Revision History

Plug-in Guide

1.0.0.1

Technical Support

For technical support and the latest news of upgrades, documentation, and related products, visit the Vizrt web site at www.vizrt.com.

Last Updated

21.12.2011

Table of Contents

1 Introduction

The Revision History plug-in adds audit trail functionality to the Content Engine, making it possible to easily view the revision history and old versions of all text content items. Once the plug-in has been installed and correctly configured, every time a content item is saved, a copy of the content item is also archived in an external document store, along with information about the version (the date of the modification and the identity of the person making the change).

The Revision History plug-in adds a **Revision history** option to the Content Studio **View** menu, which you can use to view the history of content items. The **Revision history** dialog displays two versions of a content item (initially the latest version and its predecessor) and highlights the differences between them. Previous and Next buttons allow you to move back and forth through the content item history, comparing adjacent versions.

These simple functions make it very easy to trace the history of a content item and find out when and by whom specific content was added, removed or modified.

The Revision History plug-in has the following limitations:

- The plug-in can only record revisions made **after** the plug-in has been installed and correctly configured.
- The current version of the plug-in only works with textual content. Changes made to binary objects such as images, video and audio files, PDF and Word attachments are not recorded.

The Revision History plug-in requires access to an external document store in which to store its revisions. This store can be located on one of your hosts or indeed anywhere in the network.

The Revision History plug-in currently requires you to use **CouchDB** as an external document store. CouchDB is available from http://couchdb.apache.org/.

2 Installation

The following preconditions must be met before you can install Revision History 1.0.0.1:

- Your Content Engine installation is version 5.3.0 or higher.
- The Content Engine and Escenic assembly tool have been installed as described in the **Escenic Content Engine Installation Guide** and are in working order.
- You have the required plug-in distribution file **`revision-history-dist-1.0.0.1.zip`**.
- Your Content Engine installations have access to a working CouchDB database. For advice on installing CouchDB please see **section 2.2**.

2.1 Conventions

The instructions in the following section assume that you have a standard Content Engine installation, as described in the **Escenic Content Engine Installation Guide** for releases 5.3.0 and above. *escenic-home* is used to refer to the **`/opt/escenic`** folder under which both the Content Engine itself and all plug-ins are installed.

The Content Engine and the software it depends on may be installed on one or several host machines depending on the type of installation required. In order to unambiguously identify the machines on which various installation actions must be carried out, the **Escenic Content Engine Installation Guide** defines a set of special host names that are used throughout the manual.

Some of these names are also used here:

assembly-host
The machine used to assemble the various Content Engine components into an enterprise archive or .EAR file.

database-host
The machine used to host the database server.

engine-host
A machine used to host an application servers and Content Engine instance.

editorial-host
An **engine-host** that is used solely for (internal) editorial purposes. All Content Studio clients communicate with an **editorial host**.

In addition, the following Revision History-specific host name is used:

version-storage-host
The machine used to host the external document store in which all changes are stored.

The host names always appear in a bold typeface. If you are installing everything on one host you can, of course, ignore them: you can just do everything on the same machine. If you are creating a larger multi-host installation, then they should help ensure that you do things in the right places.

2.2 Install CouchDB

CouchDB is an open source document database server. This section describe how to install it on a Debian-based Linux distribution such as Ubuntu. For further information about CouchDB, see http://couchdb.apache.org/index.html.

On your **version-storage-host**:

1. Log in as **`root`**.
2. Install CouchDB using **`apt-get`**:

   ```
   # apt-get install couchdb
   ```

   The CouchDB version packaged with your version of Ubuntu may be old (Ubuntu 10.04 LTS comes with CouchDB 0.10). To get a newer version of CouchDB, add the backports repository to your list of package repositories.

   After you have installed CouchDB, you may want to configure it. To do this, edit the file **`local.ini`**, which is usually located in **`/etc/couchdb`**. You may need to change the **`bind_address`** to point to your **version-storage-host**.
3. To verify that the **version-storage-host** is set up correctly, open a browser and enter the following URL:

   ```
   http://host-name:5984/
   ```

   where *host-name* is the host name or IP address of your **version-storage-host**.

   This should result in the following output:

   ```
   {"couchdb":"Welcome","version":"1.0.1"}
   ```

   The version number will depend on the version of CouchDB you have installed.

2.3 Install Revision History

This section describes how to install the Revision History plug-in itself.

On each of your **editorial-host**s:

1. Log in as **`escenic`**.

2. Open **`/opt/tomcat/conf/context.xml`** for editing and insert the following **`Environment`** element as a child of the root **`Context`** element:

   ```
   <Context>
     ...
     <Environment name="escenic/revision-history-webservice" override="false"
     type="java.lang.String" value="http://host:port/revision-history/"/>
     ...
   </Context>
   ```

   where:

   - *host* is the host name or IP address of the host on which the revision history web service is to run.
   - *port* is the port number on which the web service is to listen.

   This setting determines which revision history service is accessed by Content Studio when the **Revision history** option is selected. Basically, you should use the same *host* and *port* values as you use for the **`property.com.escenic.client.webservice.url`** setting in **`com/escenic/webstart/StudioConfig.properties`**. If your installation uses SSL for communication between Content Studio and the Content Engine, then you should use it here too: that is, you should use the **`https:`** protocol prefix in the URL, rather than **`http:`**.

On your **assembly-host**:

1. Log in as **`escenic`**
2. Download the Revision History distribution from the Escenic Technet web site (http://technet.escenic.com). If you have a multi-host installation with shared folders as described in the **Escenic Content Engine Installation Guide**, then it is a good idea to download the distribution to your shared **`/mnt/download`** folder:

   ```
   $ cd /mnt/download
   $ wget http://user:password@technet.escenic.com/downloads/PATH??/revision-history-
   dist-1.0.0.1.zip
   ```

   Otherwise, download it to some temporary location of your choice.
3. If the folder **`/opt/escenic/engine/plugins`** does not already exist, create it:

   ```
   $ mkdir /opt/escenic/engine/plugins
   ```

4. Unpack the Revision History distribution file:

   ```
   $ cd /opt/escenic/engine/plugins
   $ unzip /mnt/download/revision-history-dist-1.0.0.1.zip
   ```

   This will result in the creation of an **`/opt/escenic/engine/plugins/revision-history`** folder.

On your **database-host**:

1. Log in as **`escenic`**.
2. You now need to run the Revision History database scripts. If you have a single-host installation, then you will find them in **`/opt/escenic/`**

engine/plugins/revision-history/misc/database/mysql. If you have a multi-host installation with shared folders, then you have direct access to the distribution file and can just unpack it to a temporary location as follows:

```
$ cd /tmp
$ unzip /mnt/download/revision-history-dist-1.0.0.1.zip
```

You will then find the scripts in **/tmp/revision-history/misc/database/mysql**. If you don't have a shared folder set-up then you can either download the package from Escenic Technet again and unpack it, or copy the scripts over from your **assembly-host**.

Once you have located the scripts, you can run them as follows:

```
$ cd script-folder
$ for el in tables.sql ; do
> mysql -u user -ppassword db-name < $el
> done;
```

where *script-folder* is the path of the folder containing the scripts.

On your **assembly-host**:

1. Log in as **escenic**.
2. Run the **ece** script to re-assemble your Content Engine applications:

   ```
   $ ece assemble
   ```

   This generates an EAR file (**/var/cache/escenic/engine.ear**) that you can deploy on all your **engine-host**s.

On each **engine-host**:

1. Log in as **escenic**.
2. If you have a single-host installation, then skip this step.

   Copy **/var/cache/escenic/engine.ear** from the **assembly-host**. If you have installed an SSH server on the **assembly-host** and SSH clients on your **engine-host**s, then you can do this as follows:

   ```
   $ scp -r escenic@assembly-host-ip-address:/var/cache/escenic/engine.ear /var/cache/escenic/
   ```

   where *assembly-host-ip-address* is the host name or IP address of your **assembly-host**.
3. Deploy the EAR file and restart the Content Engine by entering:

   ```
   $ ece deploy
   $ ece restart
   ```

2.4 Verify The Installation

To verify the status of the Revision History plug-in, open the Escenic Admin web application (usually located at **http://*server*/admin**) and click on **View**

installed plugins. The status of all currently installed plug-ins is shown here, and indicated as follows:

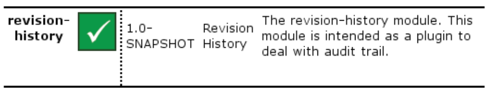
✓

The plug-in is correctly installed.

✗

The plug-in is not correctly installed.

So if the Revision History plug-in is correctly installed, you should see something like this in the displayed plug-in list:



revision-history	✓	1.0-SNAPSHOT	Revision History	The revision-history module. This module is intended as a plugin to deal with audit trail.

You now know that the plug-in is installed, but it still needs to be configured.

2.5 Configure The Revision History Plug-in

The Revision History plug-in will not work until you have configured it correctly by modifying configuration parameters in one or more of your **configuration layers**, as described below. For background information about configuration layers and how they work, refer to the **Escenic Content Engine Server Administration Guide**.

In these instructions, the placeholder *configuration-root* is used to represent the root folder of the configuration layer in which you are working. In the case of the **common** configuration layer, this will be **`/etc/escenic/engine/common`** if you have a standard Content Engine installation as described in the **Escenic Content Engine Installation Guide**. If you have a single-host installation or multi-host installation with configuration layers on a shared (i.e NFS) file system, then you will only need to carry out the changes described below once. Otherwise you will have to repeat them on each of your **engine host**s.

The general procedure for configuring the Revision History plug-in is:

1. Log in as **`escenic`**.
2. Create *configuration-root*/**`com/escenic/revision`** folders by copying them from **`/opt/escenic/engine/plugins/revision-history/misc/siteconfig/com/escenic`** as follows:

```
$ cp -r /opt/escenic/engine/plugins/revision-history/misc/siteconfig/com/escenic/
* configuration-root/com/escenic
```

3. Edit the **`.properties`** file in the folder you have created as described below.

2.5.1 RevisionManager.properties

You must configure the **`RevisionManager`** service on your **`editorial-host`**s by editing *configuration-root* **`/com/escenic/revision/scm/RevisionManager.properties`**. You must set the following properties:

couchdbURI
Set this to `http://`*host-name*`:`*port*`/`*database-name*, where:

- *host-name* is the host name or IP address of the **version-storage-host**
- *port* is the port used by CouchDB. The default is `5984`.
- *database-name* is the name you want to use for your revision database. The name must not contain any slashes (`/`), underscores (`_`) or upper case characters.

user
If you need to specify a user name and password to access the external document store, then you can use this property to specify the user name.

password
If you need to specify a user name and password to access the external document store, then you can use this property to specify the password.

serviceEnabled
Set this to `true`. This property is required as it defaults to `false`.

2.5.2 AuditPostTransactionFilter.properties

You must check the configuration of the `AuditPostTransactionFilter` service on your `engine-host`s. This service is configured in *configuration-root* `/com/escenic/revision/audit/AuditPostTransactionFilter.properties`. You need to verify that the `errorFileDirectory` property is correctly set. This property must point to a folder on the server that exists and is writable. If this is not the case, then you must either create/modify the folder or change the property setting.

By default, `errorFileDirectory` points to the following folder:

```
/var/lib/escenic/${ece_instance "default"}/failed-revisions/
```

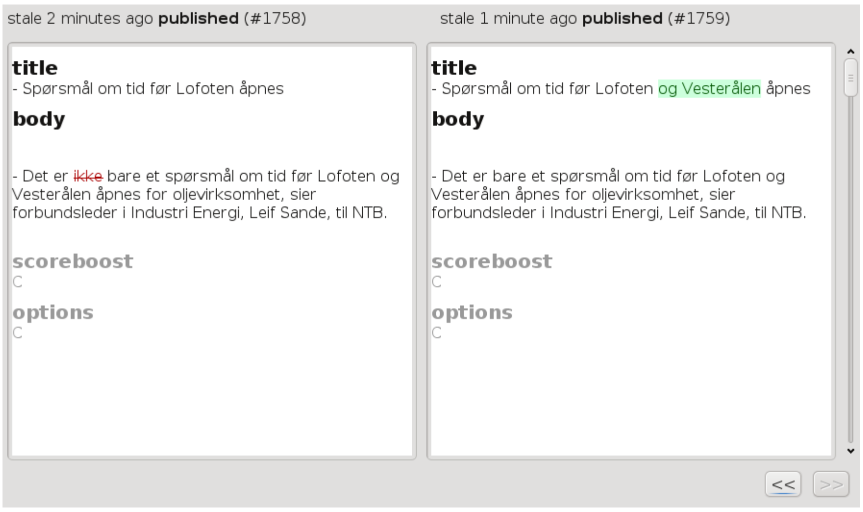
3 Using The Plug-in

Once you have installed and correctly configured the plug-in, all changes made to content items are automatically recorded. You can then begin to use the **Revision History** function added to Content Studio.

3.1 Viewing Differences between Revisions

To view the differences between different revisions of a content item:

1. Open content item.
2. Select **View** > **Revision history**. A **Revision history** dialog is then displayed. Here is a **Revision history** dialog showing changes made to a content item:



stale 2 minutes ago **published** (#1758)

title
- Spørsmål om tid før Lofoten åpnes

body

- Det er ~~ikke~~ bare et spørsmål om tid før Lofoten og Vesterålen åpnes for oljevirksomhet, sier forbundsleder i Industri Energi, Leif Sande, til NTB.

scoreboost
C

options
C

stale 1 minute ago **published** (#1759)

title
- Spørsmål om tid før Lofoten og Vesterålen åpnes

body

- Det er bare et spørsmål om tid før Lofoten og Vesterålen åpnes for oljevirksomhet, sier forbundsleder i Industri Energi, Leif Sande, til NTB.

scoreboost
C

options
C

The dialog shows differences between the current version of the content item (on the right) and the previous version (on the left). You can view the differences between earlier versions by clicking on the **<<** button: each click on the button moves the view back one version. You can move forward again by clicking on the **>>** button. Information about the versions you are looking at and when they were created is displayed at the top of the dialog.

Display conventions

Differences are shown by means of the following conventions:

- Unchanged fields are displayed in gray in both panes, changed fields are displayed in black.
- Removed fields and text are displayed in red and struck through in the left-hand pane.
- New fields and text are displayed with a green background in the right-hand pane.

- The states (draft, published, etc) of the compared versions are shown above the panes along with the revision date/time.

The following differences are not shown:

- Changes to binary objects such as images, sound and video clips, Word and PDF attachments
- Changes to relations

Formatting is not shown **Revision history** dialog, and although formatting changes are recorded by the plug-in as changes, they are not shown or highlighted in any way. Consequently, if the only difference between two versions is a formatting change, then they will appear to be identical when viewed in the **Revision history** dialog.

4 Administration

Once the plug-in is installed and configured correctly, it can be monitored using the services described below.

4.1 Component Browser Services

The Revision History plug-in provides the component browser services described in the following sections. They can be found under **`/com/escenic/revision/`** in the **`escenic-admin`** application's component browser. The component browser is described in the **Escenic Content Engine Server Administration Guide**.

4.1.1 /com/escenic/revision/audit/AuditPostTransactionFilter

The **`AuditPostTransactionFilter`** is a **`neo.xredsys.api.PostTransactionFilter`** object (see javadoc for details). It takes all the content submitted in a transaction and stores it in a queue implemented as a table in the Content Engine database. This filter should be running on every **`engine-host`**. On a working **`engine-host`**, the following properties should return **`true`**:

`serviceEnabled`
: This service is allowed to start.

`serviceRunning`
: This service is running correctly.

Should this service **not** be running on an **`engine-host`**, then any revisions made on that host will not be recorded in the revision history.

The property **`stateMessage`** contains a message describing the service's current state.

4.1.2 /com/escenic/revision/audit/AuditTrailMessageReceiver

This service takes content from the queue populated by the **`AuditPostTransactionFilter`**, and submits it to the external document store. Once a revision has been successfully submitted to the external document store, it deletes the item from the queue, and continues with the next one. This service should only run on one **`editorial-host`** at a time.

The service should only be enabled on **`editorial-host`**s (that is, it should not be enabled on **`engine-host`**s that are solely used for presentation purposes). It can, however, be enabled on **all** **`editorial-host`**s, as the service itself ensures that only one instance is running at a time, and will fail over to another host if necessary. On the host where the service is actually running, the following properties will return **`true`**:

serviceEnabled
: This service is allowed to start.

serviceRunning
: This service is running correctly.

On a host where the service is enabled but not currently running, the following properties will return **true**:

serviceEnabled
: This service is allowed to start.

serviceIdle
: This service is waiting to start. It will try to start every **validateInterval** minutes.

The property **stateMessage** contains a message describing the service's current state.

4.1.3 /com/escenic/revision/scm/RevisionManager

This service is responsible for the communication between **AuditTrailMessageReceiver** and the external document store. It is also responsible for supplying Content Studio with the versions displayed in the **Revision history** dialog. This is done in cooperation with **RevisionHistoryLinkHeaderPlugin** (see **section 4.1.4**).

This service must be running on every **editorial-host**. On a working **editorial-host** the following properties should return **true**:

serviceEnabled
: This service is allowed to start.

serviceRunning
: This service is running correctly.

The property **stateMessage** contains a message describing the service's current state.

4.1.4 /com/escenic/revision/webservice/RevisionHistoryLinkHeaderPlugin

This service supplies Content Studio with requested versions. This is done in cooperation with **RevisionManager** (see **section 4.1.3**).

This service must be running on every **editorial-host**. On a working **editorial-host** the following properties should return **true**:

serviceEnabled
: This service is allowed to start.

serviceRunning
: This service is running correctly.

The property **`stateMessage`** contains a message describing the service's current state.

4.2 Known Failure Modes

This section describes the types of failure that are known to occur with the Revision History plug-in.

4.2.1 External Document Store

If the external document store becomes unavailable, all changes are queued in the Content Engine database. When the external document store is available again, the queued changes are stored normal service is resumed. Changes are therefore never lost. While the document store is unavailable, however, the Content Studio **Revision history** dialog will not work correctly (it will be empty), since the stored versions cannot be accessed. Once the is document store is available again, the dialog will again work as expected.

If the external document store is lost due to some catastrophic failure, the only means of recovery is to restore it from a backup. There is no other way to recreate content item history.

4.2.2 AuditPostTransactionFilter

If this filter is not running or is incorrectly configured on a particular **engine-host** host, then any additions or changes made on that host will not be recorded in the revision history.

4.2.3 RevisionHistoryLinkHeaderPlugin

If this plug-in is not running or is incorrectly configured on an **editorial-host** then the **View** > **Revision history** becomes unavailable, or the content of the **Revision history** dialog becomes empty, depending on the severity of the misconfiguration. No content is lost if this is the case.

4.3 Logging

The Revision History plug-in logs messages in the category **`com.escenic.revision`** or one of its sub-categories (**`com.escenic.revision.audit.AuditPostTransactionFilter`**, for example). For general information about the Content Engine logging system, see the **Escenic Content Engine Server Administration Guide**.

4.4 Operational tasks and hints

This section describes the typical operational task that needs to be performed and some hints to other operational concerns.

4.4.1 Performing backups

Backing up your **`version-storage-host`** running CouchDB can be done in at least two different ways. Please refer to the CouchDB site to find detailed information. We will give a small overview over the two most common ways.

4.4.1.1 File system backup

For all platforms, locate your database, configuration, and log files and perform a filesystem copy. Be careful to preserve file permissions, too. Archive these files to wherever you want-- ideally on a different machine in a different physical location -- with appropriate security limiting access. For example, here are the directories to backup for a CouchDB install on Ubuntu:

- Configuration: /usr/local/etc/couchdb/
- Database files: /usr/local/var/lib/couchdb/
- Logs: /usr/local/var/log/couchdb/

And to backup for a CouchDB install on Windows where %couch% is where you've installed or unpacked CouchDB:

- Configuration: %couch%\etc\couchdb\
- Database files: %couch%\var\lib\couchdb\
- Logs: %couch%\var\log\couchdb\

Before CouchDB 1.0 intermediate releases can have incompatible database file formats.

Please see How to make filesystem backups for more information about the procedure.

4.4.1.2 Replication

To make a backup of the changed content object you can set up a second **`version-storage-host`** and do replication from the primary to the secondary one. The configuration of the replication is done in the build in administrative client of CouchDB, Futon. Here you can set up both the source and target host and database. The replication can be done in both a push, and a pull configuration. It can be configured to be a 'one time job' or it can be continuous.

The configuration for replication does not survive a restart, and needs to be set up each time the server that is to do the replication is restarted.

Please see How to replicate a database for more information about the procedure.

4.4.2 Disk space requirements

The disk space requirements for CouchDB on your **`version-storage-host`** will vary with the number of revisions and the type of content you have. We have seen that for an article that contains about 100 Kb of text, it will require about 4 Kb of disk space for each revision saved. This will depend on the nature of the text saved, as CouchDB applies some compression of the content stored.

4.4.3 Memory requirements

The memory requirements for CouchDB are quite modest. The important factor is to have a fast and caching file system on your **`version-storage-host`**.

5 Third Party Licenses

This chapter contains licensing information regarding third party products used in the Revision History plug-in.