# Office AMS: Large file upload with CSOM

|  |  |
| --- | --- |
| Summary: | Applies to: |
| This simple sample shows how to upload a large file into SharePoint using client side object model | * Office 365 Multi-Tenant (MT) * Works also with small adjustments   + Office 365 Dedicated (D)   + SharePoint 2013 on-premises |
| Solution: | Core.LargeFileUpload, version 1.0 |
| Author: | Vesa Juvonen (Microsoft) |
| //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  // THIS CODE IS PROVIDED \*AS IS\* WITHOUT WARRANTY OF  // ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING ANY  // IMPLIED WARRANTIES OF FITNESS FOR A PARTICULAR  // PURPOSE, MERCHANTABILITY, OR NON-INFRINGEMENT.  //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* | |

# General comments

This scenario shows the right approach for uploading files to the SharePoint using client side object model. There are few different ways to achieve this task and how to avoid the 2 MB file upload limit.

# Scenario: Upload large files to SharePoint using CSOM

This scenario shows how to upload large files to SharePoint using CSOM. There are different approaches which has size limitations.

## File size limit approach

When using this approach you will receive an exception if file size is larger than 2 MB (2097152 bytes). This is due to the way the information is sent from client to the server. The issue is caused by the fact that we are using the *Content* property of the *FileCreationInformation* object.

public void UploadDocumentContent(ClientContext ctx, string libraryName, string filePath)

{

Web web = ctx.Web;

// Ensure that target library exists, create if is missing

if (!LibraryExists(ctx, web, libraryName))

{

CreateLibrary(ctx, web, libraryName);

}

FileCreationInformation newFile = new FileCreationInformation();

newFile.Content = System.IO.File.ReadAllBytes(filePath);

newFile.Url = System.IO.Path.GetFileName(filePath);

// Get instances to the given library

List docs = web.Lists.GetByTitle(libraryName);

// Add file to the library

Microsoft.SharePoint.Client.File uploadFile = docs.RootFolder.Files.Add(newFile);

ctx.Load(uploadFile);

ctx.ExecuteQuery();

}

## Large file handling – option 1

This alternative method, uses the *ContentStream* property of the *FileCreationInformation* object. This is valid approach for uploading large files sizes to SharePoint.

public void UploadDocumentContentStream(ClientContext ctx, string libraryName, string filePath)

{

Web web = ctx.Web;

// Ensure that target library exists, create if is missing

if (!LibraryExists(ctx, web, libraryName))

{

CreateLibrary(ctx, web, libraryName);

}

using (FileStream fs = new FileStream(filePath, FileMode.Open))

{

FileCreationInformation flciNewFile = new FileCreationInformation();

// This is the key difference for the first case - using ContentStream property

flciNewFile.ContentStream = fs;

flciNewFile.Url = System.IO.Path.GetFileName(filePath);

flciNewFile.Overwrite = true;

List docs = web.Lists.GetByTitle(libraryName);

Microsoft.SharePoint.Client.File uploadFile = docs.RootFolder.Files.Add(flciNewFile);

ctx.Load(uploadFile);

ctx.ExecuteQuery();

}

}

## Large file handling – option 2

This is an alternative method, which uses *SaveBinaryDirect* method *File* object. This is a valid approach for uploading large file sizes to SharePoint.

public void SaveBinaryDirect(ClientContext ctx, string libraryName, string filePath)

{

Web web = ctx.Web;

// Ensure that target library exists, create if is missing

if (!LibraryExists(ctx, web, libraryName))

{

CreateLibrary(ctx, web, libraryName);

}

using (FileStream fs = new FileStream(filePath, FileMode.Open))

{

Microsoft.SharePoint.Client.File.SaveBinaryDirect(ctx, string.Format("/{0}/{1}", libraryName, System.IO.Path.GetFileName(filePath)), fs, true);

}

}