Sitecore CMS 7.0 or later

Sitecore Serialization Guide

An administrator's guide to serializing content in Sitecore

Table of Contents

Chapter 1

Introduction

The Sitecore serialization functionality is designed to help teams of developers that work on the same Sitecore solution to synchronize database changes between their individual development environments, but is also valuable when a single developer works on a solution.

Serialization allows you to serialize an entire Sitecore database or a series of items in a database to text files. You can then use these text files to transfer this database or series of items to another database or Sitecore solution. Sitecore serialization is typically used in combination with a source control system to make it easy for developers to synchronize database changes between their local databases that they use for development — with the added benefit that the source control system will keep track of database changes and allow you to compare database changes.

This document describes all the actions that you can perform with Sitecore serialization.

This document contains the following chapters:

- **Chapter 1 — Introduction**
  This introduction to the document.
- **Chapter 2 — Serialization**
  This chapter describes how to use Sitecore serialization.

Chapter 2

Serialization

This chapter describes how to use the Sitecore serialization functionality to administer and develop a Sitecore CMS solution.

This chapter contains the following sections:

- Serialization Overview
- Serializing an Item or a Tree
- Updating an Item, a Tree of Items or a Database
- Reverting an Item, a Tree of Items, or a Database
- Serializing Items Using Event Handlers
- Using the Service Page to Serialize, Update, and Revert a Database
- Using Sitecore Serialization with a Source Control System

2.1 Serialization Overview

Serialization allows you to convert a whole database or a series of items in a database to text files. You can then use these text files to transfer this database or series of items to another database or Sitecore solution.

Sitecore provides three serialization access points and all of them use the same general setting in the *web.config* file.

2.1.1 Serialization Access Points

You can perform serialization actions in three ways:

- In the Content Editor, you can manually serialize, update, and revert items.

  For more information about using the Content Editor, see the *Serializing an Item or a Tree of Items*, *Updating an Item, a Tree of Items or a Database* and *Reverting an Item, a Tree of Items, or a* Database sections.

- You can use the Sitecore event handlers to automatically serialize items.

  For more information about using event handlers, see the *Serializing Items Using Event Handlers* section.

- You can use the Sitecore service page to serialize, update, and revert a whole database.

  For more information about using the service page, see the *Using the Service Page to Serialize, Update, and Revert* a Database section.

2.1.2 Serialization Folder

When you serialize, update, or revert items, Sitecore uses only one general setting — the serialization folder. This setting contains the path to the folder in the file system where the text files of the items are stored. By default, Sitecore stores serialized items in the `website_root/data/serialization` folder.

To set another path to the serialization folder:

1. Open to the `website_root/website/web.config` file.
2. In the `SerializationFolder` parameter, specify the path to the serialization folder.

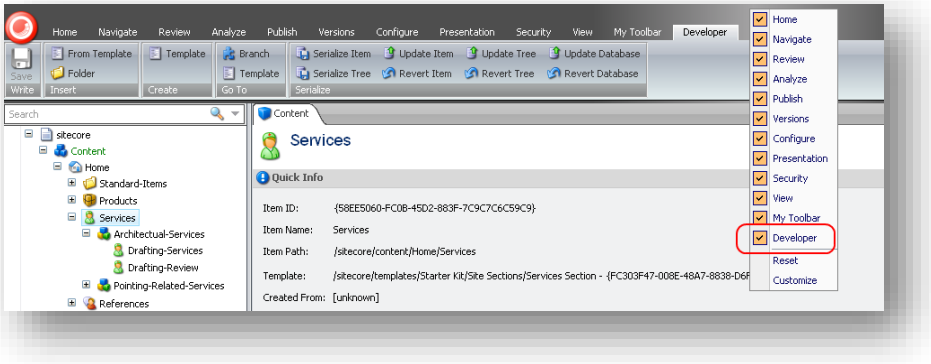
2.1.3 Enabling the Developer Tab

You can perform all the serialization actions in the Content Editor on the **Developer** tab. The **Developer** tab is disabled by default.

To enable the **Developer** tab:

1. Open the **Content Editor.**

2. Right click on the **Content Editor** ribbon and select **Developer**.



2.2 Serializing an Item or a Tree of Items

When you serialize an item or a tree of items, Sitecore stores the item hierarchy in text files.

2.2.1 Storing Text Files

When you are serializing items, Sitecore creates a separate text file for each item it processes. Sitecore generates the full path to the files:

Serialization Folder \ Database Name \ Path to the item in the Database \ Item Name \ ".item"*

For example:
D:\CMS640update1\Data\serialization\master\sitecore\content\Home\Services\Architectual-Services.item

Each text file contains the following information:

- Item definition: version, ID, path, and so on.
- Shared fields.
- Item versions:
  - Version definition, including language version, numbered version, and revision number.
  - Non-shared fields.

A serialized item might look like this:

```
----item----
version: 1
id: {7F0A4C0A-083D-4D21-BA63-C3F1051BA6D6}
database: master
path: /sitecore/content/Home/Services/Architectual-Services
parent: {58EE5060-FC0B-45D2-883F-7C9C7C6C59C9}
name: Architectual-Services
master: {00000000-0000-0000-0000-000000000000}
template: {373CAA7B-5698-4E20-AA90-7698C4CE81EA}
templatekey: Services Category

----field----
field: {F7D48A55-2158-4F02-9356-756654404F73}
name:   Standard values
key:   standard values
content-length: 0
…
<r />
```

Each ----*field*---- section in the text file contains the *content-length* parameter, which contains the length of the corresponding item field value.

Storing Text Files for Deeply Nested Items

The Windows length limit of a single path to a file is 255 characters. For content trees with multiple nested levels, this limitation can cause items in nested folders to be omitted during the serialization process.

To overcome the limitation and to serialize all of the requested items, Sitecore sometimes shortens the paths if items are deeply nested. To store the serialization output text files for these items, in addition to standard database folders, Sitecore creates auxiliary file folders with random names under the *$dataFolder\serialization* folder, for example: *D:\CMS640update1\Data\serialization\BD8C390B*.

Important
To be able to revert to the full content tree in another Sitecore environment, you must copy both the database folders and the auxiliary folders to the new destination.

2.2.2 Serializing Items

In Sitecore, you can serialize a single item or a series of items.

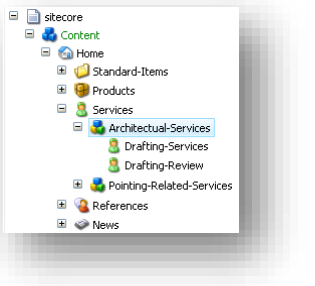
To serialize an item:

1. In the **Content Editor**, select an item that you want to store in a text file on the disk.
2. On the **Developer** tab, in the **Serialize** group, click **Serialize Item**.

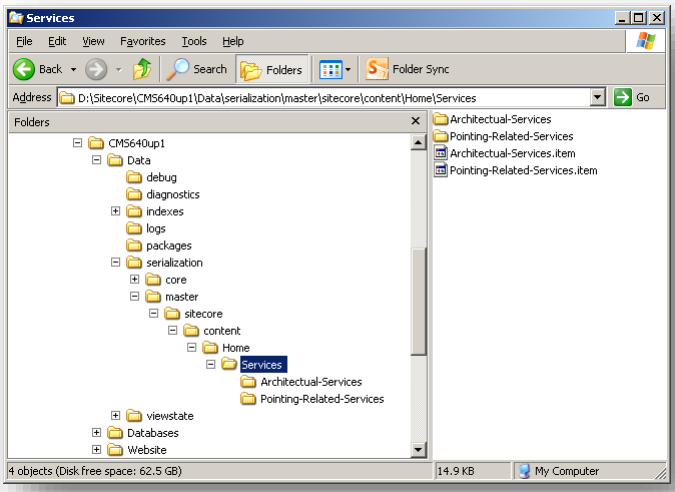
To serialize an item with all its subitems:

1. In the **Content Editor**, select a parent item that you want to store with all its subitems in text format on the disk.
2. On the **Developer** tab, in the **Serialize** group, click **Serialize Tree**.

In our example we have the following structure of items:



When we serialize a tree of items, starting from the *Services* item, Sitecore creates the following hierarchy of folders in the file system:



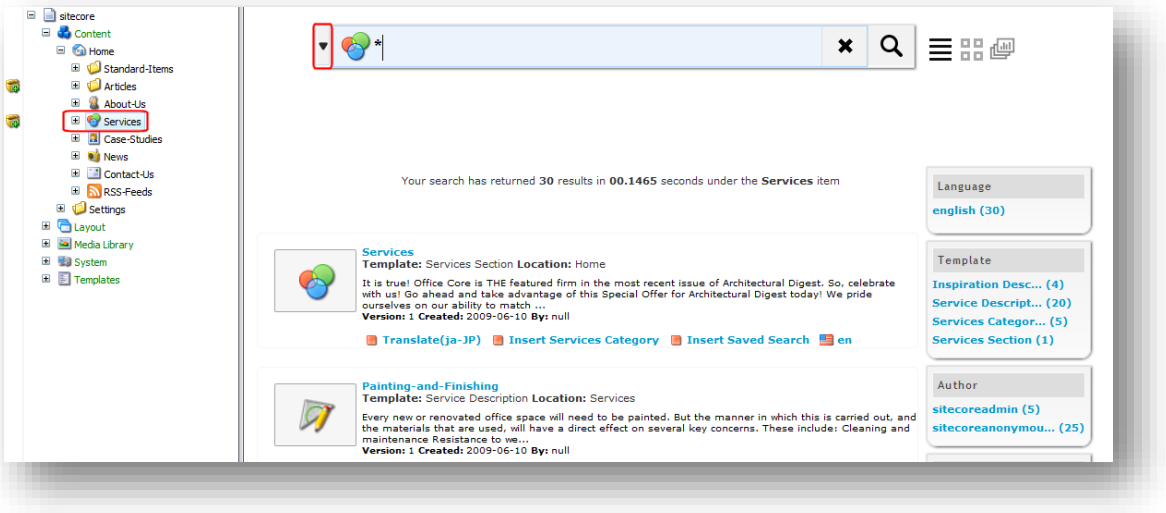
2.2.3 Serializing Items in Bulk

You can serialize items in bulk using search operations. If your website contains thousands or millions of items contained in item buckets, serializing in bulk enables you to apply changes more quickly to multiple items.

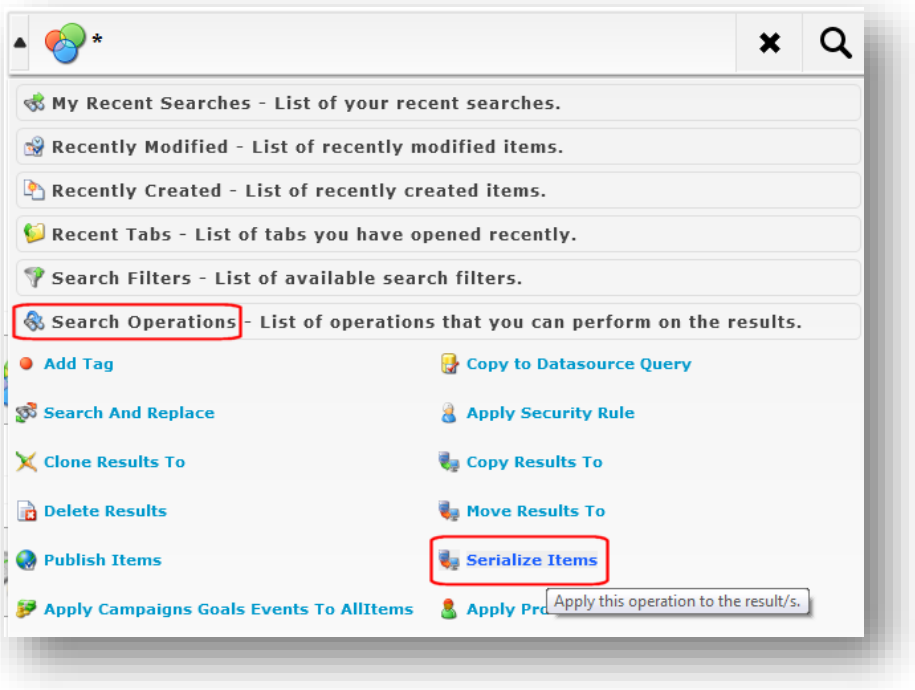
To serialize items in bulk:

1. Select a content item or item bucket to search. For example, **Services**.

2. In the search field enter * to return all items below **Services**.



3. Click the drop down to the left of the search field and select **Search Operations**.
4. In **Search Operations** click **Serialize Items**.



The **Serialize Items** operation is applied to all the items returned in your search results.

When serialization is complete you see a message dialog box informing you that serialization was completed successfully.

Sitecore 6.1.0 rev.090630 and later versions let you serialize security entities. The following table describes the serialization commands and the applications where you can perform them:

Sitecore Application	Command	Description	Storage Folder
User Manager	Serialize User	Serializes the selected user.	*Serialization folder / security / domain name / users / user name /*.user* Serialization folder is set in the *web.config* file.
	Serialize All Users	Serializes all the users.	
	Revert User	Reverts the selected user.	
	Revert All Users	Reverts all the users. **Warning**: All the users who are in the database but are not in the file system will be removed.	
Role Manager	Serialize Role	Serializes the selected role.	*Serialization folder / security / domain name / roles / role name /*.role* Serialization folder is set in the *web.config* file.
	Serialize All Roles	Serializes all the roles.	
	Revert Role	Reverts the selected role.	
	Revert All Roles	Reverts all the roles. **Warning**: All the roles that are in the database but are not in the file system will be removed.	
Domain Manager	Serialize Domain Users	Serializes all the users in the selected domain.	
	Serialize Domain Roles	Serializes all the roles in the selected domain.	
	Serialize Domain Users and Roles	Serializes all the users and roles in the selected domain.	
	Revert Domain Users	Reverts all the users in the selected domain. **Warning**: all users from the selected domain which are not in the file system will be removed.	
	Revert Domain Roles	Reverts all the roles from the selected domain. **Warning**: all roles from the selected domain which are not in the file system will be removed.	
	Revert Domain Users and Roles	Reverts all the users and roles from the selected domain. **Warning**: all users and roles from the selected domain which are not in the file system will be removed.	

2.3 Updating an Item, a Tree of Items or a Database

Sitecore lets you update items with the information that is stored in the text files in the file system. For more information about converting Sitecore items into text files, see the *Serializing an Item or a Tree of Items* section.

2.3.1 Updating Rules

When you are updating items, Sitecore merges the changes from the text files with the current database items according to the following rules:

- If an item or an item version exists in the file system, but there is no such item or item version in the database, Sitecore adds it to the database.
- If an item or an item version exists in the database but there is no such item or item version in the file system, Sitecore does not remove it from the database.
- Sitecore does not overwrite the following information about the item:
  - *Item name*
  - *Template ID*
  - *Origin ID*
  - *Parent ID*
- If an item version that is loaded from the file system contains the same number and language version, Sitecore only overwrites the existing item version if the date in the new item in the *Updated* field is later in the current one.

2.3.2 Updating Items or Database

In Sitecore, you can update items or a whole database.

To update an item from the text file:

1. In the **Content Editor**, select the item that you want to update.
2. On the **Developer** tab, in the **Serialize** group, click **Update Item**.

To update an item with all its subitems from the file system:

1. In the **Content Editor**, select the parent item that you want to update with all its subitems.
2. On the **Developer** tab, in the **Serialize** group, click **Update Tree**.

To update the whole database:

1. In the **Content Editor**, select any item.
2. On the **Developer** tab, in the **Serialize** group, click **Update Database**.

2.4 Reverting an Item, a Tree of Items, or a Database

Sitecore lets you roll back changes and revert items. When you are reverting items Sitecore loads the items from the file system and overwrites any changes. This procedure lets you bring the database to the exactly same state as it is in the file system.

2.4.1 Reverting Rules

When you are reverting items Sitecore loads content from the file system to the database according to the following rules:

- If an item or an item version exists in the database but there is no such item or item version in the file system, Sitecore deletes it from the database.
- If an item or an item version exists in the file system but there is no such item or item version in the database, Sitecore adds it to the database.
- If an item or an item version differs from the corresponding one in the file system, Sitecore overwrites it.

2.4.2 Reverting Items

In Sitecore you can revert items and a whole database.

To revert an item:

1. In the **Content Editor**, select the item that you want to revert.
2. On the **Developer** tab, in the **Serialize** group, click **Revert Item**.

To revert an item and all its subitems:

1. In the **Content Editor**, select the parent item that you want to revert with all its subitems.
2. On the **Developer** tab, in the **Serialize** group, click **Revert Tree**.

To revert the whole database:

1. In the **Content Editor**, select any item.
2. On the **Developer** tab, in the **Serialize** group, click **Revert Database**.

2.5 Serializing Items Using Event Handlers

You can use Sitecore event handlers to automatically save any database changes to the file system. By default, serialization event handlers are disabled. To enable event handlers add the following strings to the `web.config` file:

Sitecore CMS 7.0 - 7.0 Update-3

```
        <configuration>
           <sitecore>
             <events>
               <event name="item:saved">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemSaved"/>
               </event>
               <event name="item:deleted">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemDeleted"/>
               </event>
               <event name="item:moved">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemMoved"/>
               </event>
               <event name="item:versionRemoved">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemVersionRemoved"/>
               </event>
             </events>
           </sitecore>
        </configuration>
```

Sitecore CMS 7.0 Update-4

```
        <configuration>
           <sitecore>
             <events>
        <event name="item:copied">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemCopied"/>
               </event>
               <event name="item:renamed">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemRenamed"/>
               </event>
               <event name="item:saved">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemSaved"/>
               </event>
               <event name="item:deleted">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemDeleted"/>
               </event>
               <event name="item:moved">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemMoved"/>
               </event>
               <event name="item:versionRemoved">
                 <handler type="Sitecore.Data.Serialization.ItemHandler, Sitecore.Kernel"
method="OnItemVersionRemoved"/>
               </event>
             </events>
           </sitecore>
        </configuration>
```

Note
Sitecore event handlers automatically save any changes in Sitecore items to the file system and this is why we recommend using event handlers with a revision control system.

2.6 Using the Service Page to Serialize, Update, and Revert a Database

You can serialize, update, and revert a whole database without using the Content Editor. The Sitecore service page lets you perform any serialization actions on the whole database.

2.6.1 Managing a Database

You can use the Sitecore service page to manage a database.

To use the service page to serialize, update, or revert a whole database:

1. In your browser's address bar type:
   `http://web_site/sitecore/admin/serialization.aspx`
2. On the *Log into Sitecore* page enter the username (domain\user_name) and password and then click **Login**.
3. On the **Serialize and revert databases** page, in the **Select database** section, select one or more databases that you want to serialize, update, or revert.
4. Select **Serialize selected databases**, **Update selected databases,** or **Revert selected databases** depending on the action that you want to perform on the database.

2.6.2 Serializing a Preconfigured Set of Items

You can serialize not only the whole database but a preconfigured set of items on the serialization service page

Configuring a Set of Items

You can specify the predefined set of items as XML nodes in the *web.config* file or in any included configuration files using *include* and *exclude* entries.

The *Include* entry defines the paths to the items that will be serialized including all their subitems. The *Include* entry can use the following attributes:

- database — the database name.
- path — the path to the root item.

The *Exclude* entry defines the items that will not be serialized. The *Exclude* entry can use the following attributes:

- path — the path to the item.
- id — the item ID.
- templateid — the template ID of the item.
- template — the template name of the item.

In this example, we serialize all the subitems of the *Content* item in the *Master* database except for the items under the *Home* item and those items that are based on the *Sample item* template:

```
<?xml version="1.0" encoding="utf-8"?>
  <sitecore>
    <serialization>
      <default>
         <include database="master" path="/sitecore/content">
            <exclude path="/sitecore/content/Home"/>
            <exclude template="Sample item"/>
         </include>
      </default>
    </serialization>
```

```
    </sitecore>
  </configuration>
```

Serializing a Set of Items

When you have configured a set of items you can serialize it.

To serialize a preconfigured set of items:

1. In your browser's address bar type:
   `http://web_site/sitecore/admin/serialization.aspx`
2. On the *Sitecore* Log in page, enter the username (domain\user_name) and password and then click **Login**.
3. On the **Serialize and revert databases** page, in the **Serialize databases** section, click **Serialize preconfigured**.

2.6.3 Updating a Tree of Items

The Sitecore serialization service page lets you update a tree of items without using the Content Editor.

To update a tree of items:

1. In your browser's address bar type:
   `http://web_site/sitecore/admin/serialization.aspx`
2. On the *Sitecore Log in* page, enter the username (domain\user_name) and password and then click **Login**.
3. On the **Serialize and revert databases** page, in the **Update** section, enter the path to the parent item in the file system.

   We recommend copying this path from Windows Explorer. The path might look like this: *master\sitecore\content\Home\Services*
4. Click **Update specific path**.

2.6.4 Customizing Serialization

You can customize the serialization service page to make it work as you need.

Use the following classes and namespaces to customize serialization:

- *Sitecore.Data.Serialization.Manager* — this is the central class of Sitecore serialization that provides the API for all the operations.
- *Sitecore.Data.Serialization.PathUtils* — this class contains methods for mapping item paths to the serialization tree and converting absolute paths into item paths.
- *Sitecore.Data.Serialization.ObjectModel* — this namespace contains classes that manipulate the data in serialization files.
- *Sitecore.Data.Serialization.Presets* — this namespace contains classes that are responsible for the serialization presets.

2.7 Using Sitecore Serialization with a Source Control System

As Sitecore serialization lets you convert items into text files you can use a source control system, for example Subversion (SVN), to manage this data. This is very important when a Sitecore solution is developed by a team of developers.

2.7.1 Convenience of SVN Usage

Using Sitecore serialization with SVN gives you the following advantages:

- Database state is stored in a central location in the source code repository. Database files are no longer a critical component because you can restore any item version from SVN.
- SVN stores the content as plain text and you can retrieve it without any special tools.
- SVN tracks all the item modifications:
    - SVN is responsible for solving any conflicts that can occur when two people modify the same item simultaneously.
    - You can use the *SVN Blame* command to show the author and revision information in-line for the specified files.

2.7.2 Integration with SVN

To integrate the Sitecore solution with SVN, follow this general scenario:

1. Back up the Sitecore items to the file system using serialization. For more information about serializing items, see the *Serializing Items* section.
2. Copy your tree of serialized folders into a repository using *SVN Import* and *SVN Checkout* or *SVN Add* and *SVN Commit* commands.
3. Update your SVN sandbox before you start to update Sitecore items with the *SVN Update* command. Use the *SVN Edit Conflicts* command to resolve conflicts where necessary.
4. Revert the Sitecore database to the latest state. For more information about reverting items, see the *Reverting Items* section.
5. Make any changes in the Sitecore items.
6. Serialize the items or use serialization event handlers to save the modifications to the file system.
7. Use the *SVN Add* command to register the modifications in the SVN sandbox or see if there are any unnecessary items.
8. Use the *SVN Commit* to review the changes and commit them to the SVN repository.
    - If there are any missing items in the SVN commit list, it means that the items have been deleted from the database but are still registered in SVN. Use the *SVN Delete* command to delete them from the sandbox or the *SVN Revert* and *SVN Update* commands to restore the deleted files and folders.