Gallery Server - Fixed Defect Report

Version 4.4.1

Defect: Error during file replacement: "Invalid file extension uploaded"

ID: 255

Created: 07-Nov-17

Closed: 15-Nov-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: The following error may occur when a user uploads a replacement file:

"The following event occurred in the HTTP upload handler: Invalid file extension uploaded.; File Name: 4 coupes.jpg HTTP status code: 100"

Resolution: This will occur when the setting AllowUnspecifiedMimeTypes is set to true. Beginning in 4.0, this setting was automatically set to false and no longer visible in the UI. However, one customer reported it was true. It is not known whether the customer manually changed it in the database or there is an issue in the upgrade script. Regardless, Gallery Server should still be able to handle it.

The code was modified so that a user can upload a replacement file when AllowUnspecifiedMimeTypes is true.

Defect: Error "Cannot add null to an existing MediaObjectProfileCollection"

ID: 254

Created: 07-Nov-17

Closed: 15-Nov-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: A user reported the following error:

Cannot add null to an existing MediaObjectProfileCollection. Items.Count = 146
Parameter name: item

When the user looked at the relevant MediaObjectProfiles record in the UserGalleryProfile table, they noticed a null value that was causing the issue:
[…,{“Id”:18493,”Rating”:”2.5”},{“Id”:18512,”Rating”:”3”},NULL]

Resolution: It is unknown how the null value got into the database. Preventing that is the ideal solution, but a careful study of the code did not reveal any paths where this is possible. However, I discovered that if a null was in the database, we could gracefully ignore it with a simple change, so I did that.

Defect: Delay when assigning thumbnails in large gallery

ID: 257

Created: 21-Nov-17

Closed: 28-Nov-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: In very large galleries, there can be a long delay when assigning a thumbnail or deleting the media asset that serves as the thumbnail for an album.

Resolution: This occurs because the album save behavior clears the gallery cache after every save. This can be expensive to rebuild and is unnecessary during a simple operation such as updating a thumbnail. The code was modified so that rather than using the album save mechanism, the thumbnail routine directly calls the AlbumRepository to persist the new thumbnail ID, then purges only the affected album from cache, leaving the gallery cache untouched.

Defect: Page may be reassigned to another gallery

ID: 258

Created: 28-Nov-17

Closed: 28-Nov-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: In certain cases, the gallery assigned to a particular page (e.g. default.aspx) may be reassigned to another gallery. Steps to reproduce:

* Start with a large multi-gallery database that, due to its size, requires several seconds to delete an album.
* In one browser tab, navigate to a page pointing to one gallery. In another browser tab, navigate to a second page that points to a separate gallery in the same application.
* In the first tab, delete a test album.
* In the second tab, navigate to an album. If you achieved the timing such that you navigated to the second page after the first page fired Factory.LoadGalleries, but before it finished, the second page may fail to find the associated gallery (because it hasn't yet been loaded from the DB). When this happens, code in the GalleryPage.GalleryId getter thinks the gallery doesn't exists and reassigns the page to the first non-template gallery in the DB.

Resolution: The lock mechanism in Factory.LoadGalleries() was faulty in that it assumed the galleries collection was filled any time its count was greater than zero.

I changed it to use a semaphore variable (galleriesLoaded) that is set to true only when the galleries have finished loading from the DB.

Defect: Unnecessary saving of album and purging of cache

ID: 260

Created: 29-Nov-17

Closed: 06-Dec-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: When albums are inflated from a CacheItemAlbum instance, it's HasChanges property is true, causing unnecessary persisting to the database and purging of caches when in fact no changes have been made.

Resolution: Updated logic so HasChanges is set to false.

Defect: Error "Invalid state of GalleryServerRole instance"

ID: 259

Created: 29-Nov-17

Closed: 06-Dec-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: In some cases, the following error can occur:

"Invalid state of GalleryServerRole instance: The AllAlbumIds property has a count of zero but the RootAlbumIds has a count of {_rootAlbumIds.Count}. This occurred despite called the Inflate method."

Resolution: This bug has the same root cause as bug # 258 - a faulty locking mechanism in Factory.LoadGalleries().

Defect: Confusing message on Media Asset Types page

ID: 256

Created: 15-Nov-17

Closed: 07-Dec-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: The message at the top right of the Media Asset Types page says it applies to all galleries, but this is only true for the list of MIME types. Whether or not each is enabled is stored at the gallery level. This message should be clarified.

A user reported the confusion here: https://galleryserverpro.com/forum/gallery-server-support/media-asset-types-settings-do-not-apply-to-all-galleries/

Resolution: Removed the message at the top right of the page saying that it applies to all galleries. The tooltip next to Media Asset Types contains text explaining how the settings apply and that is sufficient.

Defect: Some functions don't work when using the encrypted media URL

ID: 261

Created: 07-Dec-17

Closed: 07-Dec-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: If a user has enabled encrypted media URLs by setting EncryptMediaObjectUrlOnClient to True in the gsp.AppSetting table, some functions don't work:

* Image link breaks after rotating or flipping image
* Downloading a single asset fails
* Empty album thumbnail image broken when creating an album

Resolution: Modified the encryption algorithm so that the encrypted URL is attached as a query string value assigned to a "q" query string parameter. For example, in 4.4.0 and earlier the encrypted URL may look like this:

getmedia.ashx?PneHH0S5VrXVtZWMki2k867KRGyCExF7

Now it looks like this:

getmedia.ashx?q=PneHH0S5VrXVtZWMki2k867KRGyCExF7

Besides fixing the broken functionality, it provides support for a user to add custom query string parameters for SEO purposes. For example, a user can create an URL like this:

getmedia.ashx?q=PneHH0S5VrXVtZWMki2k867KRGyCExF7&part_number=872426

Defect: Optimization: Save album during sync only when something has changed

ID: 262

Created: 07-Dec-17

Closed: 07-Dec-17

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: During a synchronization, each album is persisted to the data store. When this happens, the saved event fires, which causes the gallery cache to be purged. This is an expensive cache to rebuild because it stores a flattened list of all albums in each gallery.

If an album has not changed, don't call the save method, thus preserving the cache.

Resolution: Made two changes:

* Modified Album.VerifyThumbnailIsInflated so that it doesn't change Album.HasChanges to true.
* Modified Album.Sort so that the save method is called only when HasChanges is true.

Defect: Role/album relationships may disappear

ID: 263

Created: 03-Jan-18

Closed: 03-Jan-18

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: In multiple gallery scenarios, albums may be removed from one or more roles at seemingly random times.

Resolution: This bug has the same root cause as bug # 258 - a faulty locking mechanism in Factory.LoadGalleries().

Defect: Optimization: Uploading images may be slow in large galleries

ID: 264

Created: 03-Jan-18

Closed: 03-Jan-18

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: When media files are added to a gallery, a routine runs that checks if the containing album needs a thumbnail image, adding it if necessary. When a thumbnail image is assigned, the gallery cache is cleared, which is expensive to recalculate. It would be better if we can avoid clearing this cache.

Resolution: The code was modified so that rather than using the album save mechanism, the thumbnail routine directly calls the AlbumRepository to persist the new thumbnail ID, then purges only the affected album from cache, leaving the gallery cache untouched.

Defect: Editing an image may cause error: Cannot create a file when that file already exists

ID: 265

Created: 04-Jan-18

Closed: 04-Jan-18

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: A user reported that rotating an image caused the original image to disappear and the following entry to be logged to the event log:

Error: Cannot create a file when that file already exists.
Event Summary
Url https://www.<site>.net/api/galleryitems/rotateflip?rotateFlip=4&viewSize=2
Timestamp 12/31/2017 8:11:19 PM
Exception Type System.IO.IOException
Message Cannot create a file when that file already exists.

Stack Trace at System.IO.__Error.WinIOError(Int32 errorCode, String maybeFullPath)
at System.IO.File.InternalMove(String sourceFileName, String destFileName, Boolean checkHost)
at GalleryServer.Business.Metadata.ImageMetadataReadWriter.ReplaceFileSafely(String sourceFilePath, String destFilePath)
at GalleryServer.Business.Metadata.ImageMetadataReadWriter.PersistMetaValue(MetadataItemName metaName, MetaPersistAction persistAction)
at GalleryServer.Business.MediaObjectSaveBehavior.Save()
at GalleryServer.Business.GalleryObject.Save()
at GalleryServer.Web.Controller.GalleryObjectController.RotateFlip(GalleryItem[] galleryItems, MediaAssetRotateFlip rotateFlip, DisplayObjectType viewSize)
at GalleryServer.Web.Api.GalleryItemsController.RotateFlip(GalleryItem[] galleryItems, MediaAssetRotateFlip rotateFlip, DisplayObjectType viewSize)

Data Version: 4.4.0

Resolution: Typically this error should not occur, as Gallery Server deletes the original before moving the newly created image into the directory where the original used to be. However, according to StackOverflow (https://stackoverflow.com/questions/48067276/seemingly-impossible-error-cannot-create-a-file-when-that-file-already-exists), there can be circumstances where the file is not deleted right away, such as with certain antivirus software.

The fix was to create a helper function named MoveFileSafely that moves a file in a robust manner with retry and rollback functionality. If the file we are moving will overwrite an existing file, it makes a backup of this file by moving it to the App_Data_Temp directory. Then the source file is moved. If an IOException occurs, the function waits 500 ms and tries again. If it fails 10 times in a row, it moves the original file back and throws an exception, resulting in no loss of the original file. If an exception other than IOException occurs while moving the file, we restore the original and fail immediately. In both scenarios a log entry is recorded.

All instances of File.Move were examined in the code base and in all but one case they were replaced with a call to the new safe move function. Therefore this change may increase stability in several areas where file move operations occurred, not just the reported area.

I also fixed a minor bug in ImageMetadataReadWriter.PersistMetaValue() where a temporary file may not be deleted from App_Data_Temp.

Defect: Error "This codec does not support the specified property"

ID: 266

Created: 05-Jan-18

Closed: 05-Jan-18

State: Done

Area: Gallery Server

Iteration: 4.4.1

Issue Detail: Editing an image can sometimes result in a false error being logged in the event log and sent to the administrators if error reporting is enabled. The UI does not indicate an error so this only appears to admins. However, there is a performance cost in recording the error and that is visible to users.

The error that occurs looks like this:
System.NotSupportedException: This codec does not support the specified property.

at System.Windows.Media.Imaging.BitmapEncoder.SaveFrame(SafeMILHandle frameEncodeHandle, SafeMILHandle encoderOptions, BitmapFrame frame)
at System.Windows.Media.Imaging.BitmapEncoder.Save(Stream stream)
at GalleryServer.Business.Metadata.ImageMetadataReadWriter.TryAlternateMethod2OfPersistingMetadata(BitmapDecoder bmpDecoderOriginal, String tmpFilePath, MetadataItemName metaName, MetaPersistAction persistAction)

Meta Save Error: Unable to persist the meta property 'Tags' to the original file for media object ID 11748. It was, however, persisted to the database and is still available for viewing in the gallery.

Resolution: This error occurs when persisting tags and the date picture taken meta value to the media file when the file does not currently have a value for those properties. For example, when rotating an image, Gallery Server creates the rotated image and attempts to copy the meta properties to the new image. For tags, if the image does not have any tags, Gallery Server sets the BitmapMetadata.Keywords property, which is null, to an empty collection. This causes the BitmapMetadata.TrySave method to fail as well as the two backup methods of writing the tags. In reality, there is no need to even try persisting tags because they have not been changed.

Similarly, when rotating an image without a date picture taken property, Gallery Server attempts to set BitmapMetadata.DatePictureTaken, which is null, to an empty string, which then fails. There is no need to update this property in this scenario.

The fix was to modify these two areas so that the Keywords and DatePictureTaken properties are updated only when they have changed.

By not having to log these unnecessary errors, editing performance is increased. In one test of rotating 50 100 KB files that do not have tags or date taken properties, the elapsed time dropped from 1 minute 28 seconds to 5 seconds.

It was noticed that the DatePictureTaken property can never be set to null or an invalid date (.NET throws an exception), so there is no support in the UI to delete the DatePictureTaken property from the media file.