

How to use version control in Joomla 3

Written by [Brad Markle](#)

Views: 6,484

Published: Dec 13, 2013

Comments: [5](#)

In this tutorial:

Joomla 3.0 / Joomla 3.1

Version control was not introduced until [Joomla 3.2](#).

Joomla 3.2


A new feature in Joomla 3.2 is version control. This tool allows you to save previous versions of your articles, and also allows you to revert to any previous version at any time. In this tutorial, we will review how to use the new version control system.

Article Revisions:

Getting started: Editing an article

Site Name Goes Here

Home



Getting Started

It's easy to get started creating your website. Knowing some of the basics will help.

What is a Content Management System?

A content management system is software that allows you to create and manage webpages easily by separating the creation of your content from the mechanics required to present it on the web.

In this site, the content is stored in a database. The look and feel are created by a *template*. Joomla! brings together the template and your content to create web pages.

Logging in

To login to your site use the user name and password that were created as part of the installation process. Once logged-in you will be able to create and edit articles and modify some settings.

Latest Articles

- [Getting Started](#)

Login Form

☐ Remember Me

[Create an account >](#)

[Forgot your username?](#)

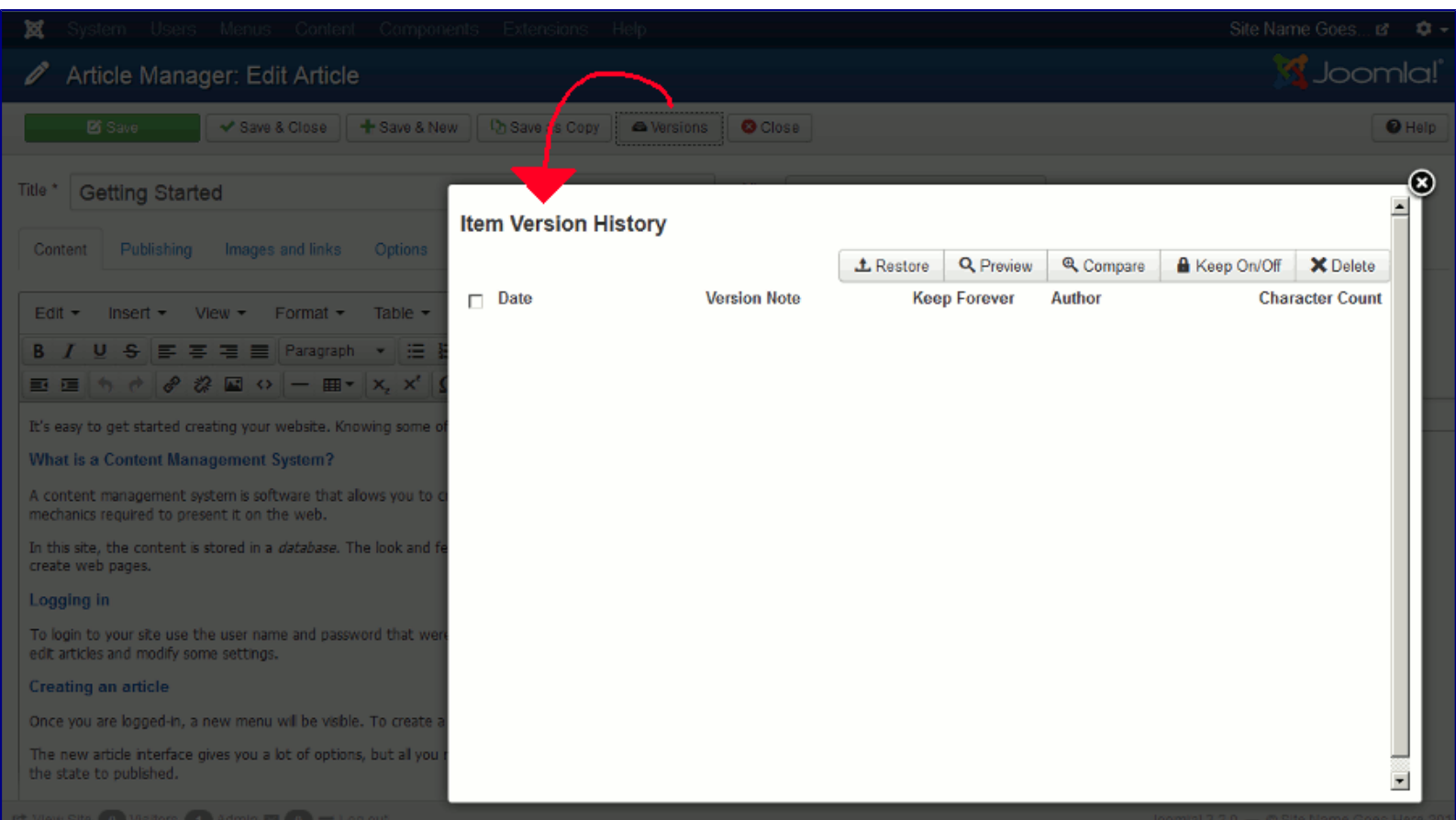
[Forgot your password?](#)

When you [install Joomla 3.2](#), the version component is setup and enabled by default. In our testing, we installed Joomla 3.2 with the [Default English \(GB\) Sample Data](#). As you

can see to the right, our homepage shows the *Getting Started* article, and this is the article that we will be working with in this tutorial.

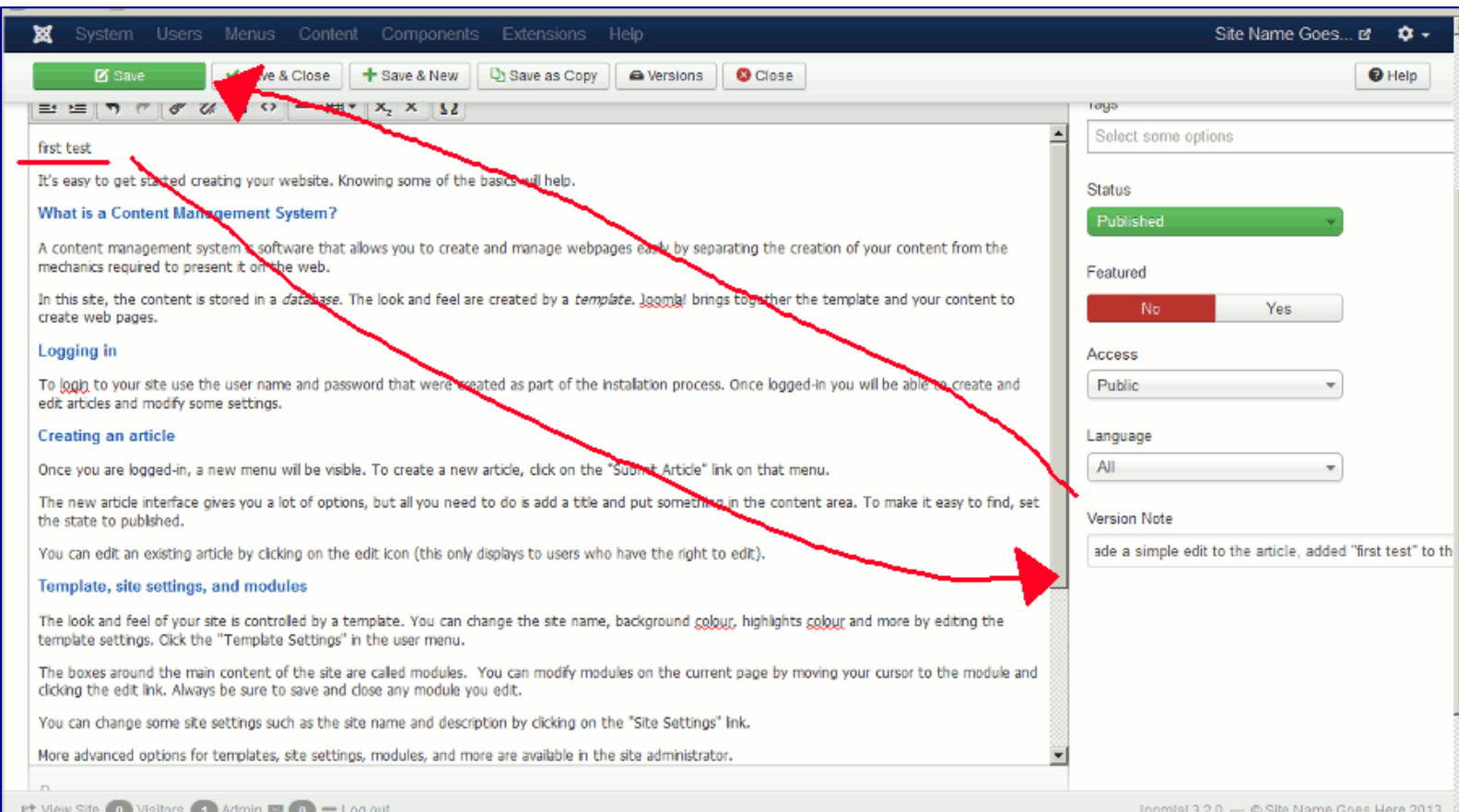
The first thing we're going to do is edit the *Getting Started* article. To do this, [log into your Joomla dashboard](#) >> **Content** >> **Article Manager** >> and then click on **Getting Started**.

New article, no revision history yet



In the top menu, you will see a button that says **Versions**. If you click it now, it will open the **Item Version History** window. It is blank at the moment, because we haven't made any edits to the article just yet.

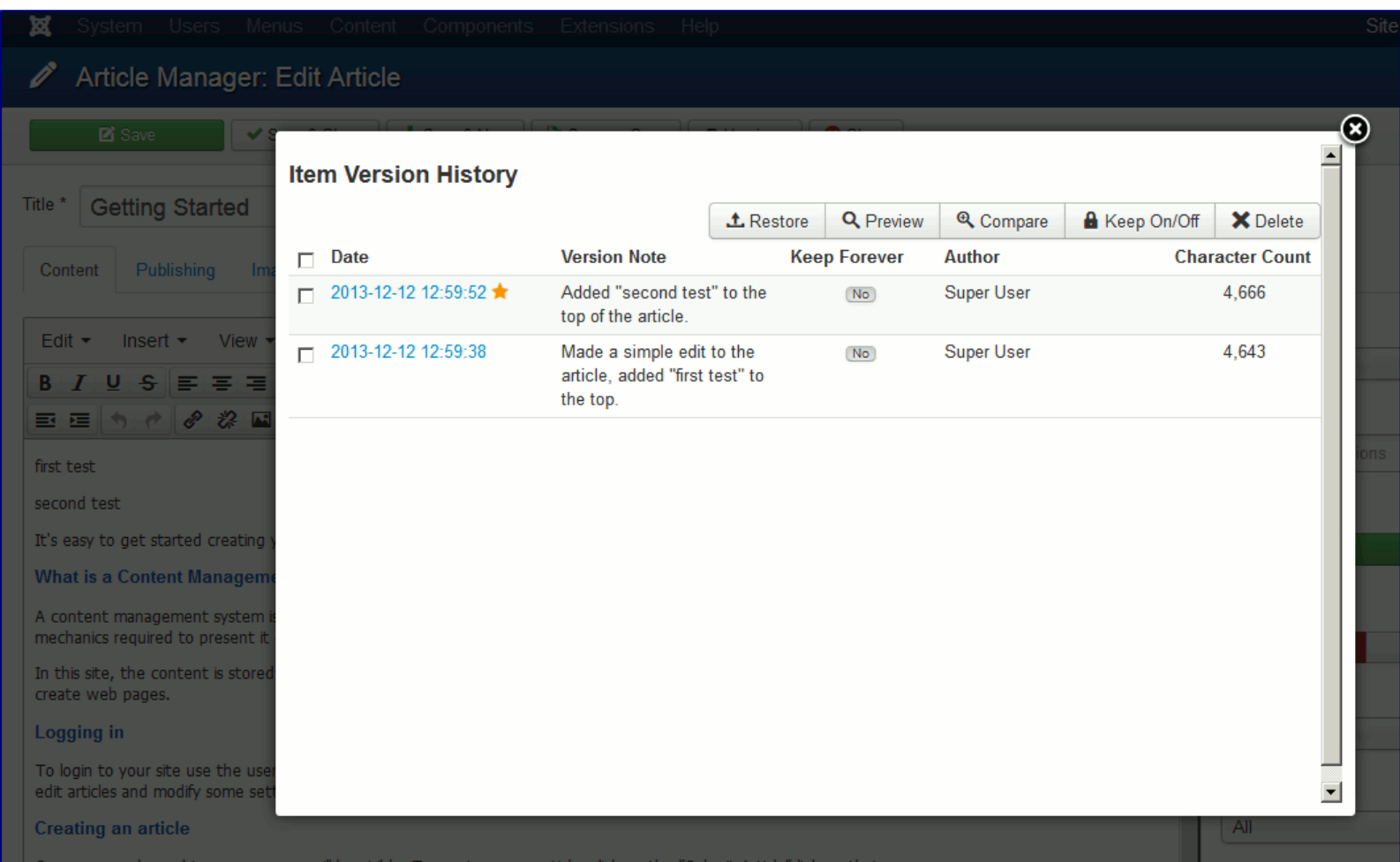
Making our first edit



In our testing, we added *first test* at the top of the article. In the right sidebar, under **Version Note**, we typed the following note to remind us of the changes we made: *Made a simple edit to the article, added "first test" to the top*. When we finished editing the article and typing a *Version Note*, we clicked **Save** in the top menu.

For the sake of testing, we made another edit. We added *second test* at the top of the article, and saved it with a similar *Version Note*.

How to view previous versions of an article



The screenshot shows the Joomla! Article Manager interface. The main window displays the 'Edit Article' page for the article titled 'Getting Started'. A modal dialog box titled 'Item Version History' is open, showing a table of previous versions of the article. The table has five columns: 'Date', 'Version Note', 'Keep Forever', 'Author', and 'Character Count'. There are two versions listed, both created by 'Super User' on December 12, 2013. The first version, at 12:59:52, has a character count of 4,666 and a note about adding 'second test' to the top. The second version, at 12:59:38, has a character count of 4,643 and a note about adding 'first test' to the top. The dialog box also includes buttons for 'Restore', 'Preview', 'Compare', 'Keep On/Off', and 'Delete'.

<input type="checkbox"/>	Date	Version Note	Keep Forever	Author	Character Count
<input type="checkbox"/>	2013-12-12 12:59:52 ★	Added "second test" to the top of the article.	No	Super User	4,666
<input type="checkbox"/>	2013-12-12 12:59:38	Made a simple edit to the article, added "first test" to the top.	No	Super User	4,643

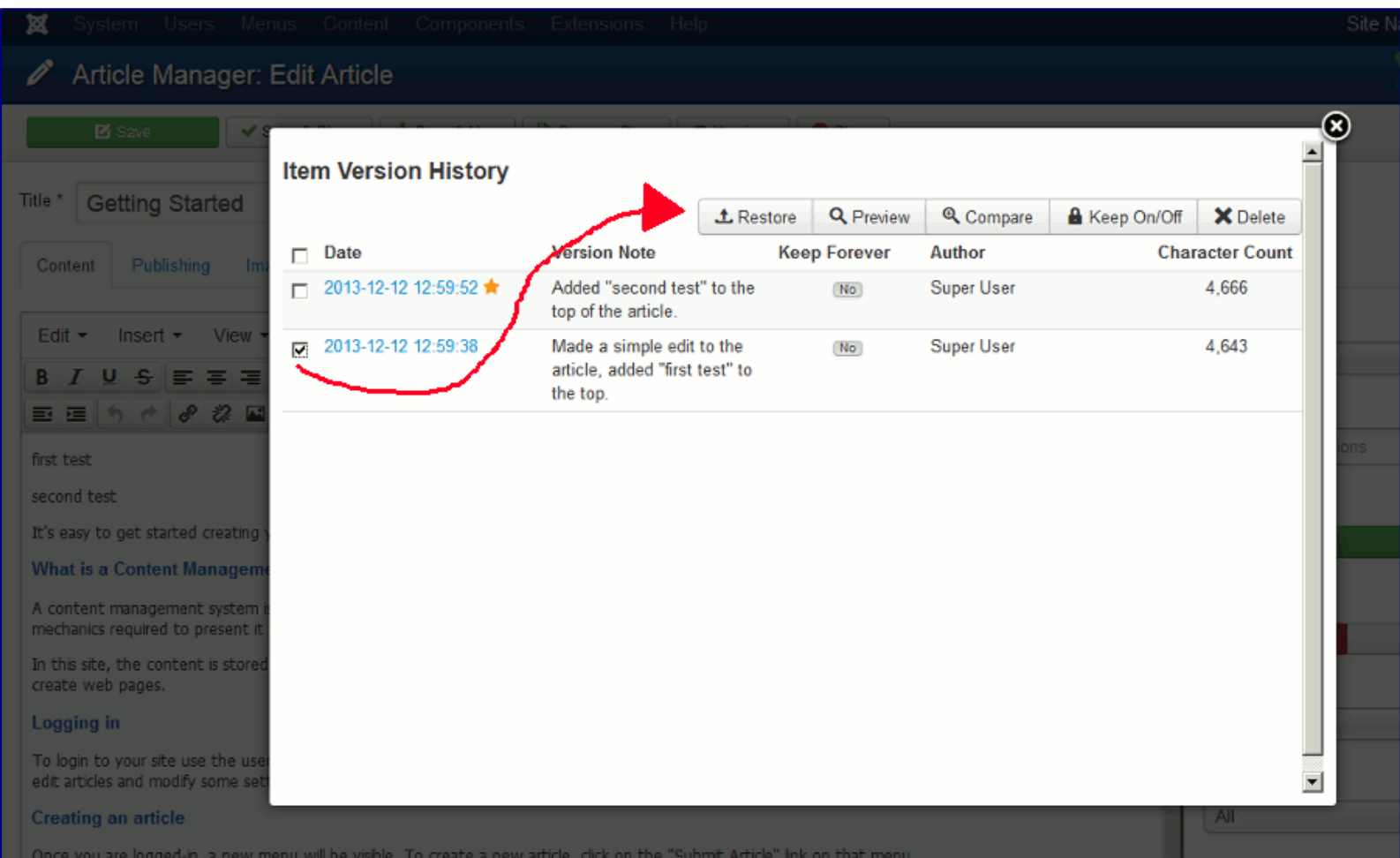
To view previous versions of an article, click the **Versions** button towards the top of the *Edit Article* page. The *Item Version History* page will show, and you will be able to see a listing of the previous versions of the article.

The screenshot shows the Joomla! Article Manager interface. On the left, the 'Item Version History' window is open, displaying a table of article versions. A red arrow points from the date '2013-12-12 12:59:38' in the table to the preview window on the right. The preview window shows the article content as it appeared at that time, with the 'Intro text' field containing the text 'first test'.

Field	Value
ID	1
Title	Getting Started
Alias	getting-started
Intro text	first test It's easy to get started creating your website. Knowing some of the basics will help. What is a Content Management System? A content management system is software that allows you to create and manage webpages by separating the creation of your content from the mechanics required to present it on the v In this site, the content is stored in a <i>database</i> . The look and feel are created by a <i>template</i> . Joomla! brings together the template and your content to create web pages. Logging in To login to your site use the user name and password that were created as part of the install process. Once logged-in you will be able to create and edit articles and modify some setting Creating an article Once you are logged-in, a new menu will be visible. To create a new article, click on the "Su Article" link on that menu. The new article interface gives you a lot of options, but all you need to do is add a title and p something in the content area. To make it easy to find, set the state to published.

To see what a specific version looked like, click on the version's **Date**. Another window will open, showing you a rough version of the article. As you can see in the screenshot to right, we clicked on the *"first test"* edit we made, and the change can be seen in the *Intro text*.

How to restore a previous version



The screenshot shows the Joomla! Article Manager interface. A modal window titled "Item Version History" is open, displaying a table of article versions. A red arrow points from the "Restore" button in the top right of the modal to the checkbox of the selected version (2013-12-12 12:59:38).

<input type="checkbox"/>	Date	Version Note	Keep Forever	Author	Character Count
<input type="checkbox"/>	2013-12-12 12:59:52 ★	Added "second test" to the top of the article.	No	Super User	4,666
<input checked="" type="checkbox"/>	2013-12-12 12:59:38	Made a simple edit to the article, added "first test" to the top.	No	Super User	4,643

Restoring a previous version of an article is easy! On the **Item Version History** screen, after finding the version you want to restore, click the checkbox next to the article version and then click the **Restore** button. After the article restoration has completed, you'll see a message similar to the following:

Prior version successfully restored. Saved on 2013-12-12 12:59:38 Made a simple edit to the article, added "first test" to the top..

How to compare different versions of an article

Sometimes it may be hard to tell exactly what changes were made based upon the *Version Notes* that you enter. However, using the **Compare** feature, Joomla will show you a side by side comparison of two different versions of your article.

The screenshot shows the Joomla! administration interface. The top menu includes System, Users, Menus, Content, Components, Extensions, and Help. The main toolbar has buttons for Save, Save & Close, Save & New, Save as Copy, Versions, and Close. The 'Item Version History' window is open, displaying a table with columns: Date, Version Note, Keep Forever, Author, and Character Count. Two versions are selected with checkboxes: 2013-12-12 12:59:52 and 2013-12-12 12:59:38. The 'Compare' button is highlighted. Below, the 'Compare View' window shows a side-by-side comparison of the two versions. The 'Changes' column highlights additions in green (e.g., 'second test') and removals in red (e.g., 'first test').

Field	Version 1 (2013-12-12 12:59:52)	Version 2 (2013-12-12 12:59:38)	Changes
ID	1	1	1
Title	Getting Started	Getting Started	Getting Started
Alias	getting-started	getting-started	getting-started
Intro text	first test It's easy to get started creating your website. Knowing some of the basics will help. What is a Content Management System? A content management system is software that allows you to create and manage webpages easily by separating the creation of your content from the mechanics required to present it on the	first test second test It's easy to get started creating your website. Knowing some of the basics will help. What is a Content Management System? A content management system is software that allows you to create and manage webpages easily by separating the creation of your content from the	first test second test It's easy to get started creating your website. Knowing some of the basics will help. What is a Content Management System? A content management system is software that allows you to create and manage webpages easily by separating the creation of your content from the mechanics required to present it on the web.

To compare two versions: On the **Item Version History** screen, check two versions of your article and then click the **Compare** button. A new window will open showing side by side comparisons of the versions you chose. Under the **Changes** column, text added will be highlighted green and text removed will be highlighted red.

How to limit the number of versions saved

If you edit articles frequently, the number of different versions saved can grow quite large over time. This can make it difficult to find the version you're looking for, for example, if you have 100 or so versions of the same article saved.

Maximum Versions saved is 10 by default

The default value for the maximum versions to save per article is 10. When you save an article and the max version save count has been reached, Joomla will delete the oldest revision to make room for the new one. You can easily change how many revisions are saved by adjusting the **Maximum Versions** setting. To do so:

Changing the Maximum Versions saved setting

In the top menu of your Joomla admin dashboard, click **Content >> Article Manager >> Options** (at the top right of the page). Then, click the **Editing Layout** tab, change the **Maximum Versions** setting, and finally click **Save & Close**.

How to prevent a previous version from being deleted

If you don't want a particular version of an article deleted when the *Maximum Versions* limit has been reached, you can lock the version to prevent it from being removed. To lock a

version, on the **Item Version History** screen, simply click **No** under **Keep Forever**. This will change the value to **Yes**, and prevent the version from being removed.

<http://www.inmotionhosting.com/support/edu/joomla-3/writing-articles/version-control>

Tutorial : How to enable content versioning in Joomla site

[Joomla Tutorials](#)

13 May 2014

Hate it when the article you are working on is lost or overwritten or that one time your content gets “mysteriously” modified, and you have no clue who did it?

I know it's rather frustrating and annoying to a certain extent confronting those situations (right?!). This is when Joomla content versioning comes in handy.



Tutorial : How to enable content versioning in Joomla site

So, what is content versioning?

Generally it's a pretty simple process that creates an iteration of the content you're working on every time you hit save.

Alex Gilgenbach - [How version control can save the day](#)

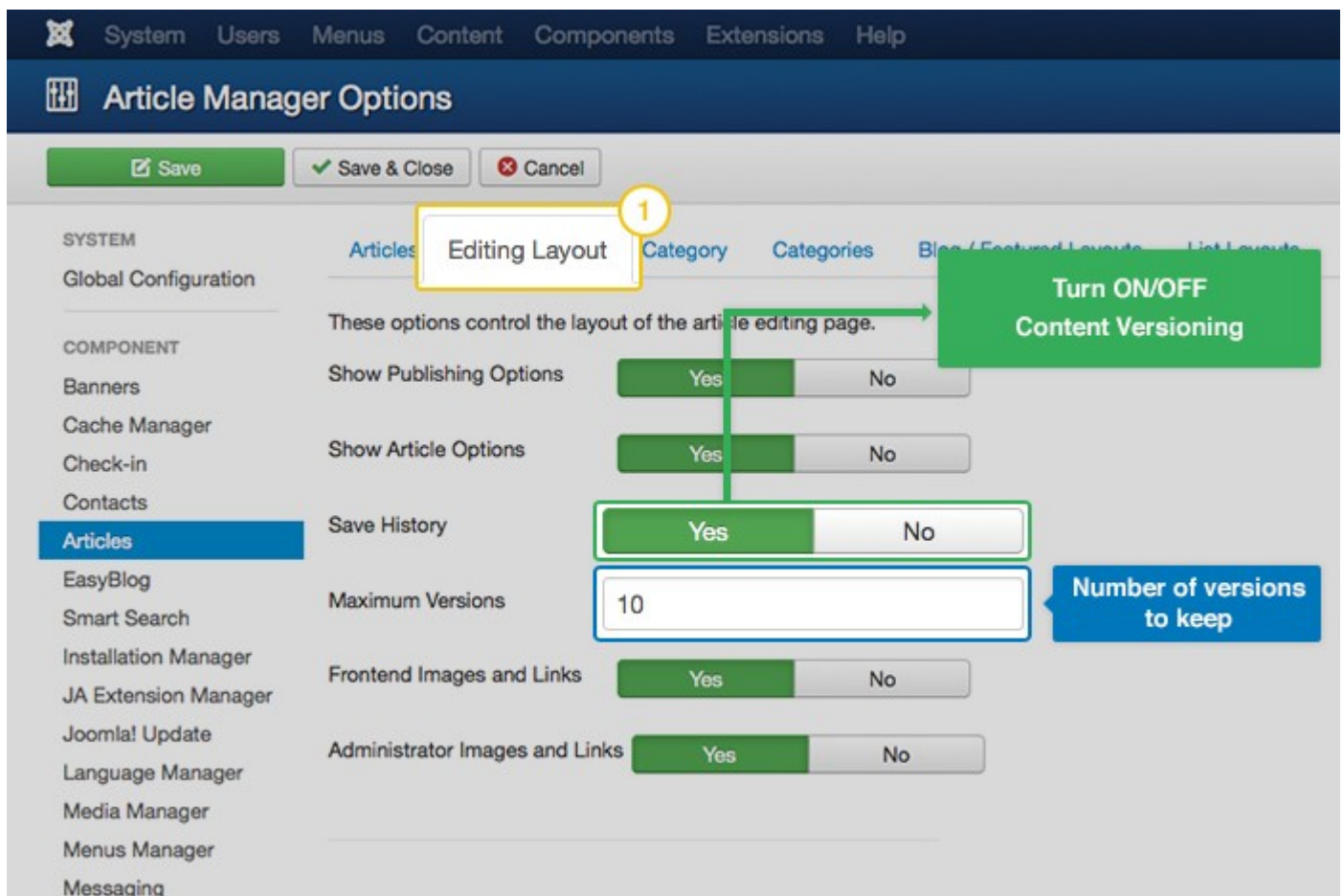
Content versioning has been incorporated into Joomla core since the release of Joomla 3.2. It also provides an API to integrate with other components (cool?!). Content versioning can help when:

- HTML formatting is deleted by an editor or Joomla's text filtering in global config.
- You make a mistake and accidentally hit save; or accidentally deleted an article.
- You want to ensure you have backups each time you save for easily recover.

Tabela prefixo_ucm_history.

Enough for the wordings, let see how and what we can do with Joomla content versioning.

How to enable and use it in Joomla



How to enable version control for content in Joomla

You can find content versioning under 'Editing Layout' tab in articles component options. Turn On 'Save History', and input the number of 'versions' to keep. Please note that the more versions you keep, the more data space it requires.

Now, the content version control feature is enabled. Create the first version of Joomla article.

Article Manager: Edit Article Joomla!

Save Save & Close Save & New Save as Copy Versions Click [Versions] to view previous saves

Save your changes

Category *
--- Sample Category 2

Tags
Select some options

Status
Published

Featured
Yes No

Access
Public

Language
English (UK)

Version Note
Add Pictures

Add change note for the version you save

Article Image Page Break Read More Toggle editor

Joomla article editing with version control enabled

With the first save, a version of your Joomla article is created. After you change the content and hit save, a copy of old version will be saved, and you will work with the new one.

Article Manager: Edit Article Joomla!

Save Save & Close Save & New Save as Copy Versions Close Help

Message
Article successfully saved

Title * Could Apple

Content ☐ Publish **Date**

		<input type="checkbox"/>	Date	Note	Keep Forever	Author	Character Count
2014-05-13 09:20:49	★	<input type="checkbox"/>	2014-05-13 09:20:49	option to picture	(No)	Super User	4,343
2014-05-13 09:20:19		<input type="checkbox"/>	2014-05-13 09:20:19	ictures	(No)	Super User	4,449

Currently edited version is marked with a star

Current version is marked with a star

You can view the previous history for the article by clicking on 'Versions'. The list of saved versions will appear. The current version that you are working on is marked with a star.

There are many actions you can do with versions of your Joomla articles:

Keep Forever option

Article Manager: Edit Article Joomla!

Save Save & Close Save & New Save as Copy Versions Close Help

Message Article successfully saved

Message Successfully changed the keep forever value for 1 history version

Item Version

Click to allow this version to be deleted automatically according to the delete schedule.

Keep Forever Author

caption to picture Yes Super User

2014-05-13 09:20:19 Add Pictures No Super User 4,449

To keep a version forever, click on Keep Forever column to change status

Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.

1. Lorem ipsum dolor sit amet, consectetur adipiscing elit. 2. Aliquam tincidunt mauris eu sed nisl.

• Home • About • Clients

Keep Forever option

To keep a version forever, click the button on '*Keep Forever*' column. '*Keep Forever*' will allow/disallow the version to be deleted automatically according to delete schedule.

Compare two versions of an article

Check the boxes before the two versions you want to compare - click '*Compare*' button

Compare View

	Older Version	Newer Version
Field	Saved on 2014-05-13 09:35:25 Add Pictures	Saved on 2014-05-13 09:36:55 remove text
Intro text	<p>Donec eu libero sit amet quam egestas semper. Pellentesque habitant morbi tristique estibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Aenean ultricies mi vitae est. Mauris placerat eleifend leo. Quisque sit amet est et sapien ullamcorper pharetra.</p> <p>Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas, auris placerat eleifend leo.</p>	<p>Donec eu libero sit amet quam egestas semper. Pellentesque habitant morbi tristique estibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Aenean ultricies mi vitae est. Mauris placerat eleifend leo. Quisque sit amet est et sapien ullamcorper pharetra.</p> <p>Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas, auris placerat eleifend leo.</p>
Full text	<p>Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo. Quisque sit amet est et sapien ullamcorper pharetra. Vestibulum erat wisi, condimentum sed, commodo vitae, ornare sit amet, wisi. Aenean fermentum, elit eget tincidunt condimentum, eros ipsum rutrum orci, sagittis tempus lacus enim ac dui. Donec non enim in turpis pulvinar facilisis. Ut felis.</p> <p>Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.</p> <p>Vestibulum erat wisi</p> <ol style="list-style-type: none"> 1. Lorem ipsum dolor sit amet, consectetur adipiscing elit. 2. Aliquam tincidunt mauris eu risus. <p>Lorem ipsum dolor sit amet,</p>	<p>Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, tempor sit amet, ante. Donec eu libero sit amet quam egestas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo. Quisque sit amet est et sapien ullamcorper pharetra. Vestibulum erat wisi, condimentum sed, commodo vitae, ornare sit amet, wisi. Aenean fermentum, elit eget tincidunt condimentum, eros ipsum rutrum orci, sagittis tempus lacus enim ac dui. Donec non enim in turpis pulvinar facilisis. Ut felis.</p> <p>Vestibulum erat wisi</p> <ol style="list-style-type: none"> 1. Lorem ipsum dolor sit amet, consectetur adipiscing elit. 2. Aliquam tincidunt mauris eu risus. <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus magna. Cras in mi at felis aliquet congue. Ut a est eget ligula molestie gravida. Curabitur massa. Donec eleifend, libero at sagittis mollis, tellus est malesuada tellus, at luctus turpis</p>

Toggle between All Values / Changed Values

All Values Code

Compare two versions

A compare view will appear, for two selected versions.

Restore a previous version of a Joomla article

Select a version you want to restore, and click 'Restore' button to go back in time. And don't worry for the newer versions, they're still there when you need them.

Delete a version of Joomla article

Just hit the 'Delete' button with a selected version, it will go for good.

These are the basics of Joomla content versioning. If you have used it before, let us know your experience in the comment section below.

<https://www.joomlart.com/tutorials/joomla-tutorials/tutorial-how-to-enable-content-versioning-in-joomla-site>

Using Content History in your Component

De Joomla! Documentation

2

In version 3.2, Joomla added the ability to track content history (also called versions) for all core components. This allows you to view and restore from prior versions of an article, banner, category, and other content types. This feature can easily be added to third-party components. This tutorial shows you how to do this. We will illustrate this by adding content history to an example component.

Important Note: This tutorial assumes that the component uses a JTable sub-class for its "CRUD" (create/update/delete) operations. If the component does not use JTable then the simplest thing to do is to re-work it to use JTable's store() and delete() methods. If you do that, you will be able to use the methods described in this tutorial.

Índice

- [1 Set up Working Environment](#)
- [2 Add Rows to Content Types Table](#)
- [3 Add Component Level Options](#)
- [4 Update Content History During Table Save and Delete](#)
- [5 Add type alias to model](#)
- [6 Add Versions Button to Tool Bar](#)
- [7 Add Version Note Field to Form](#)
- [8 Add Labels to Pop-Up Windows](#)
- [9 Update Extension SQL Files](#)

Set up Working Environment

1. Install a new instance of Joomla version 3.2 on your local workstation.
2. Install the example component using the file com_joomprosubs_3.2.0.zip in the Extension Manager: Install screen. You can get the ZIP archive here:
https://github.com/joomla/jdoc-examples/raw/master/zip_archives/com_joomprosubs-3.2.0.zip.

At this point, you should be able to see the Joompro Subscriptions component in the Components menu in the back end of Joomla. This is the component before we have added content history to it.

Add Rows to Content Types Table

The first thing we need to do is to add two new rows into the #__content_types table. This

table stores information about the different tables for each content type. As of Joomla version 3.2, this table is used by the com_tags and com_contenthistory components to get information about each content type for which history is stored. The columns for #__content_types are as follows:

- **type_id**: auto-increment id number.
- **type_title**: descriptive title for this table.
- **type_alias**: <component name>.<type name>. For example: "com_content.article" or "com_content.category".
- **table**: JSON string that contains the name of the JTable class and other information about the table.
- **rules**: Not used as of Joomla version 3.2.
- **field_mappings**: Used by the com_tags component to map database columns from the component table to the ucm_content table.
- **router**: Optional location of the component's router, if any.
- **content_history_options**: JSON string used to store information for rendering the pop-up windows in the content history component. We will discuss this in detail later in this tutorial.

Our example component uses a table called #__joompro_subscriptions to store each subscription. In addition, it uses the standard Joomla categories. So we will add a row in #__content_types for the category table and one for the component table.

The row for the categories can be copied from the row for "Weblinks Category" (or any other category row). The only columns that need to be changed are type_title and type_alias. The other columns are the same as for com_weblinks.category.

The SQL statement for adding this row is as follows:

```
INSERT INTO `#__content_types` (`type_id`, `type_title`, `type_alias`, `table`,
`rules`, `field_mappings`, `router`, `content_history_options`)
VALUES
(NULL,
'Subscription Category',
'com_joomprosubs.category',
'{"special":
{"dbtable":"#__categories","key":"id","type":"Category","prefix":"JTable","config":"array()"},"common":
{"dbtable":"#__ucm_content","key":"ucm_id","type":"Corecontent","prefix":"JTable","config":"array()"}}',
'',
'{"common":
{"core_content_item_id":"id","core_title":"title","core_state":"published","core_alias":"alias","core_created_time":"created_time","core_modified_time":"modified_time","core_body":"description",
"core_hits":"hits","core_publish_up":"null","core_publish_down":"null","core_access":"access","core_params":"params","core_featured":"null",
"core_metadata":"metadata","core_language":"language","core_images":"null",
"core_urls":"null","core_version":"version","core_ordering":"null",
"core_metakey":"metakey","core_metadesc":"metadesc","core_catid":"parent_id",
"core_xreference":"null","asset_id":"asset_id"},"special":
{"parent_id":"parent_id","lft":"lft","rgt":"rgt","level":"level","path":"path","extension":"extension","note":"note"}}',
'WeblinksHelperRoute::getCategoryRoute',
```



```
'{"formFile":"administrator\\components\\com_categories\\models\\forms\\category.xml", "hideFields":
["asset_id","checked_out","checked_out_time","version","lft","rgt","level","path",
,"extension"], "ignoreChanges":["modified_user_id", "modified_time",
"checked_out", "checked_out_time", "version", "hits", "path"],"convertToInt":
["publish_up", "publish_down"], "displayLookup":
[{"sourceColumn":"created_user_id","targetTable":"#__users","targetColumn":"id",
"displayColumn":"name"},
{"sourceColumn":"access","targetTable":"#__viewlevels","targetColumn":"id","displayColumn":"title"},
{"sourceColumn":"modified_user_id","targetTable":"#__users","targetColumn":"id",
"displayColumn":"name"},
{"sourceColumn":"parent_id","targetTable":"#__categories","targetColumn":"id","displayColumn":"title"}]'});
```

For the actual subscriptions, we only need the following information in the content types table:

- **type_title:** Subscriptions
- **type_alias:** com_joomprosubs.subscription
- **table:**

```
{ "special":
{ "dbtable": "#__joompro_subscriptions", "key": "id", "type": "Subscription",
"prefix": "JoomprosubsTable" }}
```

This creates a JSON object as follows:

```
special:
dbtable = #__joompro_subscriptions
key = id
type = Subscription
prefix = JoomprosubsTable
```

The type_alias column is used by the com_contenthistory component to find the row for each component in the #__content_types table. The table column gives the com_contenthistory component the information it needs to work with the JTable class for each component. The None of the other columns need to be set for now. Note that we will come back to the content_history_options column later in the tutorial to improve the appearance of the pop-up window.

The SQL for adding this second row to the #__content_types table is as follows:

```
INSERT INTO `#__content_types` (`type_id`, `type_title`, `type_alias`, `table`,
`rules`, `field_mappings`, `router`, `content_history_options`)
VALUES
(NULL,
'Subscriptions',
'com_joomprosubs.subscription',
'{ "special":
{ "dbtable": "#__joompro_subscriptions", "key": "id", "type": "Subscription", "prefix":
"JoomprosubsTable" } }',
'', '', '', '');
```

These lines should be added to the file administrator/components/com_joomprosubs/sql/install.mysql.utf8.sql so that these rows will be created when the component is installed. You should also run

these commands on your database to add the two rows to your #__content_types table before going to the next section. Remember to change the placeholder prefix "#__" to the actual prefix for your database before running the SQL commands on your local database.

Tip: When creating the values for the JSON strings, one trick is to copy from an existing row and edit the strings.

Add Component Level Options

The content history component uses two options as follows.

- **save_history:** If Yes, history is saved. Otherwise, no history is saved.
- **history_limit:** If > 0, limits the number of different history versions saved in the database. For example, if this is set to 10, then when the 11th history version is saved, the oldest is deleted automatically.

If our example component, we will add the following lines to the file administrator/components/com_joomprosubs/config.xml.

```
<fieldset name="component">
    <field
        name="save_history"
        type="radio"
        class="btn-group btn-group-yesno"
        default="1"
        label="JGLOBAL_SAVE_HISTORY_OPTIONS_LABEL"
        description="JGLOBAL_SAVE_HISTORY_OPTIONS_DESC"
    >
        <option
            value="0">JNO</option>
        <option
            value="1">JYES</option>
    </field>

    <field
        name="history_limit"
        type="text"
        filter="integer"
        label="JGLOBAL_HISTORY_LIMIT_OPTIONS_LABEL"
        description="JGLOBAL_HISTORY_LIMIT_OPTIONS_DESC"
        default="5"
    />
</fieldset>
```

With this code, we will now be able to add and save these options. Go into the Options for the component and set save_history to Yes and history_limit to 10.

Update Content History During Table Save and Delete

Next we need to tell our component to call the content history methods during the save and delete operations. We can do this by adding this line of code to the __construct()

method of the JoomprosubsTableSubscription class.

```
JObserverMapper::addObserverClassToClass('JTableObserverContenthistory',  
'JoomprosubsTableSubscription', array('typeAlias' =>  
'com_joomprosubs.subscription'));
```

This code registers the content history table as an observer class for our component table. When a row is saved or deleted from our table, the appropriate methods are called automatically for the content history table.

At this point, we should be able to see content history working for the Joomprosubs categories. Test this by adding a new category for the component and then clicking on the Versions button in the toolbar. It should work exactly the same as categories for the core Joomla components.

Add type alias to model

To make restoring work, you need to add \$typeAlias property to your model class.

Model is located in

administrator/components/com_joomprosubs/models/subscription.php

```
public $typeAlias = 'com_joomprosubs.subscription';
```

If this is not set, you will get Error restoring item version from history. error, because Joomla! can not find correct row from #__content_types table.

Add Versions Button to Tool Bar

At this point, categories are working completely and we are saving history versions in the #__ucm_history table for our component. However, we need to be able to access these changes on our edit screen. To do this, we need to add the Versions button to the edit screen's toolbar. We do this by adding this code to the file administrator/components/com_joomprosubs/views/subscription/view.html.php, just before the code that adds the "cancel" button, as shown here.

```
else {  
    if ($this->state->params->get('save_history', 1) && $user->  
>authorise('core.edit')) {  
        JToolBarHelper::versions('com_joomprosubs.subscription', $this->item->id);  
    }  
    JToolBarHelper::cancel('subscription.cancel', 'JTOOLBAR_CLOSE');  
}
```

This code checks that we have the save_history option set and that we have edit permission for this component. If so, we show the Versions button using the JToolBarHelper::versions() method. This method has two arguments: the type_alias and the primary key of the component item row in the database.

With this code added, we can now go into the Joomprosubs component, add and save a new subscription, and now see the Versions button. If we click on it, we will see the Item

Version History modal window open with our saved version.

There are still two things missing, however. One is that we don't have a way to add the Version Note to each version. The second is that we don't currently have meaningful labels for the versions when we use the Preview or Compare features. We will address these in the next two sections of this tutorial.

Add Version Note Field to Form

The content history feature adds a new field called "version_note" to the component edit form. This optional field is not saved in the component table. Instead, it is used to label the version in the content history table.

To add this field to our component edit screen, we do changes to two files. First, we add the field to the XML form file `administrator/components/com_joomprosubs/models/forms/subscription.xml` as follows:

```
<field name="version_note"
      type="text"
      label="JGLOBAL_FIELD_VERSION_NOTE_LABEL"
      description="JGLOBAL_FIELD_VERSION_NOTE_DESC"
      class="inputbox" size="45"
      labelclass="control-label"
/>
```

We can add it anywhere inside the fieldset element. Because we will place the field "created_by_alias" on the screen, we add it in the XML file after this element.

Second, we add this code to the file `administrator/components/com_joomprosubs/views/subscription/template/edit.php`, just after the control group for "created_by_alias", as follows:

```
<div class="control-group">
    <div class="control-label"><?php echo $this->form-
>getLabel('version_note'); ?></div>
    <div class="controls"><?php echo $this->form->getInput('version_note'); ?
></div>
</div>
```

Now we can add in a version note and see this in the Item Version History modal window.

Add Labels to Pop-Up Windows

At this point, everything works. However, when we use the pop-up Preview or Compare windows in the Item Version History modal, we see the database column names instead of the translated label from the form. Also, we see the category and user id numbers instead of the category name and user name.

We can improve the readability of windows by entering in some additional about our component's table. This information is entered as a JSON-formatted string in the

content_history_options column of the #__content_types table. The following information can be entered in this column.

- **formfile:** This is the path to the XML JForm file for this form. If you add this, the Preview and Compare views will look up the labels from this XML file. This way the user will see translated labels instead of the database column name.
- **hideFields:** Some database columns are not meaningful for the user when viewing the item. For example, asset_id or check_out_time are not things that appear in the form and are not helpful to the user when figuring out the contents of an item. This is entered as an array of column names.
- **ignoreChanges:** The content history component uses a "hash" calculation (Sha1) to determine whether an item has changed. This allows you to see which version in history matches the current version. It also prevents duplicate versions from being saved in the content history table (for example, if you press the "save" button without making any changes). For this to work properly, we need to exclude some columns from the hash calculate. The "ignoreChanges" lets you exclude some database columns from the hash so that changes to these columns will not be considered real changes to the item. For example, columns such as "hits" or "modified_time" will change with each save, even if no meaningful data was changed in the item. This is an array of database column names.
- **convertToInt:** When the hash value is created, it uses a JSON array of the database column values. In some cases, such as start and stop publishing dates, the value might be blank when a row is first created and then a different value after the item is saved. To get a consistent hash value for the first and subsequent saves, these values can be converted to integers before the hash is created. That way, we don't think a value has changed when it really hasn't. This is an array of database column names.
- **displayLookup:** Here we can define how to display more meaningful data, for example, displaying a category title or user name instead of the ID. This is stored as an array of PHP standard class objects. Each object has the following fields.
 - *sourceColumn:* The column in the form to replace. For example, the "created_user_id" or "catid".
 - *targetTable:* The database table to get the title or name. For example, "#__users" or "#__categories".
 - *targetColumn:* The column in the target table to use in the SQL query JOIN statement. For example, "id".
 - *displayColumn:* The column in the target table to display in the Preview or Compare pop-up window. For example, "name" or "title".

In our example component, we need the following values.

- **formfile:**
administrator/components/com_joomprosubs/models/forms/subscription.xml.
- **hideFields:** checked_out, checked_out_time, params, language. The last two columns are not used.

- **ignoreChanges:** modified_by, modified, checked_out, checked_out_time.
- **convertToInt:** publish_up, publish_down.
- **displayLookup:** We look up five fields: catid, group_id, created_by, access, and modified_by. The values for each are as follows.
 - *sourceColumn:* catid
 - *targetTable:* #__categories
 - *targetColumn:* id
 - *displayColumn:* title
- *sourceColumn:* group_id
- *targetTable:* #__usergroups
- *targetColumn:* id
- *displayColumn:* title
- *sourceColumn:* created_by
- *targetTable:* #__users
- *targetColumn:* id
- *displayColumn:* name
- *sourceColumn:* access
- *targetTable:* #__viewlevels
- *targetColumn:* id
- *displayColumn:* title
- *sourceColumn:* modified_by
- *targetTable:* #__users
- *targetColumn:* id
- *displayColumn:* name

The entire JSON string for this column is as follows.

```
{ "formFile": "administrator\\components\\com_joomprosubs\\models\\forms\\subscription.xml",
  "hideFields": [ "checked_out", "checked_out_time", "params", "language" ],
  "ignoreChanges": [ "modified_by", "modified", "checked_out", "checked_out_time" ],
  "convertToInt": [ "publish_up", "publish_down" ],
  "displayLookup": [
    { "sourceColumn": "catid", "targetTable": "#__categories", "targetColumn": "id", "displayColumn": "title" },
    { "sourceColumn": "group_id", "targetTable": "#__usergroups", "targetColumn": "id", "displayColumn": "title" },
    { "sourceColumn": "created_by", "targetTable": "#__users", "targetColumn": "id", "displayColumn": "name" },
    { "sourceColumn": "access", "targetTable": "#__viewlevels", "targetColumn": "id", "displayColumn": "title" },
    { "sourceColumn": "modified_by", "targetTable": "#__users", "targetColumn": "id", "displayColumn": "name" } ] }
```

Note that the slashes are escaped inside the JSON string. In order to get this column correctly into the database when our component is installed, we need to modify the file administrator/components/com_joomprosubs/sql/install.mysql.utf8.sql where we create the #__content_types for the component as follows:

```
INSERT INTO `#__content_types` (`type_id`, `type_title`, `type_alias`, `table`,
```

```
`rules`, `field_mappings`, `router`, `content_history_options`) VALUES
(NULL, 'Subscriptions', 'com_joomprosubs.subscription', '{"special":
{"dbtable":"#__joompro_subscriptions","key":"id","type":"Subscription","prefix":
"JoomprosubsTable"}}',' ',' ',' ',
'{"formFile":"administrator\\components\\com_joomprosubs\\models\\forms\\su
bscription.xml", "hideFields":
["checked_out","checked_out_time","params","language"], "ignoreChanges":
["modified_by", "modified", "checked_out", "checked_out_time"], "convertToInt":
["publish_up", "publish_down"], "displayLookup":
[{"sourceColumn":"catid","targetTable":"#__categories","targetColumn":"id","disp
layColumn":"title"},
{"sourceColumn":"group_id","targetTable":"#__usergroups","targetColumn":"id","di
splayColumn":"title"},
{"sourceColumn":"created_by","targetTable":"#__users","targetColumn":"id","displ
ayColumn":"name"},
{"sourceColumn":"access","targetTable":"#__viewlevels","targetColumn":"id","disp
layColumn":"title"},
{"sourceColumn":"modified_by","targetTable":"#__users","targetColumn":"id","disp
layColumn":"name"} ]}');
```

Note that the "/" characters are now preceeded by two escape "\" characters. The first one is the SQL escape character for the second "\" character. The second "\" is the JSON escape character for the "/" character.

With this change to the SQL install file, the changes for content history are complete.

Update Extension SQL Files

To tidy things up, should should edit the uninstall SQL file to remove the rows from the #__content_types table. Also, if your component uses the one-click update feature, you should create a SQL file to create these rows when the component is updated.

Hopefully, this tutorial will help you as you update your component to take advantage of the content history feature in Joomla. You can download the com_joomprosubs component with the code for content history added here: https://github.com/joomla/jdoc-examples/raw/master/zip_archives/com_joomprosubs-3.2.1.zip.

https://docs.joomla.org/Using_Content_History_in_your_Component

Content Versions in Joomla 3.2

By Steve Burge 09 September 2013



Do you work with clumsy people?

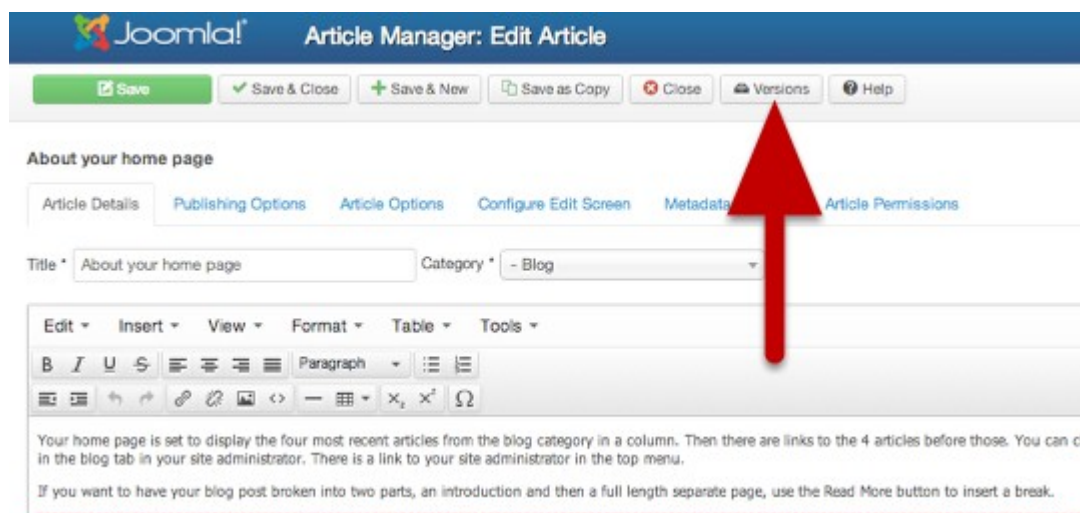
Are you sometimes clumsy?

If you answered "Yes" to either question, then we've a new Joomla feature that's perfect for you.

Versions is a security feature for your content. Versions allow you to keep a copy of your Joomla articles every time you save a change.

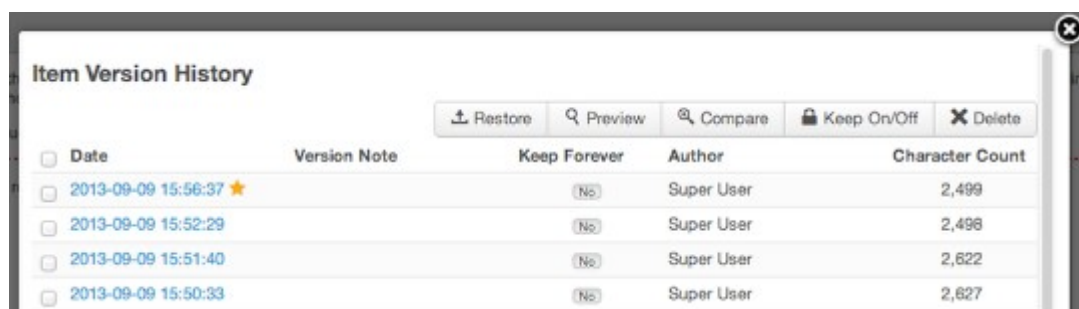
The code for Versions has already been committed, so it is certain to ship with Joomla 3.2, which is due in the next couple of months.

The Versions button appears in the toolbar at the top of your articles screen:

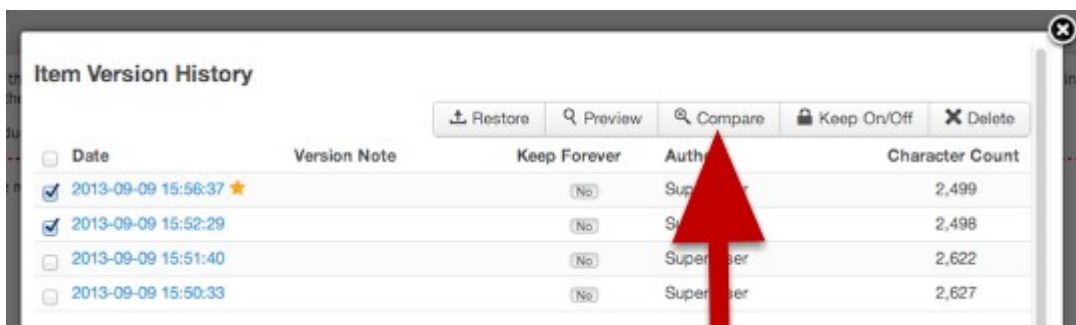


Every time you save your article, a new version will be accessible via this pop-up window.

The current version of the article is marked by a gold star.



To compare the current version with a previous version, check the boxes next to the versions and click Compare.



Text that has been removed is marked in red and text that has been added will be marked in green.

Saved on 2013-09-09 15:51:40	Saved on 2013-09-09 15:52:29	Changes
On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.	On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.	On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.
On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want..	Extra test for the new version.	On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want. Extra test for the new version.

Joomla doesn't just store all of your content, but also the settings for your articles.

In this example, you can see that the version on the right has been changed so that Featured set to Yes.

Ordering	1	1	1
Meta Keywords			
Meta Description			
Access	Public	Public	Public
Hits	4	4	4
Metadata Options			
Robots	Use Global	Use Global	Use Global
Featured	No	Yes	No Yes
Language	*	*	*

To roll back to a previous version, check the box next to the version and click Restore:

Item Version History		
		<input type="button" value="Restore"/> <input type="button" value="Preview"/>
<input type="checkbox"/> Date	Version Note	Keep Forever
<input type="checkbox"/> 2013-09-09 15:56:37 ★		<input type="button" value="No"/>
<input checked="" type="checkbox"/> 2013-09-09 15:52:29		<input type="button" value="No"/>
<input type="checkbox"/> 2013-09-09 15:51:40		<input type="button" value="No"/>
<input type="checkbox"/> 2013-09-09 15:50:33		<input type="button" value="No"/>

Options to see changes more easily

Because all of the options for the article are saved, it can be easier to see only what has changed.

Click Changed Values in the top-right corner.

		<input type="button" value="Changed Values"/> <input type="button" value="Show HTML Code"/>
Saved on 2013-09-09 15:56:37	Saved on 2013-09-09 18:23:43	Changes
4	4	
About your home page	About your home page	About your home page
about-your-home-page	about-your-home-page	about-your-home-page

You'll now see only the article options that were changed:

		<input type="button" value="All Values"/> <input type="button" value="Show HTML Code"/>
Saved on 2013-09-09 15:56:37	Saved on 2013-09-09 18:23:43	Changes
On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.	On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.	On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.
Extra test for the new version.	On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.	Extra test for the new version. On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.
2013-09-09 15:56:37	2013-09-09 15:50:33	2013-09-09 15:56:37
Yes	No	Yes No

By default, versions will show only the article text, but there is a Show HTML Code button:

		Changed Values	Show HTML Code
Saved on 2013-09-09 15:56:37	Saved on 2013-09-09 18:23:43	Changes	
4	4	4	
About your home page	About your home page	About your home page	
about-your-home-page	about-your-home-page	about-your-home-page	

That will show you the underlying HTML code in the article:

		All Values	Show Text
Saved on 2013-09-09 18:23:43	Changes		
On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.	<p><p>On the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want.</p></p> <p><p>Extra test for the new versionOn the full page you will see both the introductory content and the rest of the article. You can change the settings to hide the introduction if you want. </p></p> <p><p> </p></p> <p><p> </p></p>		
2013-09-09 15:50:33	2013-09-09 15:56:37		
No	YesNo		

This HTML feature will be useful because currently the versions don't display images. Even if they did, there wouldn't be room inside the pop-up to cleanly display large images.

		Changed Values	Show HTML Code
Saved on 2013-09-09 18:23:43	Saved on 2013-09-10 14:01:36	Changes	
4	4	4	
About your home page	About your home page	About your home page	
about-your-home-page	about-your-home-page	about-your-home-page	
Your home page is set to display the four most recent articles from the blog category in a column. Then there are links to the 4 articles before those. You can change those numbers by editing the content options settings in the blog tab in your site administrator. There is a link to your site administrator in the	Your home page is set to display the four most recent articles from the blog category in a column. Then there are links to the 4 articles before those. You can change those numbers by editing the content options settings in the blog tab in your site administrator. There is a link to your site administrator in the	Your home page is set to display the four most recent articles from the blog category in a column. Then there are links to the 4 articles before those. You can change those numbers by editing the content options settings in the blog tab in your site administrator. There is a link to your site administrator in the	

Version options

There are two new configuration options for Versioning.

- Save History allows you to turn Versioning on and off.

- Maximum versions allows to limit the number of versions that are stored.

Articles Editing Layout Category Categories Blog / Fe

These options control the layout of the article editing page.

Show Publishing Options

Show Article Options

Save History

Maximum Versions

Frontend Images and Links

Even if the maximum versions is set to 10, it is possible to mark a version as so that it can not be deleted:

Item Version History

		<input type="button" value="Restore"/>	<input type="button" value="Preview"/>
<input type="checkbox"/>	Date	Version Note	Keep Forever
<input type="checkbox"/>	2013-09-09 18:23:43		<input checked="" type="button" value="Yes"/>
<input type="checkbox"/>	2013-09-09 15:56:37 ★		<input type="button" value="No"/>

The versions are stored it a new database table called `_ucm_history`. UCM stands for Unified Content Model.

Over time, the goal of Joomla's developers is to unify the way that Joomla handles content whether it's articles, users or data from other extensions. This table should allow other extensions to also tap into Joomla's new versioning feature.

localhost » Joomla3 » c1cbf_ucm_history

[Browse](#)
[Structure](#)
[SQL](#)
[Search](#)
[Insert](#)
[Export](#)
[Import](#)
[Operations](#)
[Triggers](#)

Column	Type	Function	Null	Value
version_id	int(10) unsigned			8
ucm_item_id	int(10) unsigned			4
ucm_type_id	int(10) unsigned			1
version_note	varchar(255)			
save_date	datetime			2013-09-09 15:56:3
editor_user_id	int(10) unsigned			504
character_count	int(10) unsigned			2499
sha1_hash	varchar(50)			b91b41ec1399c99a51b94b20c03f91d5b2fc65e3
version_data	mediumtext			<pre>{ "show_urls_images_backend": "\\", "show_urls_images_frontend": "\\", "version": 10, "ordering": 1, "metakey": "", "metadesc": "", "access": 1, "hits": 4, "metadata": { "\robots": "\\", "\author": "\\", "\rights": "\\", "\xreference": "\\", "featured": 1, "language": "*", "xreference": "" } }</pre>
keep_forever	tinyint(4)			0

Go

<https://www.ostraining.com/blog/joomla/content-versions/>