S3-Link Global Apex Methods

S3-Link also provides few global apex methods. You can call those global methods from anywhere like developer console, custom apex class, custom trigger etc. All those global apex methods will be available S3-Link paid edition only. Here is the detail of those global apex methods. All are global static methods.

Class Name: **NEILON.apGlobalUtils**

Method

createS3File(fileContent, fileName, file, folderId)

Signature

global static NEILON__File__c createS3File(Blob fileContent, String fileName, NEILON__File__c file, String folderIdOrFilePath)

Parameters

1. fileContent

   Type: Blob

   Blob content of the file which needs to be uploaded in Amazon S3.

2. fileName

   Type: String

   Name of the file.

3. file

   Type: NEILON__File__c

   File with other field values to be saved in Salesforce.

4. folderIdOrFilePath

   Type: String

   Id of the S3-Folder or Amazon S3 folder path. File will be placed in that folder.

Return Value

Type: NEILON__File__c

Usage

This method uploads single file blob content in Amazon S3 using the path of S3-Folder passed as parameter and inserts the S3-File in Salesforce and returns that file record.

Limit

The file content larger than 12 MB will not be uploaded using this method.

Code

```
// Get bucket to upload file

List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where NEILON__Parent__c = null];

// Create S3 file with custom field values

NEILON__File__c file = new NEILON__File__c();

file.NEILON__Description__c = 'Test Description';

NEILON.apGlobalUtils.createS3File(Blob.valueOf('Test'), ‘textfile.txt’, file, buckets[0].Name + '/test1/test2/textfile.txt');
```

Method

createS3Files(files, fileContentsByKey)

Signature

```
global static List<NEILON__File__c> createS3Files(List<NEILON__File__c> files, Map<String, Blob> fileContentsByKey)
```

Parameters

1. files

   Type: List<NEILON__File__c>

   List of files with all the custom field values and Amazon S3 file path where file needs to be uploaded. NEILON__Amazon_File_Key__c is the field which will be used as the AWS file

location where the actual file will be uploaded.

2. fileContentsByKey

   Type: Map<String, Blob>

   Map of blob contents by NEILON__Amazon_File_Key__c.

Return Value

Type: List<NEILON__File__c>

Usage

This method uploads multiple file blob contents in Amazon S3 using NEILON__Amazon_File_Key__c of NEILON__File__c. It inserts S3-Files in Salesforce and returns those file records.

Limit

Total blob content size must not exceed 12 MB.

Code

```
// Get bucket to upload file

List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where NEILON__Parent__c = null];

// List of files

List<NEILON__File__c> files = new List<NEILON__File__c>();

// File contents by path

Map<String, Blob> fileContentsByKey = new Map<String, Blob>();

// Files with custom field values

NEILON__File__c file1 = new NEILON__File__c(Name = 'Test1.txt');

file1.NEILON__Amazon_File_Key__c = buckets[0].Name+'/test1/test1/Test1.txt';

file1.NEILON__Description__c = 'Test Description 1';

files.add(file1);

NEILON__File__c file2 = new NEILON__File__c(Name = 'Test2.txt');
```

```
file2.NEILON__Amazon_File_Key__c = buckets[0].Name+'/test2/test2/Test2.txt';

file2.NEILON__Description__c = 'Test Description 2';

files.add(file2);

// Prepare content

fileContentsByKey.put(file1.NEILON__Amazon_File_Key__c, Blob.valueOf('Test1'));

fileContentsByKey.put(file2.NEILON__Amazon_File_Key__c, Blob.valueOf('Test2'));

// Create files

NEILON.apGlobalUtils.createS3Files(files, fileContentsByKey);
```

Method

createS3Files(filesByContentDocumentId, folderIdOrFolderPath, contentDocumentIds, isDeleteContentDocument)

Signature

global static List<NEILON__File__c> createS3Files(Map<Id, NEILON__File__c> filesByContentDocumentId, String folderIdOrFolderPath, Set<Id> contentDocumentIds, Boolean isDeleteContentDocument)

Parameters

1. filesByContentDocumentId

   Type: Map<Id, NEILON__File__c>

   Map of S3-Files by Content Document Ids. Pass this map if you want to set some custom field values for S3-File created for Content Document.

2. folderIdOrFolderPath

   Type: String

   Id of the S3-Folder or Amazon S3 folder path. File will be placed in that folder.

3. contentDocumentIds

Type: Set<Id>

Set of Content Document Ids which need to be uploaded in Amazon S3.

4. isDeleteContentDocument

   Type: Boolean

   True, if you want to delete Content Documents once they are uploaded in Amazon S3.

Return Value

Type: List<NEILON__File__c>

Usage

This method uploads Content Documents into specific AWS folders, creates S3-Files in Salesforce and returns a list of S3-Files.

Limit

Content Documents larger than 11 MB will not be uploaded using this method.

Method

createS3LinkFilesForAttachments(attachmentIds)

Signature

global static void createS3LinkFilesForAttachments(Set<Id> attachmentIds)

Parameters

1. attachmentIds

   Type: Set<Id>

   Ids of the Standard Salesforce records which needs to be moved to Amazon S3.

Return Value

Type: void

Usage

This method moves your existing Standard Salesforce attachment record into Amazon S3 and creates S3-File records and links them to respective parent records. Basically to replace “Notes & Attachments” with “S3-Files”. “File Export Configuration” will be used while moving attachments into Amazon S3.

Limit

Attachments larger than 11 MB will not be uploaded using this method.

Code

```
// Create account

Account testAccount = new Account(Name = 'Test Account');

insert testAccount;

// Create attachments

Attachment attachment = new Attachment(Name = 'Test1.pdf', Body = Blob.valueOf('Test'), ParentId = testAccount.Id);

insert attachment;

// Create s3 link files for attachment

NEILON.apGlobalUtils.createS3LinkFilesForAttachments(new Set<Id>{attachment.Id});
```

Method

createS3LinkFilesForAttachments(attachments, isAlwaysDeleteAttachments, isAsyncJob)

Signature

global static void createS3LinkFilesForAttachments(List<Attachment> attachments, Boolean isAlwaysDeleteAttachments, Boolean isAsyncJob)

Parameters

1. attachments

   Type: List<Attachment>

   List of Standard Salesforce attachment records which needs to be moved to Amazon S3.

2. isAlwaysDeleteAttachments

   Type: Boolean

   True, if you want to delete Salesforce attachments once they are uploaded in Amazon S3.

3. isAsyncJob

   Type: Boolean

   True, if you want to start a batch in the backend to upload attachments in Amazon S3.

Return Value

Type: void

Usage

This method moves your existing Standard Salesforce attachment record into Amazon S3 and creates S3-File records and links them to respective parent records. Basically to replace “Notes & Attachments” with “S3-Files”. Attachments will be deleted after moved to Amazon S3 as per the flag passed as parameter.

Limit

Attachments larger than 11 MB will not be uploaded using this method.

Code

```
// Create account

Account testAccount = new Account(Name = 'Test Account');

insert testAccount;

// Create attachments

Attachment attachment = new Attachment(Name = 'Test1.pdf', Body = Blob.valueOf('Test'), ParentId = testAccount.Id);

insert attachment;

// Create s3 link files for attachment

NEILON.apGlobalUtils.createS3LinkFilesForAttachments(new List<Attachment>{attachment}, true, true);
```

Method

createS3LinkFilesForContentVersion(contentDocumentLinkIds, isDeleteOrphanedContentDocuments, isAsyncJob)

Signature

global static void createS3LinkFilesForContentVersion(Set<Id> contentDocumentLinkIds, Boolean isDeleteOrphanedContentDocuments, Boolean isAsyncJob)

Parameters

1. contentDocumentLinkIds

   Type: Set<Id>

   Set of Salesforce Content Document Link Ids.

2. isDeleteOrphanedContentDocuments

   Type: Boolean

   True, if you want to delete Content Documents once they are uploaded in Amazon S3.

3. isAsyncJob

   Type: Boolean

   True, if you want to start a batch in the backend to upload Content Documents in Amazon S3.

Return Value

Type: void

Usage

This method moves your existing Salesforce Content Versions record into Amazon S3 and creates S3-File records and links them to respective parent records. Basically to replace “Files” with “S3-Files”.“File Export Configuration” will be used while moving attachments into Amazon S3.

Limit

Content Documents larger than 11 MB will not be uploaded using this method.

Code

```
// Create account

Account testAccount = new Account(Name = 'Test Account');

insert testAccount;

// Create content version

ContentVersion contentVersion = new ContentVersion();

contentVersion.Description = 'Test';

contentVersion.Title = 'Test1';

contentVersion.OwnerId = UserInfo.getUserId();

contentVersion.VersionData = Blob.valueOf('Test');

contentVersion.PathOnClient = 'Bucket 1/'+contentVersion.Title+'.pdf';

contentVersion.ContentLocation = 'S';

// Insert the new content version for the file type version

insert contentVersion;

// Get the content document id of content version

List<ContentVersion> contentVersions = [Select Id, ContentDocumentId From ContentVersion Where
Id = :contentVersion.Id LIMIT 1];

// Create content document link

ContentDocumentLink contentDocumentLink = new ContentDocumentLink(LinkedEntityId =
account.Id,

ContentDocumentId = contentVersions[0].ContentDocumentId,

ShareType = 'V');

insert contentDocumentLink;

// Create s3 link files for attachment
```

NEILON.apGlobalUtils.createS3LinkFilesForContentVersion(new Set<Id>{contentDocumentLink.Id}, true, true);

Method

getAmazonS3FileMetadata(bucket, fileKey)

Signature

global static Blob getAmazonS3FileContent(String bucket, String fileKey)

Parameters

1. bucket

   Type: String

   Amazon S3 bucket name

2. fileKey

   Type: String

   Amazon S3 file path excluding bucket

Return Value

Type: Map<String, String>

Usage

This method gets base64 file content from Amazon S3 using bucket name and file path

Code

// Get bucket to upload file

List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where NEILON__Parent__c = null];

// Get file content

Map<String, String> fileMatadata = NEILON.apGlobalUtils.getAmazonS3FileMetadata(buckets[0].Name, 'test1/test2/textfile.txt');

Method

getAmazonS3FileContent(bucket, fileKey, versionId)

Signature

global static Blob getAmazonS3FileContent(String bucket, String fileKey, String versionId)

Parameters

1. bucket

   Type: String

   Amazon S3 bucket name

2. fileKey

   Type: String

   Amazon S3 file path excluding bucket

3. versionId

   Type: String

   Amazon S3 File version id

Return Value

Type: Blob

Usage

This method gets the blob content of the file version from Amazon S3 using bucket name and file path and version id.

Limit

The file larger than 12 MB will not be fetched using this method.

Method

createS3Folders(folderPaths)

Signature

global static List<Boolean> createS3Folders(List<String> folderPaths)

Parameters

1. folderPaths

   Type: List<String>

   List of Amazon S3 folder paths (including bucket name) to create folders in Amazon S3

Return Value

Type: List<Boolean>

Usage

This method will create Amazon S3 folders with specified paths

Code

```
// Get bucket to get file keys

List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where NEILON__Parent__c = null];

// Create folder test 1

List<Boolean> status = NEILON.apGlobalUtils.createS3Folders(new List<String>{buckets[0].Name + '/test1');
```

Method

createS3LinkFilesForAmazonFiles(files, validateFilePaths)

Signature

global static List<NEILON__File__c> createS3LinkFilesForAmazonFiles(List<NEILON__File__c> files, Boolean validateFilePaths)

Parameters

1. files

   Type: List<NEILON__File__c>

   List of S3-Files with file path of Amazon S3.

2. validateFilePaths

   Type: Boolean

   True, if the Amazon S3 file path needs to be validated before creating a file in Salesforce.

Return Value

Type: List<NEILON__File__c>

Usage

This method creates S3-Files along with the full folder structure provided in Amazon file path. Basically it will create all required S3-Folders and S3-Files. Make sure you provide the proper value for “Amazon S3 File Key”. You can also validate the path before creating folder structure and files.

Code

```
List<NEILON__File__c> files  = new List<NEILON__File__c>();

NEILON__File__c file = new NEILON__File__c();

file.NEILON__Bucket_Name__c = 'bucketname';

file.NEILON__Amazon_File_Key__c = 'test1/test2/test.png';

files.add(file);

List<NEILON__File__c> newFiles = NEILON.apGlobalUtils.createS3LinkFilesForAmazonFiles(files, false);
```

Method

createS3LinkFoldersForAmazonFolders(folders)

Signature

global static List<NEILON__Folder__c>

createS3LinkFoldersForAmazonFolders(List<NEILON__Folder__c> folders)

Parameters

3. folders

   Type: List<NEILON__Folder__c>

   List of S3-Folders with folder path of Amazon S3.

Return Value

Type: List<NEILON__Folder__c>

Usage

This method creates S3-Folders along with the full folder structure provided in Amazon file path. Basically it will create all required S3-Folders. Make sure you provide the proper value for “Amazon S3 File Key”.

Code

```
List<NEILON__Folder__c> folders  = new List<NEILON__Folder__c>();

NEILON__Folder__c folder = new NEILON__Folder__c();

file.NEILON__Bucket_Name__c = 'bucketname';

file.NEILON__Amazon_File_Key__c = 'test1/test2/test3';

folders .add(folder );

List<NEILON__Folder__c > newFolders =
NEILON.apGlobalUtils.createS3LinkFoldersForAmazonFolders(folders);
```

Method

createAttachmentsForS3Files(fileIds, isAlwaysDeleteFiles, isAsyncJob)

Signature

global static void createAttachmentsForS3Files(Set<Id> fileIds, Boolean isAlwaysDeleteFiles, Boolean isAsyncJob)

Parameters

1. fileIds

   Type: Set<Id>

   Set of S3-File Ids for which Salesforce attachments or content documents needs to be created.

2. isAlwaysDeleteFiles

   Type: Boolean

   True, if you want to delete S3-Files once Attachments / Content Documents are created for S3-Files.

3. isAsyncJob

   Type: Boolean

   True, if you want to start a batch in the backend to create Attachment / Content Documents for S3-Files.

Return Value

Type: void

Usage

This method creates Attachments / Content Documents for S3-Files. It also links attachments and content documents with S3-File's parent record. Below code creates attachments for all the S3-Files related to the account record.

Limit

Attachments will not be created for Amazon S3 file larger than 12 MB.

Code

```
List<NEILON__File__c> files  = [Select Id From NEILON__File__c Where NEILON__Account__c = '0019000001sRVk9'];

Set<Id> fileIds = new Set<Id>();

for(NEILON__File__c file : files){

        fileIds.add(file.Id);
```

}

NEILON.apGlobalUtils.createAttachmentsForS3Files(fileIds, false, true);

Method

getAmazonFilePaths(folderPaths)

Signature

global static Set<String> getAmazonFilePaths(Set<String> folderPaths)

Parameters

1. folderPaths

   Type: Set<String>

   List of folder paths with bucket whose file keys you want to get from Amazon S3

Return Value

Type: Set<String>

Usage

This method will return list of Amazon S3 keys placed in folder paths

Limit

It will return the first 15000 files inside the folders.

Code

// Get bucket to get file keys

List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where NEILON__Parent__c = null];

// Get file keys

Set<String> fileKays = NEILON.apGlobalUtils.getAmazonFilePaths(new Set<String>{buckets[0].Name + '/test1');

Method

importAmazonFiles(folderPaths, isIncludeSubFolders, isAsyncJob, isLogInvalidFiles)

Signature

global static List<NEILO___File__c> importAmazonFiles(Set<String> folderPaths, Boolean isIncludeSubFolders, Boolean isAsyncJob, Boolean isLogInvalidFiles)

Parameters

1. folderPaths

   Type: Set<String>

   List of Amazon S3 folder paths whom you want to sync with Salesforce

2. isIncludeSubFolders

   Type: Boolean

   True, if files inside subfolders also need to be imported

3. isAsyncJob

   Type: Boolean

   True, if this method is being invoked from any batch apex

4. isLogInvalidFiles

   Type: Boolean

   True, if you want to log files and folders with names having special characters(not supported by S3-Link) or names larger than 80 characters.

Return Value

Type: List<NEILO___File__c>

Usage

This method will sync Amazon S3 folders with Salesforce

Limit

It will import the first 15000 files inside the folders.

Code

```
// Get bucket to get file keys

List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where
NEILON__Parent__c = null];

// Sync folder test 1

List<NEILON__File__c> files = NEILON.apGlobalUtils.importAmazonFiles(new
Set<String>{buckets[0].Name + '/test1', true, false, false);
```

Method

copyS3Files(targetFilePathsBySourceFilePath)

Signature

global static Map<String, Boolean> copyS3Files(Map<String, String> targetFilePathsBySourceFilePath)

Parameters

1. targetFilePathsBySourceFilePath

   Type: Map<String, String>

   List of target file paths by source file paths

Return Value

Type: Map<String, Boolean>

Usage

This method will copy the source file into the target file using paths.

Code

```
// Get bucket to get file keys
```

```
List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where
NEILON__Parent__c = null];

// Copy files

Map<String, String> targetFilePathsBySourceFilePath = new Map<String,
String>{buckets[0].Name+'/test1/test1.txt' => buckets[0].Name+'/test1/test2.txt'};

Map<String, Boolean> successStatusBySourceFilePath =
NEILON.apGlobalUtils.copyS3Files(targetFilePathsBySourceFilePath);
```

Method

moveS3Files(targetFilePathsBySourceFilePath)

Signature

global static Map<String, Boolean> moveS3Files(Map<String, String>
targetFilePathsBySourceFilePath)

Parameters

1. targetFilePathsBySourceFilePath

   Type: Map<String, String>

   List of target file paths by source file paths

Return Value

Type: Map<String, Boolean>

Usage

This method will copy the source file into the target file using paths.

Code

```
// Get bucket to get file keys

List<NEILON__Folder__c> buckets = [Select Id, Name From NEILON__Folder__c Where
NEILON__Parent__c = null];

// Copy files
```

```
Map<String, String> targetFilePathsBySourceFilePath = new Map<String,
String>{buckets[0].Name+'/test1/test1.txt' => buckets[0].Name+'/test1/test2.txt'};

Map<String, Boolean> successStatusBySourceFilePath =
NEILON.apGlobalUtils.moveS3Files(targetFilePathsBySourceFilePath);
```

Method

createS3LinkFilesForEmailMessageAttachments(emailMessageIds)

Signature

global static void createS3LinkFilesForEmailMessageAttachments(Set<Id> emailMessageIds)

Parameters

1. emailMessageIds

   Type: Set<Id>

   Set of EmailMessage ids

Return Value

Type: void

Usage

This method moves Standard Salesforce attachments linked to EmailMessages into Amazon S3 and creates S3-File records and links them to respective EmailMessages.

Limit

Email attachments larger than 11 MB will not be uploaded using this method.

Method

createS3LinkFilesForEventAttachments(eventIds)

Signature

global static void createS3LinkFilesForEventAttachments(Set<Id> eventIds)

Parameters

1. eventIds

   Type: Set<Id>

   Set of Event ids

Return Value

Type: void

Usage

This method moves Standard Salesforce attachments linked to Event into Amazon S3 and creates S3-File records and links them to respective Event.

Limit

Event attachments larger than 11 MB will not be uploaded using this method.

Method

createS3LinkFilesForTaskAttachments(taskIds)

Signature

global static void createS3LinkFilesForTaskAttachments(Set<Id> taskIds)

Parameters

1. taskIds

   Type: Set<Id>

   Set of Task ids

Return Value

Type: void

Usage

This method moves Standard Salesforce attachments linked to EmailMessages into Amazon S3 and

creates S3-File records and links them to respective Tasks.

Limit

Task attachments larger than 11 MB will not be uploaded using this method.

Method

createS3LinkFilesForReports(salesforceReportIds)

Signature

global static void createS3LinkFilesForReports(Set<String> salesforceReportIds)

Parameters

1. salesforceReportIds

   Type: Set<String>

   Ids of Salesforce reports.

Return Value

Type: void

Usage

This method generates EXCEL or CSV for Salesforce reports and moves it to Amazon S3. “Salesforce Report Export Configuration” will be used.

Limit

Salesforce reports larger than 11 MB will not be uploaded using this method.

Method

buildFolderArchitecture(parentId)

Signature

global static NEILON__Folder__c buildFolderArchitecture(Id parentId)

Parameters

1. parentId

   Type: Id

   Id of the Salesforce standard or custom object record for which folder default folder structure needs to be created.

Return Value

Type: NEILON__Folder__c

Usage

This method creates a default folder structure like S3 bucket / Accounts / Test Account and returns home S3-Folder (i.e Test Account) for the record.

Method

buildFolderArchitecture(parentIds, parentNameById)

Signature

global static Map<Id,NEILON__Folder__c> buildFolderArchitecture(Set<Id> parentIds, Map<Id,String> parentNameById)

Parameters

1. parentIds

   Type: Set<Id>

   Set of Salesforce standard or custom object record Ids for which folder default folder structures need to be created.

2. parentNameById

   Type: Map<Id,String>

   Map of home folder names by Salesforce standard or custom object record Ids for which folder default folder structures need to be created.

Return Value

Type: Map<Id,NEILON__Folder__c>

Usage

This method creates default folder structures like S3 bucket / Accounts / Test Account for multiple records and returns home S3-Folders for each record.

Method

copyFiles(fileIds, destinationFolderId, isCopyVersions, isAsyncJob)

Signature

global static void copyFiles(Set<String> fileIds, String destinationFolderId, Boolean isCopyVersions, Boolean isAsyncJob)

Parameters

1. fileIds

   Type: Set<String>

   Set of S3-File Ids.

2. destinationFolderId

   Type: String

   Id of S3-Folder.

3. isCopyVersions

   Type: Boolean

   True, if you want to copy file versions along with files.

4. isAsyncJob

   Type: Boolean

   This is not used anymore. You can set any values for this parameter.

Return Value

Type: void

Usage

This method copies all the files into the destination folder. This will only copy AWS files which are linked in Salesforce.

Method

copyFolders(folderIds, destinationFolderId, isCopyVersions)

Signature

global static void copyFolders(Set<String> folderIds, String destinationFolderId, Boolean isCopyVersions)

Parameters

1. folderIds

   Type: Set<String>

   Set of S3-Folder Ids.

2. destinationFolderId

   Type: String

   Id of S3-Folder.

3. isCopyVersions

   Type: Boolean

   True, if you want to copy file versions along with files inside folders.

Return Value

Type: void

Usage

This method copies folders into the destination folder. This will only copy AWS folders and files which are linked in Salesforce.

Method

createS3LinkFilesForEventLogFiles(eventLogFiles)

Signature

global static void createS3LinkFilesForEventLogFiles(List<SObject> eventLogFiles)

Parameters

1. eventLogFiles

   Type: List<SObject>

   List of EventLogs.

Return Value

Type: void

Usage

This method moves EventLogs of your Salesforce org into Amazon S3 and creates S3-File records.

Limit

Event logs larger than 11 MB will not be uploaded using this method.

Method

createS3LinkFilesForEmailFiles(emailFiles)

Signature

global static void createS3LinkFilesForEmailFiles(List<Messaging.InboundEmail.BinaryAttachment> emailFiles)

Parameters

5. emailFiles

   Type: List<Messaging.InboundEmail.BinaryAttachment>

   List of inbound email attachments.

Return Value

Type: void

Usage

This method moves Inbound Email attachments into Amazon S3 and creates S3-File records for those

attachments.

Limit

Inbound Email attachments larger than 11 MB will not be uploaded using this method.

Method

clearAWSKeys()

Signature

global static void clearAWSKeys()

Return Value

Type: void

Usage

This method clears AWS Access Key Id, AWS Secret Key & KMS Master Key in S3-Link Administration > System Configuration.

Method

setAWSKeys(accessKeyId, secretKey, kmsMasterKey)

Signature

global static void setAWSKeys(String accessKeyId, String secretKey, String kmsMasterKey)

Parameters

1. accessKeyId

   Type: String

   AWS Access Key Id to use in Salesforce.

2. secretKey

   Type: String

   AWS Secret Key to use in Salesforce.

3. kmsMasterKey

   Type: String

   KMS Master Key to use in Salesforce.

Return Value

Type: void

Usage

This method sets AWS Access Key Id, AWS Secret Key & KMS Master Key in S3-Link Administration > System Configuration.

Method

schedulePresignedURLJob(cronExpression, isAbort)

Signature

global static void schedulePresignedURLJob(String cronExpression, Boolean isAbort)

Parameters

1. cronExpression

   Type: String

   Cron expression to set frequency of schedule job. If null, it will just run the job at that moment. It will not schedule the job.

2. isAbort

   Type: Boolean

   True, if you want to schedule multiple instances of the same apex class.

Return Value

Type: void

Usage

This method schedules the apex job to reset Pre Signed URLs.

Method

scheduleExportReportsJob(cronExpression, isAbort)

Signature

global static void scheduleExportReportsJob(String cronExpression, Boolean isAbort)

Parameters

1. cronExpression

   Type: String

   Cron expression to set frequency of schedule job. If null, it will just run the job at that moment. It will not schedule the job.

2. isAbort

   Type: Boolean

   True, if you want to schedule multiple instances of the same apex class.

Return Value

Type: void

Usage

This method schedules the apex job to auto upload Salesforce reports in Amazon S3.

Method

scheduleAutoImportFilesJob(cronExpression, isAbort)

Signature

global static void scheduleAutoImportFilesJob(String cronExpression, Boolean isAbort)

Parameters

1. cronExpression

   Type: String

   Cron expression to set frequency of schedule job. If null, it will just run the job at that moment. It will not schedule the job.

2. isAbort

   Type: Boolean

   True, if you want to schedule multiple instances of the same apex class.

Return Value

Type: void

Usage

This method schedules the apex job to auto link existing AWS files with Salesforce.

Method

createS3LinkFilesForContentDocuments(contentDocumentIds, isAsyncJob)

Signature

global static Map<String, String> createS3LinkFilesForContentDocuments(Set<Id> contentDocumentIds, Boolean isAsyncJob)

Parameters

1. contentDocumentIds

   Type: Set<Id>

   Set of Content Document Ids which need to be uploaded in Amazon S3.

2. isAsyncJob

   Type: Boolean

   True, if you want to start a batch in the backend to upload Content Documents in Amazon S3.

Return Value

Type: Map<String, String>

Usage

This method will upload Content Documents in the “Salesforce Content Documents” folder inside Amazon S3 bucket and return a map of file ids by content document ids.

Limit

Content Documents larger than 11 MB will not be uploaded using this method.

Method

syncS3FileVersions(fileIds, isAsyncJob)

Signature

global static void syncS3FileVersions(Set<Id> fileIds, Boolean isAsyncJob)

Parameters

1. fileIds

   Type: Set<Id>

   Set of S3-File Ids whose versions needs to be synced.

2. isAsyncJob

   Type: Boolean

   True, if you want to start a batch in the backend to sync versions for S3-Files.

Return Value

Type: void

Usage

This method syncs AWS file versions with Salesforce.

Code

```
List<NEILON__File__c> files  = [Select Id From NEILON__File__c Where NEILON__Account__c = '0019000001sRVk9'];

Set<Id> fileIds = new Set<Id>();

for(NEILON__File__c file : files){

        fileIds.add(file.Id);

}

NEILON.apGlobalUtils.syncS3FileVersions(fileIds, true);
```

Method

deleteS3FileVersions(versionIds)

Signature

global static void deleteS3FileVersions(Set<Id> versionIds)

Parameters

1. versionIds

   Type: Set<Id>

   Set of S3-File Version Ids which needs to be deleted from AWS.

Return Value

Type: void

Usage

This method deletes S3-File Versions and related file versions in AWS.

Method

getPresignedURL(filePath, expireInSecond)

Signature

global static String getPresignedURL(String filePath, Long expireInSecond)

Parameters

1. filePath

   Type: String

   Amazon S3 file path including bucket

2. expireInSecond

   Type: Long

   Expiry duration of Presigned URL in second

Return Value

Type: String

Usage

This method returns a Pre Signed URL for Amazon S3 file. The Presigned URL will auto expire after the duration passed as the parameter.

Method

resetConfigurations()

Signature

global static void resetConfigurations()

Return Value

Type: void

Usage

This method will reset all S3-Link configurations and restart all scheduled jobs. Note: This will not reset object level S3-Link configurations.

Method

grantS3LinkAccess(usersToGrantAccess)

Signature

global static void grantS3LinkAccess(List<User> usersToGrantAccess)

Parameters

1. usersToGrantAccess

   Type: List<User>

   List of users to assign S3-Link permission sets.

Return Value

Type: void

Usage

This method assigns S3-Link permission sets to users so they can upload, download and view files.

Method

grantS3LinkAccess(userIdsToGrantAccess)

Signature

global static void grantS3LinkAccess(Set<String> userIdsToGrantAccess)

Parameters

1. userIdsToGrantAccess

   Type: Set<String>

   Set of user ids to assign S3-Link permission sets.

Return Value

Type: void

Usage

This method assigns S3-Link permission sets to users so they can upload, download and view files.