

Quick Tips for Managing the SharePoint 2010 Office Web Apps' Cache

Introduction

One of the hotly anticipated features in SharePoint 2010 is the introduction of the Microsoft Office Web Applications feature, aka “Office Web Apps.” The release of Office Web Apps opens up new possibilities for those working with documents and files that are tied to Microsoft Word and other applications in the Microsoft Office suite of applications.

What is Office Web Apps?

In prior versions of SharePoint, viewing and editing Office documents located in SharePoint document libraries normally required a client computer possessing the Microsoft Office suite of applications. For example, if you wanted to view or edit a Word document in SharePoint, you needed Microsoft Word (or an equivalent application) installed on your computer.

That situation changes with the arrival of Office Web Apps. When a SharePoint 2010 farm is properly set up and configured with Office Web Apps, it is possible to view and edit several different Office document types directly from within a browser as shown in Figure 1 below.

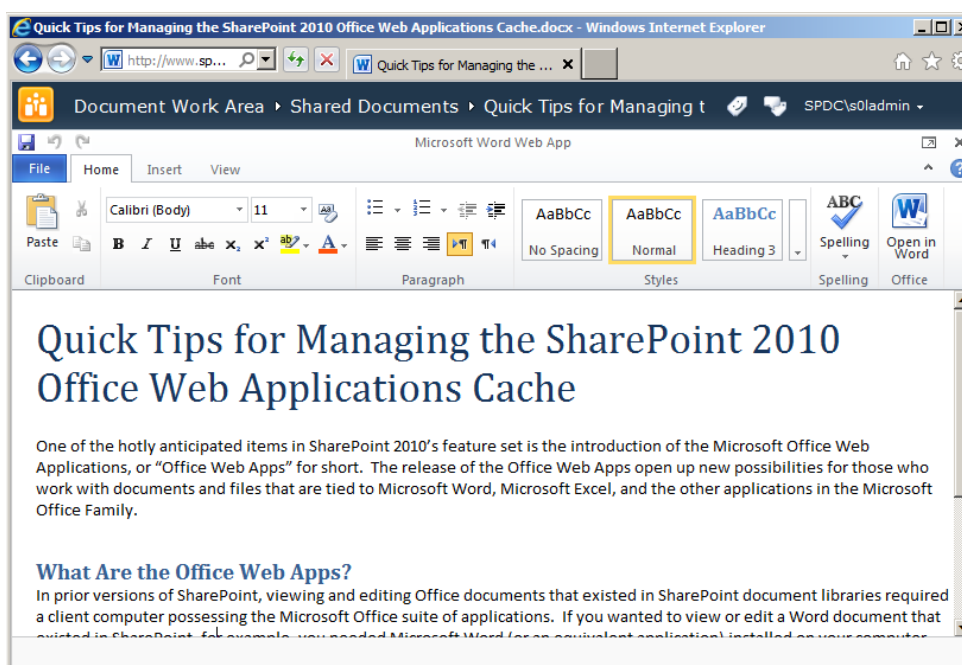


Figure 1: Browser-based editing of a Microsoft Word document

Office Web Apps provides browser-based viewing and editing support for Microsoft Excel, OneNote, PowerPoint, and Word document types. This support extends to more than just Internet Explorer with Firefox 3.x, Safari 4.x, and Google Chrome browser types also supported for viewing and editing. This makes Office Web Apps an enabler of cross-platform collaboration that centers on Office documents.

A Word about the Plumbing

As you might imagine, browser-based rendering and editing of Office documents involves a number of complex processes that engage a variety of front-end, middle-tier, and back-end components. The front-end and middle-tier tasks that are tied to document viewing and editing are handled primarily by a new set of service applications that appear when Office Web Apps is installed. These service applications (and their associated pages, handlers, and worker processes) take care of the business of document conversion, load-balancing, and rendering for browser consumption.

Document conversion and rendering typically generate a combination of images, HTML, JavaScript, and XAML (or eXtensible Application Markup Language) that are sent to consuming browsers. The creation of these document resources is an expensive process, both in terms of CPU cycles and storage. It makes sense to generate these document resources only as needed and reuse them whenever possible to improve performance levels. That's where Office Web Apps' cache comes in.

The Office Web Apps' cache is the back-end store that is responsible for housing images, HTML, JavaScript, and XAML resources once they have been created for a document. Each time a document is converted into a set of these resources, the resources are stored in the Office Web Apps' cache. When a request for a document comes into SharePoint, the cache is checked to see if the document had been previously requested and rendered. If it has, and the cached document resources are up-to-date for the document, then the document request is served from the cache instead of engaging Office Web Apps to convert and re-render it. Serving document resources from the Office Web Apps' cache can yield significant performance improvements over scenarios where no cache is employed.

Quick note before going too far: the Office Web Apps' cache is only employed for Word and PowerPoint document types. It is not used for OneNote or Excel documents.

Inside the Office Web Apps' Cache

The Office Web Apps' cache takes the form of a single site collection for each Web application within a SharePoint farm. When Office Web Apps is installed and configured in a SharePoint environment, a couple of new timer jobs are installed and run regularly within the farm. One of those timer jobs, the *Office Web Apps Cache Creation* timer job,

ensures that each Web application where Office Web Apps is running has a site collection like the one shown below in Figure 2.

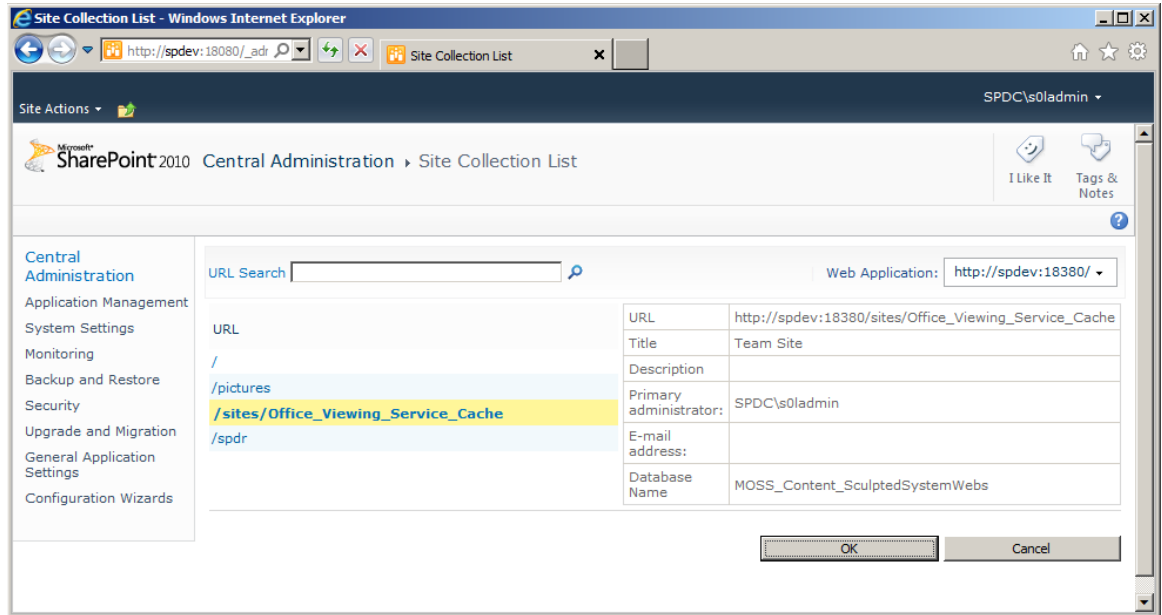


Figure 2: The Office_Viewing_Service_Cache site collection

The *Office_Viewing_Service_Cache* site collection is a standard Team Site, and it is the location where resources are stored after the conversion and rendering of either a Word or PowerPoint document by Office Web Apps.

The Team Site can be accessed just like any other SharePoint Team Site. A glimpse inside the All Documents library (showing a number of document resources) appears below in Figure 3.

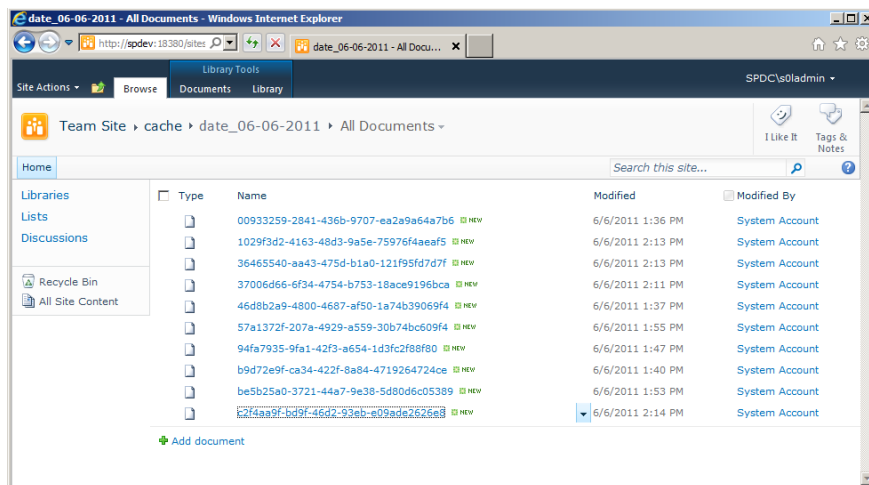


Figure 3: All Documents library in an Office Web Apps' cache site collection

Managing the Cache

For such a complex system, Office Web Apps' components do a pretty good job of maintaining themselves without external intervention. This extends to the site collections that are used by Office Web Apps for caching purposes. For example, the *Office Web Apps Expiration* timer job that is installed with Office Web Apps removes old document resources from cache site collections once they've hit a certain age. The timer job also ensures that each of the site collections responsible for caching has adequate space to serve its purpose.

This doesn't mean that there aren't opportunities for tuning and maintenance, though. In fact, there are a couple of things that every administrator should do and review when it comes to the Office Web Apps' cache.

Tip No. 1: Relocate the Cache to a New Database

The Office Web Apps Cache Creation timer job creates an Office_Viewing_Service_Cache site collection in a content database that is collocated with one or more of the "real" site collections within each of your content Web applications. Since the cache site collection is allowed to grow by default to a beefy 100GB, it makes sense to relocate the cache site collection to its own (new) content database. Relocating the cache site collection to its own content database makes it easier to exclude it from other maintenance, such as backups.

Relocating the cache site collection is pretty straightforward. It can be accomplished easily by following the *RelocateOwaCache.ps1* PowerShell script below. Simply save the script, execute it, and supply the URL of a Web application in your farm where Office Web Apps are running. The script will take care of creating a new content database within the Web application. It will then move the Web application's Office Web Apps' cache site collection to the newly created content database.

```
<#
.SYNOPSIS
    RelocateOwaCache.ps1
.DESRIPTION
    Relocates the Office Web Apps cache for a specified Web application to a new
    content database that is created by the script
.NOTES
    Author: Sean McDonough
    Last Revision: 07-June-2011
.PARAMETER targetUrl
    A Web application where Office Web Apps are in use
.EXAMPLE
    RelocateOwaCache.ps1 http://www.TargetWebApplication.com
#>
param
(
    [string]$targetUrl = "$(Read-Host 'Target Web application URL [e.g.
http://hostname]') "
)

function RelocateCache($targetUrl)
```

```

{
    # Ensure that the SharePoint cmdlets are loaded before continuing
    $spCmdlets = Get-PSSnapin Microsoft.SharePoint.PowerShell -ErrorAction
silentlycontinue
    if ($spCmdlets -eq $Null)
    { Add-PSSnapin Microsoft.SharePoint.PowerShell }

    # Get the name of the current database where the cache is located; it
    # will serve as the basis for a new content database name.
    $cacheSite = Get-SPOfficeWebAppsCache -WebApplication $targetUrl -
ErrorAction stop
    $newDbName = $cacheSite.ContentDatabase.Name + "_OWACache"

    # Create a new content database and relocate the cache to it. Make sure
the
    # user knows what's happening each step of the way.
    Write-Host "- creating a new content database ..."
    $cacheDb = New-SPContentDatabase -Name $newDbName -WebApplication
$targetUrl -ErrorAction stop
    Write-Host "- moving the Office Web Apps cache ..."
    Move-SPSite $cacheSite -DestinationDatabase $cacheDb -Confirm:$false -
ErrorAction stop
    Write-Host "- performing required IISRESET ..."
    iisreset | Out-Null

    # Let the user know where the cache is now located
    Write-Host "Cache successfully relocated to the '$newDbName' database."

    # Abort script processing in the event an exception occurs.
    trap
    {
        Write-Warning "`n*** Script execution aborting. See below for
problem encountered during execution. ***"
        $_.Message
        break
    }
}

# Launch script
RelocateCache $targetUrl

```

Tip No. 2: Review Size and Expiration Settings

When an Office_Viewing_Service_Cache site collection is provisioned within a Web application by the Office Web Apps Cache Creation timer job, it is initially configured to hold cached document resources for 30 days. As mentioned in Tip No. 1, a cache site collection can also grow by default to a maximum of 100GB.

Whether or not these default settings are appropriate for a Web application depends primarily upon the nature of the site collections housed within the Web application. When site collections contain primarily static documents or content that changes infrequently, it makes sense to allow the cache to grow larger and delete content less often than normal. This maximizes the benefit obtained from caching since document content turns over less frequently.

On the other hand, site collections that experience frequent document turnover and heavy collaboration traffic tend to benefit very little from large cache sizes and long expiration periods. In site collections of this nature, cached content tends to become

stale very quickly. There is little benefit derived from holding onto document resources that may only be good for days or even hours, so maximum cache size is reduced and expiration periods are shortened.

Tip No. 3: Give Yourself Some Warning

You can leverage standard SharePoint site collection features and capabilities to help you out since each Web application's Office Web Apps' cache is a Team Site. One such helpful mechanism is the ability to have an e-mail warning sent to site collection owners once a site collection's size hits a pre-defined threshold. In the case of the Office Web Apps' cache, such a warning could be a cue to increase the maximum size of the cache site collection or perhaps shorten the expiration period for document resources housed within the site collection.

Like the maximum cache size setting described in Tip No. 2, the ability to send e-mail warnings once the cache reaches a threshold is actually tied to SharePoint's site collection quota capabilities. The maximum size of the cache site collection is handled as a storage quota, and the warning threshold maps directly to the quota's warning threshold as shown below in Figure 4. In the case of Figure 4, a maximum cache size of 50GB is in effect for the cache site collection, and the e-mail warning threshold is set for 25GB.

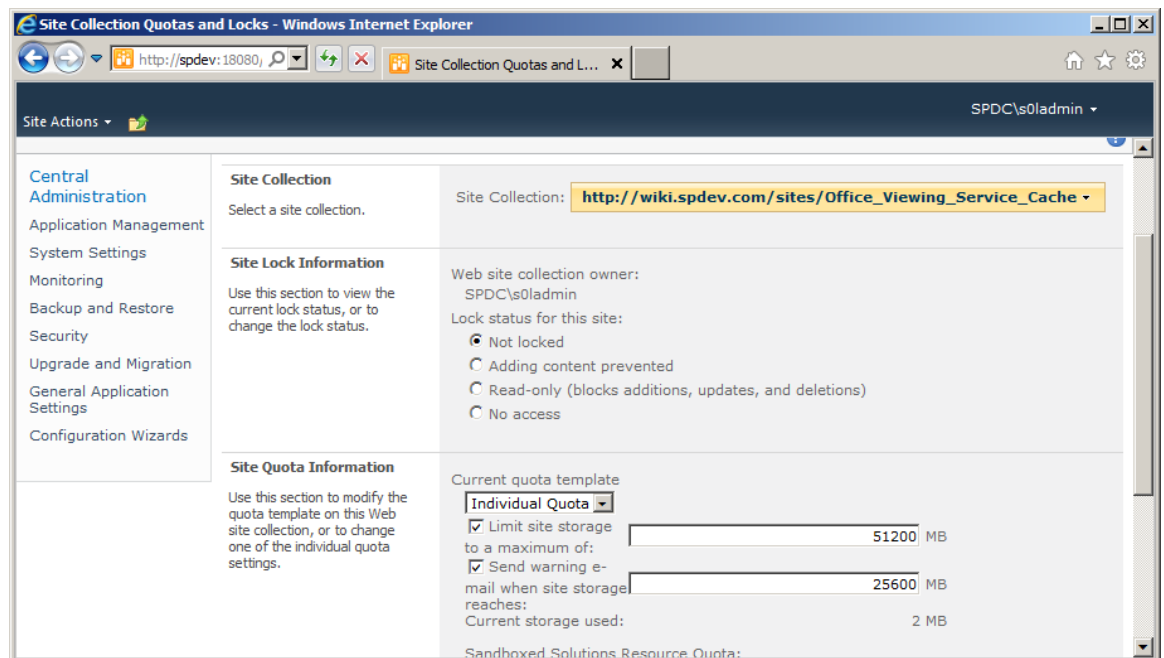


Figure 4: Quota settings for an Office Web Apps' cache site collection

Knobs and Dials

Tips No. 2 and 3 discussed some of the more straightforward Office Web Apps' cache settings that are available to you, but you may be wondering how to actually go about changing them.

The *AdjustOwaCache.ps1* PowerShell script that appears below provides you with an easy way to review and change the settings discussed. Simply save the script, execute it, and supply the URL of the Web application containing the Office Web Apps' cache you'd like to adjust. The script will show you the cache's current settings and give you the opportunity to modify them.

```
<#
.SYNOPSIS
    AdjustOwaCache.ps1
.DESCRIPTION
    Dumps several common OWA cache settings to the console for a selected Web
    application and provides a mechanism for altering the those values
.NOTES
    Author: Sean McDonough
    Last Revision: 08-June-2011
.PARAMETER targetUrl
    A Web application where Office Web Apps are in use
.EXAMPLE
    AdjustOwaCache.ps1 http://www.TargetWebApplication.com
#>
param
(
    [string]$targetUrl = "$(Read-Host 'Target Web application URL [e.g.
http://hostname]')"
)

function AdjustCache($targetUrl)
{
    # Ensure that the SharePoint cmdlets are loaded before continuing
    $spCmdlets = Get-PSSnapin Microsoft.SharePoint.PowerShell -ErrorAction
silentlycontinue
    if ($spCmdlets -eq $Null)
    { Add-PSSnapin Microsoft.SharePoint.PowerShell }

    # Create an easy converter for GB to bytes
    $GBtoBytes = 1024 * 1024 * 1024

    # Get a reference to the cache site collection and extract the values
we'll be
    # working with and (potentially) altering.
    $cacheSite = Get-SPOfficeWebAppsCache -WebApplication $targetUrl -
ErrorAction stop
    $wacSize = $cacheSite.Quota.StorageMaximumLevel / $GBtoBytes
    $wacWarn = $cacheSite.Quota.StorageWarningLevel / $GBtoBytes
    $wacExpire = 30
    if ($cacheSite.RootWeb.Properties.Contains("waccacheexpirationperiod"))
    { $wacExpire = $cacheSite.RootWeb.Properties["waccacheexpirationperiod"] }
    Write-Host "Current OWA cache values for '$targetUrl'"
    Write-Host "- Maximum Cache Size (GB): $wacSize"
    Write-Host "- Warning Threshold (GB): $wacWarn"
    Write-Host "- Expiration Period (Days): $wacExpire"

    # Give the user the option to make changes.
    $yesOrNo = Read-Host "Would you like to change one or more values? [y/n]"
    if ($yesOrNo -eq "y")
    {
```

```
[Int64]$newWacSize = Read-Host "- Maximum Cache Size (GB)"
Write-Host "- Warning Threshold (GB)"
[Int64]$newWacWarn = Read-Host " (supply 0 for no warning)"
[int]$newWacExpire = Read-Host "- Expiration Period (Days)"

# Convert GB values to bytes and set the cache
$newWacSize = ($newWacSize * $GBtoBytes)
$newWacWarn = ($newWacWarn * $GBtoBytes)
Set-SPOfficeWebAppsCache -WebApplication $targetUrl -
ExpirationPeriodInDays $newWacExpire -MaxSizeInBytes $newWacSize -
WarningSizeInBytes $newWacWarn -ErrorAction stop
}

# Abort script processing in the event an exception occurs.
trap
{
    Write-Warning "`n*** Script execution aborting. See below for
problem encountered during execution. ***"
    $_.Message
    break
}

# Launch script
AdjustCache $targetUrl
```

Conclusion

Office Web Apps is a powerful addition to SharePoint 2010. It paves the way for greater collaboration on Office documents without requiring the Microsoft Office suite of client applications. The Office Web Apps' cache is an important part of the larger Office Web Apps equation, and the cache is generally pretty good about taking care of itself. It is still a good idea to relocate the cache from its default location. At the same time, a little bit of tuning and e-mail alerting can go a long way toward ensuring that the cache operates optimally in your environment.

About the Author

Sean McDonough is a Product Manager for SharePoint Products at Idera, a Microsoft gold certified partner and creator of tools for SharePoint, SQL Server, PowerShell and Dynamics. As a consultant, Sean has worked with a number of Fortune 500 companies to architect, implement, troubleshoot, tune, and customize their SharePoint environments. Sean is an MCPD, an MCTS, and the co-author of the "SharePoint 2007 Disaster Recovery Guide" (<http://tinyurl.com/SPDRBook>) and the "SharePoint 2010 Disaster Recovery Guide" (<http://tinyurl.com/SPDRBook2010>). He can be reached through his blog (<http://SharePointInterface.com>) or Twitter (@spmcdonough).

About Idera

Idera provides tools for Microsoft SQL Server, SharePoint and PowerShell management and administration. Our products provide solutions for performance monitoring, backup and recovery, security and auditing and PowerShell scripting. Headquartered in Houston, Texas, Idera is a Microsoft Gold Partner and has over 5,000 customers worldwide.

For more information, or to download a free 14-day full-functional evaluation copy of any of Idera's tools for SQL Server, SharePoint or PowerShell, please visit www.idera.com.