real time upload progress. For 10.2 we have added AWS instance role support for better security.

More on JIRA ticket [NXP-24490](#) and [NXP-24748](#).

Document, Image, Video Annotations with ARender Connector

A first implementation of the ARender SPI bridge has been done so as to be able to preview content stored in Nuxeo using the [ARender previewer](#). It allows to preview and annotate content, may it be an office file, an image or a video, with respect to the repository security, using JWT tokens. A first integration to Web UI is done in the addon, the ARender previewer appears in a new "annotations" tab. Also, comments made for a given annotation are synced with Nuxeo Comments. Deeper integration will be done in the future. The connector also exposes an operation for getting an URL that will open a diff view of binary content ([Document.ARenderGetDiffUrl](#)). A package is available on the marketplace.

More on JIRA ticket [NXP-24104](#), [NXP-24724](#) and [NXP-25071](#) , [NXP-25444](#)().

MS Office 356 Integration: WOPI Implementation

Nuxeo Server is now integrated with Office Online through the implementation of the WOPI protocol. It enables users to benefit from all Office Online features such as:

- viewing and editing Excel, PowerPoint, and Word files, stored in a Nuxeo Server, directly in the browser.
- supporting multiple users editing a document at the same time.
- co-authoring including real-time content updates between all users editing the document, as well as presence information and real-time cursor tracking for each user.

This integration works together with Nuxeo versioning and locking/ auto-locking policies. It also provides an enricher for letting app builders also benefit from this integration, fetching WOPI urls to open and edit documents using the enricher [WOPIEnricher](#).

More on JIRA ticket [NXP-23174](#) and [NXP-25300](#).

Simflofy Connector

A blob provider has been implemented for Simflofy, allowing to access from Nuxeo to any content that Simflofy knows how to federate.

```
<component
name="com.nuxeo.simflofy.blob.provider.test"
version="1.0.0">
<extension
target="org.nuxeo.ecm.core.blob.BlobManager"
point="configuration">
  <blobprovider name="testSimflofy">

<class>com.nuxeo.simflofy.blob.SimflofyBlobProvider</
class>
  <property name="simflofy.url">http://localhost:
8080/simflofy</property>
  <property name="simflofy.username">admin</
property>
  <property name="simflofy.password">admin</
property>
  </blobprovider>
</extension>
</component>
```

The Nuxeo connector of Simflofy knows how to create such documents that reference binaries stored in legacy repositories.

More on JIRA ticket [NXP-24714](#).

WEBDAV

MS Office Temporary Files mime-type Set as Application/ Octet-Stream

They are no more detected as word documents, which avoids applying some useless conversions, and avoid confusion for the users and in the repository, and improve the behavior for Microsoft Office.

More on JIRA ticket [NXP-25817](#).

Microsoft Office Temporary Files Not Trashed

Office temporary files are not immediately deleted, they are not sent to trash, avoiding to fill the trash with documents that are not user-level documents.

More on JIRA ticket [NXP-25818](#).

Nuxeo Platform Importer

Image and Video capabilities for the Nuxeo Stream Random Importer

The Nuxeo importer has an option for filling the repository with random content. This content can now be images or videos, randomly produced by watermarking a set of sample media.

More on JIRA ticket [NXP-24318](#).

IMAP Connector

Nuxeo IMAP Connector addon is now available on Web UI. You can create and configure IMAP folder documents on WebUI, with a sync action to import all unread emails from account. This addon adds a

new custom list view for emails on email folder documents and a custom view layout to email message documents with relevant information about the content, senders, receivers and attachments.

More on JIRA ticket [NXP-23951](#).

Drive (Server part)

The release notes of the Drive client part can be found on [GitHub](#).

Direct Edit Enabled on All Binary Properties

It is now possible to get a Direct Edit link for any binary properties stored on a document in the Nuxeo repository (may it be a custom single binary property or a multi-valued attachment).

More on JIRA ticket [NXP-25994](#).

.lnk Files Ignored

Windows symlink files `.lnk` are now ignored by default.

More on JIRA ticket [NXP-24490](#).

Full Scan Query Optimized

A great optimization has been added lowering heavily the charge of the Elasticsearch cluster when using Nuxeo Drive with Nuxeo.

More on JIRA ticket [NXP-24232](#) and [NXP-24637](#)

Authentication from the Browser

Nuxeo Drive no longer launches its own "embedded browser" and relies on the default desktop browser to open the `drive_login.jsp` page, making it more compatible with various authentication protocols.

More on JIRA ticket [NXP-25519](#)

Nuxeo DAM

Improve Video Processing

The video info (duration, format, etc.) is now computed by an asynchronous work to avoid loading the blob and running `ffmpeg-info` synchronously. This work, in turn, schedules two asynchronous works to process the video storyboard and conversions.

As a consequence, the user might not have the video info in the UI immediately after creating / updating a Video document, needing to refresh the page once the asynchronous work is done. This change allows a better behaviour when bulk importing videos.

More on JIRA ticket [NXP-24316](#).

Operation to Recompute Image Renditions

You may want to recompute your stored image renditions when you change the resolutions you want to use. A new operation is available for this: `Picture.RecomputeViews`.

More on JIRA ticket [NXP-25791](#).

Nuxeo Vision

Support to Detect Unsafe Content with Aws Rekognition

Using the feature "SAFE_SEARCH_DETECTION" is now possible with the Amazon implementation of Nuxeo Vision.

More on JIRA ticket [NXP-25646](#).

All Backend Service features can be leveraged in Nuxeo Vision

The features to use (and sent to the provider) are no more checked against a predefined list. This allows to use any new feature available without waiting for an update of the plugin. After using such new feature, the caller should use the `getNativeObject` method and handle the results based on the provider's documentation.

More on JIRA ticket [NXP-24499](#).

Amazon Implementation Does a Fallback on Environment Variables Aws Services Configuration Keys

If the keys are not set in `nuxeo.conf` a lookup is made in the environment. More on JIRA ticket [NXP-25596](#).

Nuxeo Quota

Operation to Recompute Partially Quotas

A new operation: `Quotas.RecomputeStatistics` is provided, with optional parameters:

- tenantId / username / path (only one allowed)
- updaterName (defaults to `documentsSizeUpdater`)

More on JIRA ticket [NXP-21017](#).

FS Exporter

New Drive Exporter

The FS Exporter addon that allows to easily perform exports of the repository as a files and folders hierarchy has a new "Drive like" flavor that handles exports like Drive, not exporting attachments and all this, and dealing with conflictual names at a same folder level. It uses the title of the document instead and avoid name collision by adding an integer at the end of tshe path.

More on JIRA ticket [NXP-25846](#).

Farewell

WebEngine GWT Integration

The WebEngine GWT Integration that used for the former annotations system has been removed.

More on JIRA ticket [NXP-24317](#).