Anthony Giretti's .NET blog (http://anthonygiretti.com/)

Ain't no mountain high enough

Home (http://anthonygiretti.com/) / 2015 (http://anthonygiretti.com/2015/) / October (http://anthonygiretti.com/2015/10/) / 25 (http://anthonygiretti.com/2015/10/25/) / Google Drive API with .NET, an alternative to Azure Blob Storage

.NET | GOOGLE API

Google Drive API with .NET, an alternative to Azure Blob Storage

by anthonygiretti (http://anthonygiretti.com/author/anthonygiretti/) | 2015-10-25

Google Drive REST API Overview

The Drive platform gives you a group of APIs along with client libraries, language-specific examples, and documentation to help you develop apps that integrate with Drive.

The core functionality of Drive apps is to download and upload files in Google Drive. However, the Drive platform provides a lot more than just storage, such as Create and open files, Search for files……

It's a very interesting alternative to Azure Blob Storage, because Google offers you 15 GB free storage, but there is one limitation : 500.000 requests maximum per day.

.NET Quickstart

Complete the steps described in the rest of this page, and in about five minutes you'll have a simple .NET console application that makes requests to the Drive API.

<u>Prerequisites</u>

To run this quickstart, you'll need:

- Visual Studio 2013 or later.
- Access to the internet and a web browser.
- A Google account with Google Drive enabled.

<u>Step 1: Enable the Drive API </u>

- Use this wizard (https://console.developers.google.com/flows/enableapi?apiid=drive&pli=1)

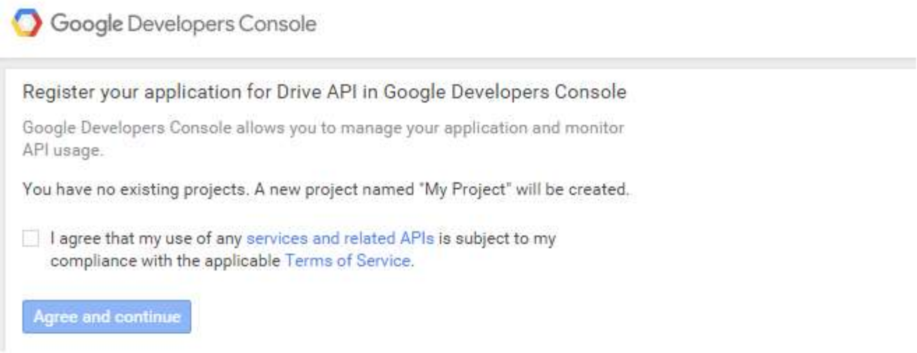
FOLLOW ME!

(https://www.linkedin.com/in/anthony-g-98670426/) (https://twitter.com/anthonygiretti)

RECENT POSTS

Building a Windows service with Worker Services and .NET Core 3.1, part 2: Migrate a timed service built with TopShelf (http://anthonygiretti.com/2020/01/03/buil a-windows-service-with-service-workers-and-net-core-3-1-part-2-migrate-a-timed-service-built-with-topshelf/)

Building a Windows service with Worker Services and .NET Core 3.1, part 1: Introduction (http://anthonygiretti.com/2020/01/02/buil a-windows-service-with-worker-to create or select a project in the Google Developers Console and automatically enable the API. Click the **Go to credentials** button to continue.

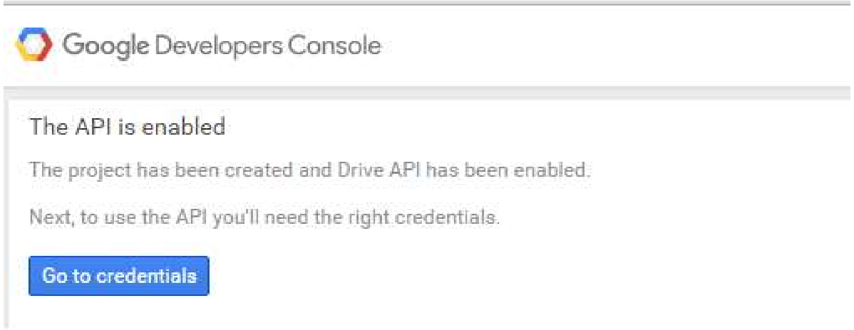


Register your application for Drive API in Google Developers Console

Google Developers Console allows you to manage your application and monitor API usage.

You have no existing projects. A new project named "My Project" will be created.

□ I agree that my use of any services and related APIs is subject to my compliance with the applicable Terms of Service.

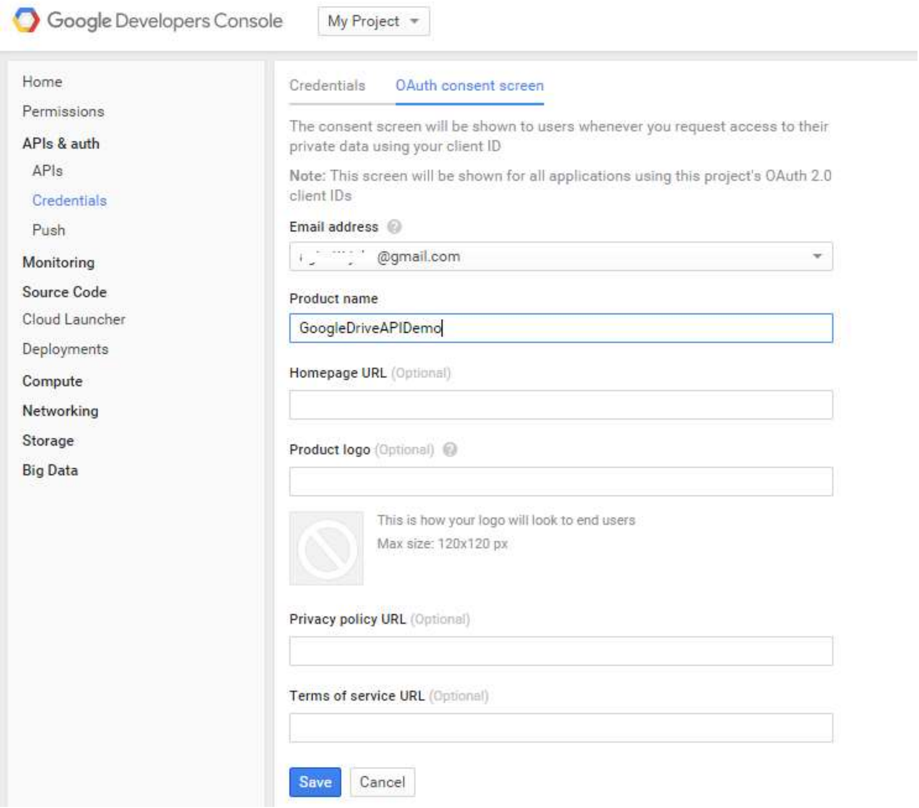


The API is enabled

The project has been created and Drive API has been enabled.

Next, to use the API you'll need the right credentials.

- At the top of the page, select the **OAuth consent screen** tab. Select an **Email address**, enter a **Product name** if not already set, and click the Save button.



The consent screen will be shown to users whenever you request access to their private data using your client ID

Note: This screen will be shown for all applications using this project's OAuth 2.0 client IDs

Email address: [redacted]@gmail.com

Product name: GoogleDriveAPIDemo

Homepage URL (Optional)

Product logo (Optional)

This is how your logo will look to end users
Max size: 120x120 px

Privacy policy URL (Optional)

Terms of service URL (Optional)

- Back on the Credentials tab, click the Add credentials button and select OAuth 2.0 client ID.

services-and-net-core-3-1-part-1-introduction/)

My migration experience from ASP.NET Core 2.2 to 3.1, part 4: Changes in the rest of the application (http://anthonygiretti.com/2019/12/31/my-migration-experience-from-asp-net-core-2-2-to-3-1-part-4-changes-in-the-rest-of-the-application/)

My migration experience from ASP.NET Core 2.2 to 3.1, part 3: Changes in Startup.cs (http://anthonygiretti.com/2019/12/29/my-migration-experience-from-asp-net-core-2-2-to-3-1-part-3-changes-in-startup-cs/)

My migration experience from ASP.NET Core 2.2 to 3.1, part 2: Changes in Program.cs (http://anthonygiretti.com/2019/12/29/my-migration-experience-from-asp-net-core-2-2-to-3-1-part-2-changes-in-program-cs/)

ARCHIVES

January 2020 (http://anthonygiretti.com/2020/01/)

December 2019 (http://anthonygiretti.com/2019/12/)

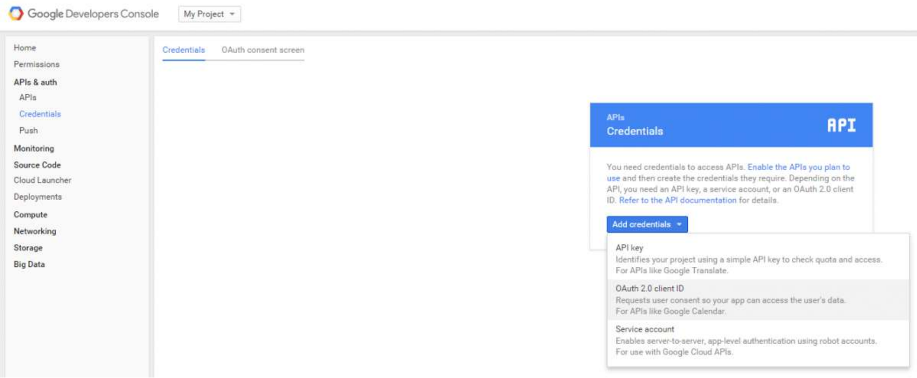
November 2019 (http://anthonygiretti.com/2019/11/)

October 2019 (http://anthonygiretti.com/2019/10/)

September 2019 (http://anthonygiretti.com/2019/09/)

August 2019 (http://anthonygiretti.com/2019/08/)

July 2019 (http://anthonygiretti.com/2019/07/)



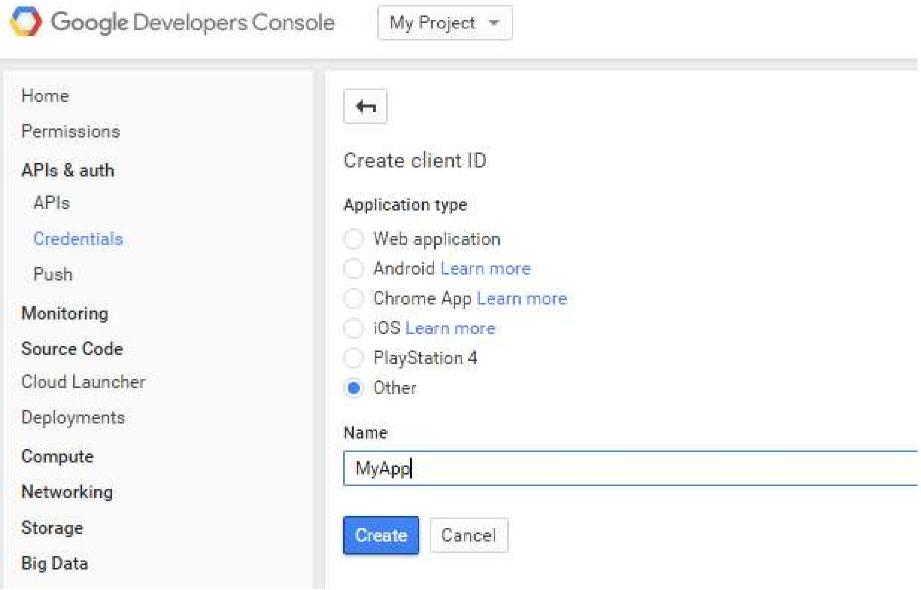
You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.

API key
Identifies your project using a simple API key to check quota and access.
For APIs like Google Translate.

OAuth 2.0 client ID
Requests user consent so your app can access the user's data.
For APIs like Google Calendar.

Service account
Enables server-to-server, app-level authentication using robot accounts.
For use with Google Cloud APIs.

- Select the application type Other and click the Create button.



Create client ID

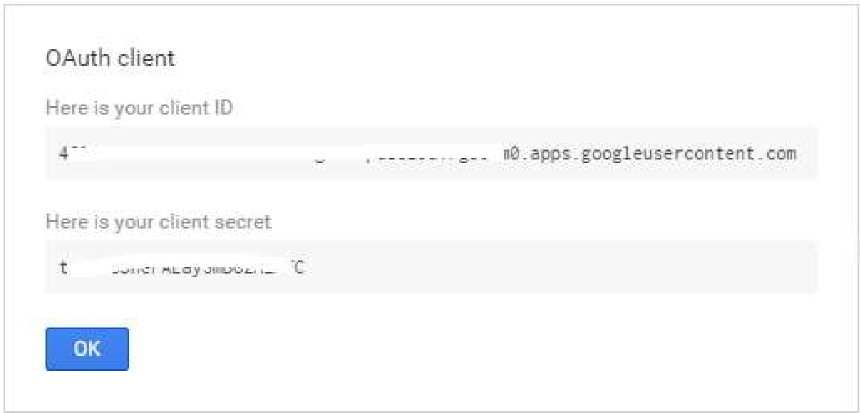
Application type

○ Web application
○ Android Learn more
○ Chrome App Learn more
○ iOS Learn more
○ PlayStation 4
● Other

Name

MyApp

- Click OK to dismiss the resulting dialog.



OAuth client

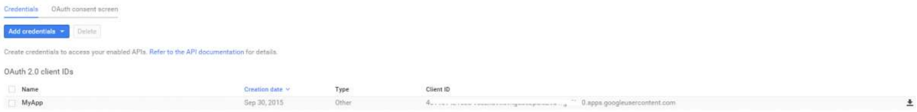
Here is your client ID

4 [redacted] m0.apps.googleusercontent.com

Here is your client secret

t [redacted] C

- Click the (Download JSON) button to the right of the client ID. Move this file to your working directory and rename it client_secret.json.



Create credentials to access your enabled APIs. Refer to the API documentation for details.

OAuth 2.0 client IDs

Name	Creation date	Type	Client ID
MyApp	Sep 30, 2015	Other	4 [redacted] 0.apps.googleusercontent.com

Step 2: Prepare the project

June 2019
(http://anthonygiretti.com/2019/06/)

May 2019
(http://anthonygiretti.com/2019/05/)

April 2019
(http://anthonygiretti.com/2019/04/)

March 2019
(http://anthonygiretti.com/2019/03/)

January 2019
(http://anthonygiretti.com/2019/01/)

December 2018
(http://anthonygiretti.com/2018/12/)

November 2018
(http://anthonygiretti.com/2018/11/)

October 2018
(http://anthonygiretti.com/2018/10/)

September 2018
(http://anthonygiretti.com/2018/09/)

August 2018
(http://anthonygiretti.com/2018/08/)

June 2018
(http://anthonygiretti.com/2018/06/)

May 2018
(http://anthonygiretti.com/2018/05/)

April 2018
(http://anthonygiretti.com/2018/04/)

March 2018
(http://anthonygiretti.com/2018/03/)

February 2018
(http://anthonygiretti.com/2018/02/)

January 2018
(http://anthonygiretti.com/2018/01/)

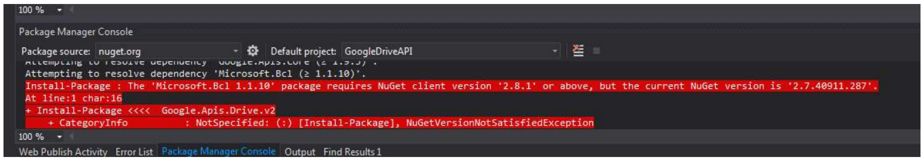
December 2017
(http://anthonygiretti.com/2017/12/)

October 2017
(http://anthonygiretti.com/2017/10/)

- Create a new Visual C# Console Application project in Visual Studio.
- Open the NuGet Package Manager Console, select the package source nuget.org, and run the following command:

```
PM> Install-Package Google.Apis.Drive.v2
```

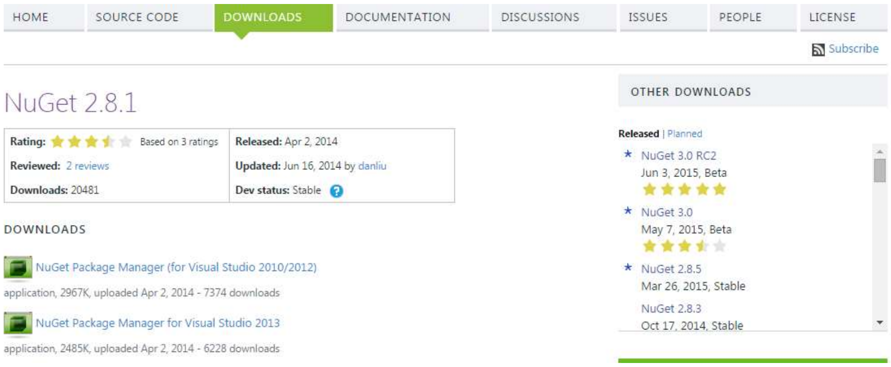
You may have this issue :



Package source: nuget.org
Default project: GoogleDriveAPI

```
Attempting to resolve dependency 'Microsoft.Bcl (≥ 1.1.10)'.
Install-Package : The 'Microsoft.Bcl 1.1.10' package requires NuGet client version '2.8.1' or above, but the current NuGet version is '2.7.40911.287'.
At line:1 char:16
+ Install-Package <<<<  Google.Apis.Drive.v2
    + CategoryInfo          : NotSpecified: (:) [Install-Package], NuGetVersionNotSatisfiedException
```

Your NuGet client version is not up to date, you have to donwload and install the lastest release here : https://nuget.codeplex.com/releases/view/118318 (https://nuget.codeplex.com/releases/view/118318)



NuGet 2.8.1

Rating: ★★★½☆ Based on 3 ratings
Reviewed: 2 reviews
Downloads: 20481
Released: Apr 2, 2014
Updated: Jun 16, 2014 by danliu
Dev status: Stable

DOWNLOADS

NuGet Package Manager (for Visual Studio 2010/2012)
application, 2967K, uploaded Apr 2, 2014 - 7374 downloads

NuGet Package Manager for Visual Studio 2013
application, 2485K, uploaded Apr 2, 2014 - 6228 downloads

OTHER DOWNLOADS

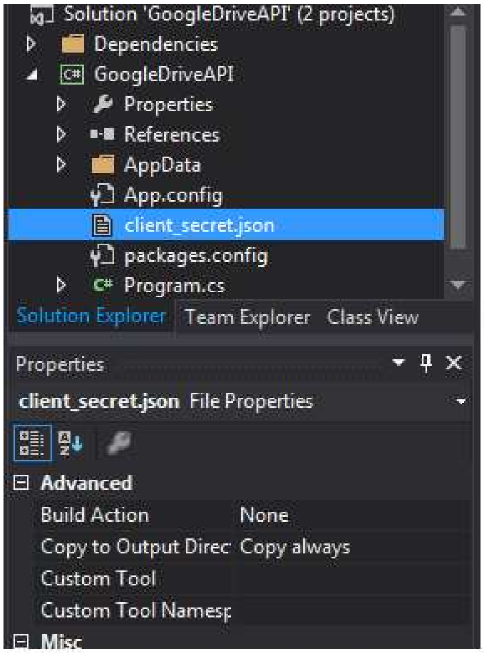
Released | Planned

- NuGet 3.0 RC2
  Jun 3, 2015, Beta ★★★★★
- NuGet 3.0
  May 7, 2015, Beta ★★★½☆
- NuGet 2.8.5
  Mar 26, 2015, Stable
- NuGet 2.8.3
  Oct 17, 2014, Stable

<u>Step 3: Set up the sample</u>

Drag *client_secret.json* (downloaded in Step 1) into your Visual Studio Solution Explorer.

Select *client_secret.json*, and then go to the Properties window and set the **Copy to Output Directory** field to **Copy always**.



Build Action None
Copy to Output Direc Copy always
Custom Tool
Custom Tool Namesp

July 2017 (http://anthonygiretti.com/2017/07/)

June 2017 (http://anthonygiretti.com/2017/06/)

January 2016 (http://anthonygiretti.com/2016/01/)

December 2015 (http://anthonygiretti.com/2015/12/)

November 2015 (http://anthonygiretti.com/2015/11/)

October 2015 (http://anthonygiretti.com/2015/10/)

September 2015 (http://anthonygiretti.com/2015/09/)

May 2015 (http://anthonygiretti.com/2015/05/)

September 2014 (http://anthonygiretti.com/2014/09/)

August 2014 (http://anthonygiretti.com/2014/08/)

May 2014 (http://anthonygiretti.com/2014/05/)

CATEGORIES

.NET (http://anthonygiretti.com/category/net/)

.NET Core (http://anthonygiretti.com/category/net-core/)

Adal-Angular5 (http://anthonygiretti.com/category/adal-angular5/)

ADAL.JS (http://anthonygiretti.com/category/adal-js/)

Replace the contents of Program.cs with the following code:

Angular
(http://anthonygiretti.com/category/angula

Angular 4
(http://anthonygiretti.com/category/angula
4/)

Angular 5
(http://anthonygiretti.com/category/angula
5/)

Angular 6
(http://anthonygiretti.com/category/angula
6/)

ASP.NET CORE
(http://anthonygiretti.com/category/asp-net-core/)

ASP.NET Core 2.1
(http://anthonygiretti.com/category/asp-net-core-2-1/)

ASP.NET Core 2.2
(http://anthonygiretti.com/category/asp-net-core-2-2/)

ASP.NET MVC
(http://anthonygiretti.com/category/asp-net-mvc/)

AutoMapper
(http://anthonygiretti.com/category/autom

Azure
(http://anthonygiretti.com/category/azure/

Azure AD
(http://anthonygiretti.com/category/azure-ad/)

C#
(http://anthonygiretti.com/category/c/)

CI
(http://anthonygiretti.com/category/ci/)

CodeAnalysis
(http://anthonygiretti.com/category/codear

Dapper
(http://anthonygiretti.com/category/dappe

```
using System;
using Google.Apis.Auth.OAuth2;
using Google.Apis.Drive.v2;
using Google.Apis.Services;
using Google.Apis.Util.Store;
using System.Collections.Generic;
using System.Diagnostics;
using System.IO;
using System.Threading;
using File = Google.Apis.Drive.v2.Data.File;

namespace GoogleDriveAPI
{
    class Program
    {
        static string[] Scopes = { DriveService.Scope.Drive };
        static string ApplicationName = "GoogleDriveAPIDemoApp";

        static void Main(string[] args)
        {
            UserCredential credential;

            using (var stream =
                new FileStream("client_secret.json", FileMode.Open, FileAcces
            {
                string credPath = System.Environment.GetFolderPath(
                    System.Environment.SpecialFolder.Personal);
                credPath = Path.Combine(credPath, ".credentials/drive-dotnet-

                credential = GoogleWebAuthorizationBroker.AuthorizeAsync(
                    GoogleClientSecrets.Load(stream).Secrets,
                    Scopes,
                    "user",
                    CancellationToken.None,
                    new FileDataStore(credPath, true)).Result;
                Console.WriteLine("Credential file saved to: " + credPath);
            }

            // Create Drive API service.
            var service = new DriveService(new BaseClientService.Initializer(
            {
                HttpClientInitializer = credential,
                ApplicationName = ApplicationName,
            });

            /////////////////////////////// UPLOAD FILE /////////////////////
            UploadFile(service);

            ////////////////////////////// LIST FILES ///////////////////////
            ListFiles(service);


        }

        private static void UploadFile(DriveService service)
        {
            File body = new File();
            body.Title = "test upload";
            body.Description = "test upload";
            body.MimeType = "application/vnd.ms-excel";
```

DataTable
(http://anthonygiretti.com/category/datata

Documentation
(http://anthonygiretti.com/category/docum

Duck Casting
(http://anthonygiretti.com/category/duck-casting/)

Entity Framework
(http://anthonygiretti.com/category/entity-framework/)

Entity Framework Core
(http://anthonygiretti.com/category/entity-framework-core/)

Entity Framework Core 2
(http://anthonygiretti.com/category/entity-framework-core-2/)

FluentAssertions
(http://anthonygiretti.com/category/fluenta

FluentValidation
(http://anthonygiretti.com/category/fluentv

Function App
(http://anthonygiretti.com/category/functic app/)

Google API
(http://anthonygiretti.com/category/google api/)

Google Charts
(http://anthonygiretti.com/category/google charts/)

HTML 5
(http://anthonygiretti.com/category/html-5/)

InjectionDependency
(http://anthonygiretti.com/category/injectic

Integration tests
(http://anthonygiretti.com/category/integra tests/)

Javascript
(http://anthonygiretti.com/category/javascr

```
            // File's content.
            byte[] byteArray = System.IO.File.ReadAllBytes("/Temp/testUploadE
            MemoryStream stream = new MemoryStream(byteArray);
            try {
                FilesResource.InsertMediaUpload request = service.Files.Inser
                request.Upload();

              File file = request.ResponseBody;

              // Uncomment the following line to print the File ID.
              // Console.WriteLine("File ID: " + file.Id);

            } catch (Exception e) {
              Console.WriteLine("An error occurred: " + e.Message);
            }
        }


        private static void ListFiles(DriveService service)
        {
            // Define parameters of request.
            FilesResource.ListRequest listRequest = service.Files.List();
            listRequest.MaxResults = 100;

            // List files.
            IList files = listRequest.Execute()
                .Items;
            Console.WriteLine("Files:");
            if (files != null && files.Count > 0)
            {
                foreach (var file in files)
                {
                    Console.WriteLine("{0} ({1})", file.Title, file.Id);
                    Debug.WriteLine("{0} ({1})", file.Title, file.Id);
                }
            }
            else
            {
                Console.WriteLine("No files found.");
            }
            Console.Read();
        }
    }
}
```

Step 4: Upload a file :

When a file is uploaded, Google Drive API provides you in the upload result,the file Id, and its donwload url :

JSON (http://anthonygiretti.com/category/json/)

JWT (http://anthonygiretti.com/category/jwt/)

Karma (http://anthonygiretti.com/category/karma

Logging (http://anthonygiretti.com/category/logging

Massive (http://anthonygiretti.com/category/massiv

Micro ORM (http://anthonygiretti.com/category/micro-orm/)

middleware (http://anthonygiretti.com/category/middle

MiniProfiler (http://anthonygiretti.com/category/minipr

MSTest V2 (http://anthonygiretti.com/category/mstest v2/)

NDepend (http://anthonygiretti.com/category/ndepe

Newtonsoft (http://anthonygiretti.com/category/newto

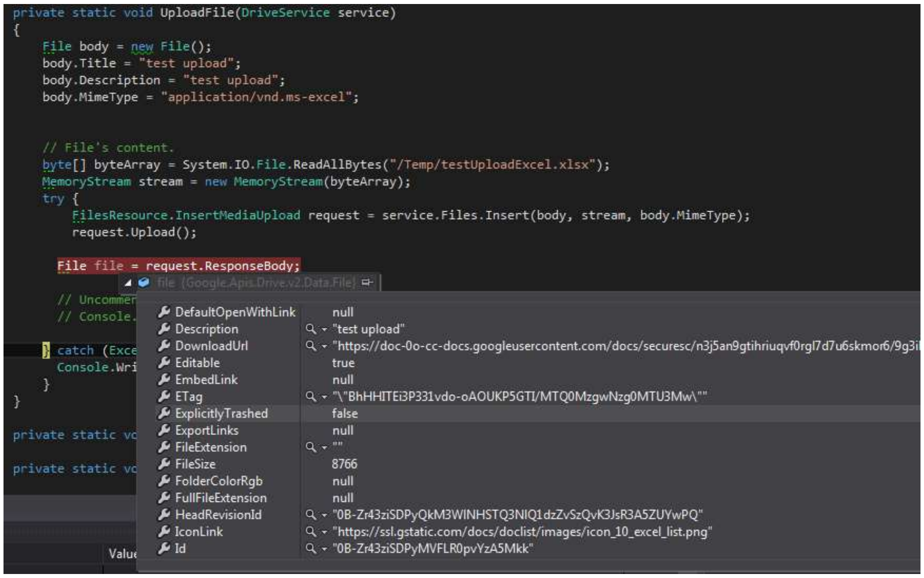
Non classé (http://anthonygiretti.com/category/non-classe/)

NPoco (http://anthonygiretti.com/category/npoco/

Open Id Connect (http://anthonygiretti.com/category/open-id-connect/)

OrmLite (http://anthonygiretti.com/category/ormlite

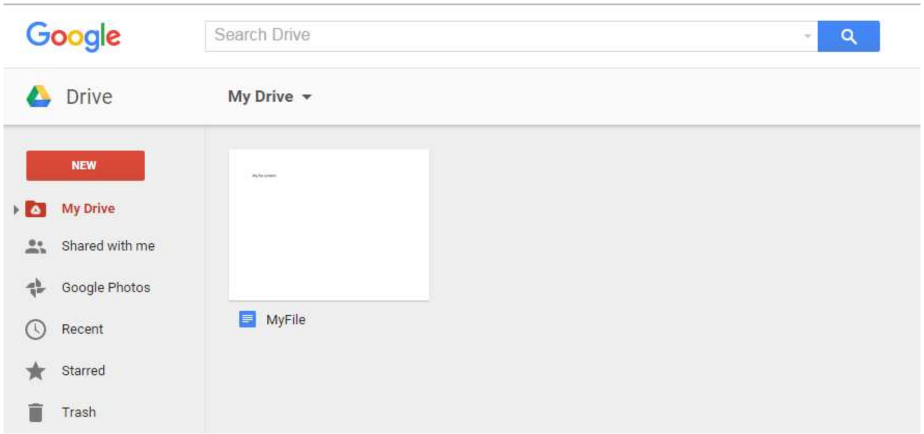
Owned types (http://anthonygiretti.com/category/owned types/)



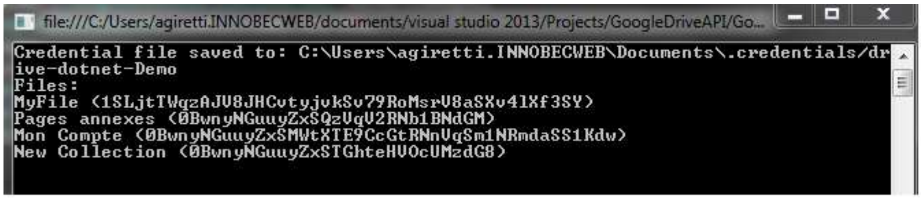
```
private static void UploadFile(DriveService service)
{
    File body = new File();
    body.Title = "test upload";
    body.Description = "test upload";
    body.MimeType = "application/vnd.ms-excel";


    // File's content.
    byte[] byteArray = System.IO.File.ReadAllBytes("/Temp/testUploadExcel.xlsx");
    MemoryStream stream = new MemoryStream(byteArray);
    try {
        FilesResource.InsertMediaUpload request = service.Files.Insert(body, stream, body.MimeType);
        request.Upload();

      File file = request.ResponseBody;

      // Uncommen
      // Console.

    } catch (Exce
      Console.Wri
    }
}

private static vo

private static vo
```

```
file {Google.Apis.Drive.v2.Data.File}
DefaultOpenWithLink   null
Description           "test upload"
DownloadUrl           "https://doc-0o-cc-docs.googleusercontent.com/docs/securesc/n3j5an9gtihriuqvf0rgl7d7u6skmor6/9g3i
Editable              true
EmbedLink             null
ETag                  "\"BhHHITEi3P331vdo-oAOUKP5GTI/MTQ0MzgwNzg0MTU3Mw\""
ExplicitlyTrashed     false
ExportLinks           null
FileExtension         ""
FileSize              8766
FolderColorRgb        null
FullFileExtension     null
HeadRevisionId        "0B-Zr43ziSDPyQkM3WINHSTQ3NIQ1dzZvSzQvK3JsR3A5ZUYwPQ"
IconLink              "https://ssl.gstatic.com/docs/doclist/images/icon_10_excel_list.png"
Id                    "0B-Zr43ziSDPyMVFLR0pvYzA5Mkk"
```

The uploaded file is also available on the Google Drive UI :



Step 5: List files:


file:///C:/Users/agiretti.INNOBECWEB/documents/visual studio 2013/Projects/GoogleDriveAPI/Go...

```
Credential file saved to: C:\Users\agiretti.INNOBECWEB\Documents\.credentials/dr
ive-dotnet-Demo
Files:
MyFile (1SLjtTWqzAJV8JHCvtyjvkSv79RoMsrV8aSXv41Xf3SY)
Pages annexes (0BwnyNGuuyZxSQzVqV2RNb1BNdGM)
Mon Compte (0BwnyNGuuyZxSMWtXTE9CcGtRNnVqSm1NRmdaSS1Kdw)
New Collection (0BwnyNGuuyZxSTGhteHVOcUMzdG8)
```

That’s it! in a next article i will describe how to create Sheets (excel files), and manipulate it through Google Drive API and Google Sheet API 🙂

.NET (HTTP://ANTHONYGIRETTI.COM/TAG/NET/)

GOOGLE API (HTTP://ANTHONYGIRETTI.COM/TAG/GOOGLE-API/)

« How to Load an HTML View Other Than With an MVC View File? (http://anthonygiretti.com/2015/09/20/how-to-load-an-html-view-other-than-with-an-mvc-view-file/)

How to Create Your Own Membership Provider Instead of Using SqlMembershipProvider »

Peta POCO (http://anthonygiretti.com/category/peta-poco/)

PhantomJS (http://anthonygiretti.com/category/phantc

PowerShell (http://anthonygiretti.com/category/power

Profiling (http://anthonygiretti.com/category/profilir

Scalar Function (http://anthonygiretti.com/category/scalar-function/)

Serilog (http://anthonygiretti.com/category/serilog

SignalR Core (http://anthonygiretti.com/category/signalr core/)

Simple Injector (http://anthonygiretti.com/category/simple injector/)

Simple.Data (http://anthonygiretti.com/category/simple data/)

SQL (http://anthonygiretti.com/category/sql/)

SQL SERVER (http://anthonygiretti.com/category/sql-server/)

SqlTable dependency (http://anthonygiretti.com/category/sqltabl dependency/)

SSL (http://anthonygiretti.com/category/ssl/)

StyleCop (http://anthonygiretti.com/category/stylecc

SVG (http://anthonygiretti.com/category/svg/)

Leave a Reply

You must be logged in (http://anthonygiretti.com/wp-login.php?redirect_to=http%3A%2F%2Fanthonygiretti.com%2F2015%2F10%2F25%2Fgoogle-drive-api-with-net-an-alternative-to-azure-blob-storage%2F) to post a comment.

Table splitting (http://anthonygiretti.com/category/table-splitting/)

Test Server (http://anthonygiretti.com/category/test-server/)

TLS (http://anthonygiretti.com/category/tls/)

TsLint (http://anthonygiretti.com/category/tslint/)

Typescript (http://anthonygiretti.com/category/javascr

Typescript (http://anthonygiretti.com/category/typesc 2/)

Underscore.js (http://anthonygiretti.com/category/unders js/)

UNIT TESTS (http://anthonygiretti.com/category/unit-tests/)

Visual Studio Code (http://anthonygiretti.com/category/visual-studio-code/)

Visual Studio Team Services (http://anthonygiretti.com/category/visual-studio-team-services/)

VSTS (http://anthonygiretti.com/category/vsts/)

Web API (http://anthonygiretti.com/category/web-api/)

Windows services (http://anthonygiretti.com/category/windov services/)

XUnit (http://anthonygiretti.com/category/xunit/)