

Autodesk Construction Cloud Backup

ACCBBackup is a C# console application built to backup all Autodesk Construction Cloud/BIM360 projects in your account via the [Autodesk Forge API](#).

It can be run as a one-off or scheduled via a script. It was designed to fulfill a need that Veeam Backup didn't support. The only products on the market at the time were \$6k AUD per year, took 15-20 hours to backup what ACCBackup does in 3 hours, and required each project to be manually configured as a separate task/job.

By default, ACCBackup will backup all projects in your account.

It is used daily to backup 100~ projects (100 GB~) in 3 hours, 30 minutes.

Prerequisites

1. Create an app via <https://forge.autodesk.com/>. At the time of writing all the APIs the app we are creating will use are free.
2. You will need the Client ID, Client Secret, App Name = "Backup", Description = "Backup", Callback URL: point to your business URL but it is not important (this is for 3 legged auth and we only use 2 legged), and "APIs this app will be able to access": Autodesk Construction Cloud API, BIM 360 API, Data Management API.
3. Enable Custom Integrations on your Autodesk admin account (if you don't see the tab email autodesk support as [this article](#) describes -- just say you want to build a custom integration)
4. From Accounts Admin -> Settings -> Custom Integrations
5. Add Custom Integration -> provide Client ID and App Name.
6. Once this is done you will have a forge app with clientid & clientsecret that will be approved for your accountid.

NOTE: The above steps (roughly) mapped onto pure Autodesk Construction Cloud (the new Autodesk model/signup process) projects and the backup still worked.

Install

- Install latest [.NET Runtime 6.X.X](#) -- only the console version is required ("The .NET Runtime contains just the components needed to run a console app. Typically, you'd also install either the ASP.NET Core Runtime or .NET Desktop Runtime.")
- Download the .zip via [the releases tab](#)
- Unzip and run (see examples below)

Examples

ACCBBackup.exe --help

| | |
|-------------------|---------------------------------------------------|
| --backupdirectory | Required. Backup directory. |
| --clientid | Required. Client ID of Autodesk Forge app. |
| --clientsecret | Required. Client secret of Autodesk Forge app. |
| --accountid | Required. Autodesk Construction Cloud account ID. |

--hubid Autodesk Construction Cloud HubId, defaults to b.AccountId. See https://forge.autodesk.com/en/docs/data/v2/reference/http/hubs-hub_id-projects-project_id-GET/ for more information.

--maxdegreeofparallelism (Default: 8) Number of files to download in parallel.

--retryattempts (Default: 15) Amount of times to retry when there are errors communicating with the Autodesk API.

--initialretryinseconds (Default: 2) Each subsequent retry is RetryAttempt# * InitialRetryInSeconds. Settings of 15 RetryAttempts with InitialRetryInSeconds 2 totals 4 minutes of retrying.

--dryrun (Default: false) Backup will only create 0 byte placeholder files. If not specified, will still create full file structure.

--backupstorotate (Default: 1) Number of backups to maintain.

--projectstobackup Comma separated list of project names OR project ids to backup. If not specified, all projects are backed up.

--projectstoexclude Comma separated list of project names OR project ids to exclude from backup. If not specified, all projects in 'projectstobackup'.

--debuglogging (Default: false) Enable debug logging. Verbose.

--tracelogging (Default: false) Enable trace logging. Extremely verbose.

--smtpport (Default: 25) Backup summary notification email: SMTP port.

--smtphost Backup summary notification email: SMTP server name.

--smtpfromaddress Backup summary notification email: SMTP from address.

--smtpfromname Backup summary notification email: SMTP from name.

--smtptoaddress Backup summary notification email: SMTP to address.

--smtpusername Backup summary notification email: SMTP username.

--smtppassword Backup summary notification email: SMTP password.

--smtpenablessl (Default: false) Backup summary notification email: SMTP over SSL.

--help Display this help screen.

--version Display version information.

Basic:

```
ACCBBackup.exe --backupdirectory "C:\ACCBBackup" --clientid "DR04zxzt71HCkL34cn2tAUSRS00QGaRT"
```

Basic with email summary:

```
-ACCBBackup.exe -backupdirectory "C:\ACCBBackup" --clientid "DR04zxzt71HCkL34cn2tAUSRS00QGaRT"
```

Example of a daily backup with 5 daily backup folders rotating:

```
ACCBBackup.exe --backupdirectory "C:\ACCBBackup\Daily" --clientid "DR04zxzt71HCkL34cn2tAUSRS00QGaRT"
```

Example of a weekly backup with 4 weekly backup folders rotating:

```
ACCBakup.exe --backupdirectory "C:\ACCBakup\Weekly" --clientid "DRO4zxzt71HCkL34cn2tAUSRS0C
```

Example of a monthly backup with 12 monthly backup folders rotating:

```
ACCBakup.exe --backupdirectory "C:\ACCBakup\Monthly" --clientid "DRO4zxzt71HCkL34cn2tAUSRS0C
```

Example backing up two projects (you can use project name or project id, as project name can change it is recommended to use project id)

```
ACCBakup.exe --projectstobackup "Test Project, 7468066c-53e6-4acf-8086-6b5fce0048d6" --backu
```

Example backing all projects BUT two projects

```
ACCBakup.exe --projectstoexclude "Test Project, 7468066c-53e6-4acf-8086-6b5fce0048d6" --bac
```

Debugging

- Log files are located in ACCBackup.exe\Logs
- Try running with --hubid "YourAccountldHere" OR --hubid "b.YourAccountldHere" (Autodesk is migrating from BIM360 to 'Autodesk Construction Cloud' but internally, at the moment, the API hubid needs b.Accountld (b. = BIM360).
- First step is to run with the --debuglogging flag
- If the --debuglogging flag doesn't point you in the right direction the --tracelogging flag should be used.
- If all else fails log an issue via the repo with a link to your --tracelogginng log file and I will investigate.
- Or/and email bold.oil7762@fastmail.com with your log file attached.

ApiClient

ACCBakup is designed around an ApiClient class library I wrote that does all the heavy lifting. This makes it very easy for you to take the class library and do your own thing with it if needed. Below is a copy of ApiClient.UsageExample:Program.cs

```
using ACC.ApiClient;
using ACC.ApiClient.Entities;
using Library.Logger;
using Library.SecretsManager;
using File = ACC.ApiClient.Entities.File;
using LogLevel = Library.Logger.LogLevel;

/*
 * Required parameters
 */
string clientId = SecretsManager.GetEnvironmentVariableOrDefaultTo("acc:clientid", "InvalidCl
string clientSecret = SecretsManager.GetEnvironmentVariableOrDefaultTo("acc:clientsecret", "I
string accountId = SecretsManager.GetEnvironmentVariableOrDefaultTo("acc:accountid", "Invalid

/*
 * Building ApiClient with default parameters
 */
ApiClient defaultApiClient = TwoLeggedApiClient
```

```

        .Configure()
        .WithClientId(clientId)
        .AndClientSecret(clientSecret)
        .ForAccount(accountId)
        .Create();

/*
 * Building ApiClient with all options set
 */
ApiClient client = TwoLeggedApiClient
    .Configure()
    .WithClientId(clientId)
    .AndClientSecret(clientSecret)
    .ForAccount(accountId)
    .WithOptions(options =>
    {
        options.HubId = $"b.{accountId}";
        options.HttpClient = new HttpClient();
        options.DryRun = false;
        options.Logger = new NLogLogger(new NLogLoggerConfiguration()
        {
            LogLevel = LogLevel.Trace,
            LogToConsole = true,
            LogToFile = true
        });
        options.RetryAttempts = 1;
        options.InitialRetryInSeconds = 2;
    })
    .Create();

/*
 * Getting all projects with above ApiClient
 */
List<Project> projects = await client.GetProjects();
foreach (Project p in projects)
{
    Console.WriteLine(p.Name);
}

/*
 * Get project by ID
 */
Project project = await client.GetProject("projectIdGoesHere");

/*
 * Load root folder contents
 */
await project.GetRootFolder();

foreach (File f in project.RootFolder.Files)
{
    Console.WriteLine(f.Name);
}

foreach (Folder f in project.RootFolder.Subfolders)
{
    Console.WriteLine(f.Name);
}

/*
 * GetContents or GetContentsRecursively can be called on any folder
 */
await project.RootFolder.Subfolders[0].GetContents();
await project.RootFolder.Subfolders[1].GetContentsRecursively();

/*

```

```

    * GetContentsRecursively can be called directly on a project to enumerate a projects entire
    */
await project.GetContentsRecursively();

/*
 * Once the contents are loaded into memory they can be downloaded
 */
await project.DownloadContentsRecursively(@"C:\ExampleBackupDirectory");

/*
 * Folders can be downloaded by themselves...
 */
await project.RootFolder.Subfolders[0].DownloadContents(@"C:\ExampleBackupDirectory");

/*
 * ...or recursively. Remember to enumerate contents before recursively downloading.
 */
await project.RootFolder.Subfolders[0].GetContentsRecursively();
await project.RootFolder.Subfolders[0].DownloadContentsRecursively(@"C:\ExampleBackupDirectory");

/*
 * Example of how to get all projects then back them all up
 */
List<Project> allProjects = await client.GetProjects();
var downloadRoot = @"C:\ExampleBackupDirectory";
foreach (Project p in allProjects)
{
    await p.GetContentsRecursively();
    await p.DownloadContentsRecursively(Path.Combine(downloadRoot, p.Name));
}

```