IPG_LOGO_DEF

**TEMPLATE OFFERTE**

**PNERSHIP EL**

Local Nuget Packages

Table of Contents

[1 Create a nuget package 3](#_Toc61850324)

[**1.1** Prerequisite: 3](#_Toc61850325)

[**1.2** .NET Standard Project 3](#_Toc61850326)

[**1.3** .NET Framework Project 3](#_Toc61850327)

[2 Using the created package 5](#_Toc61850328)

[3 Local Nuget server 7](#_Toc61850329)

# Create a nuget package

## Prerequisite:

Nuget CLI (see <https://docs.microsoft.com/en-us/nuget/reference/nuget-exe-cli-reference> )

## .NET Standard Project

In order to create a nuget package, you have to start by creating a Class Library (Standard) project in Visual studio

* 0pen the project’s properties (select project -> Alt+Enter)
* Open ‘Package’ and fill out the necessary info
* Naming conventions and mandatory fields (proposal)
  + Id: eu.ipggroup.client.feature
  + Version: x.y.z (see https://semver.org/ )
  + Authors
  + Company
  + Description
  + Repository URL: https://p01vtfs01.ipglobal.corp/Development/.....
  + Repository type: GIT
  + …
* Create package by right-clicking the project and choosing ‘Pack’
* The resulting file can be found in the build folder (Debug/Release) with the name ‘eu.ipggroup. client.feature.x.y.z.nupkg’
* Add the package to the local source
  + Open cmd or bash and navigate to the folder where the package lives
  + Type ‘nuget add <eu.ipggroup.client.feature.x.y.z.nupkg > -source “<network\_share OR URL-to\_your\_local\_source>”’ (double quotes around the network share AND DOUBLE BACKSLASHES )

## .NET Framework Project

In order to create a nuget package, you have to start by creating a Class Library (Framework) project in Visual studio

* 0pen the project’s properties (select project -> Alt+Enter)
* Open ‘Application’ tab, choose an assembly name (should already be filled in) and click ’Assembly Information’
* Fill out the necessary info and click ‘OK’
  + Naming conventions and mandatory fields (proposal)
    - Title:
    - Id: eu.ipggroup.client.feature
    - Description
    - Company
    - Product: same as title ?
    - Copyright
    - Trademark
    - Assembly Version: x.y.z (see https://semver.org/ )
    - File Version (is used in the windows explorer)
* Set solution to ‘Release’
* Rebuild the solution to update the DLL’s.
* Open a command prompt and navigate to your solution.
* Run the command ‘nuget spec’ (Do this only once!)
* Still in the command prompt navigate to the project you want to share.
* Run the command ‘nuget spec <-projectname->.csproj’
* Edit the created <-projectname->.nuspec (depending on some unknown configs, the created file is all ready to go or you need to update it manually) It should look something like this:
  + <?xml version="1.0" encoding="utf-8"?>

<package >

<metadata>

<id>eu.ipggroup. <-projectname-> </id>

<version>1.0.0</version>

<title> <-projectname-> </title>

<authors>sven.lavigne@ipggroup.eu</authors>

<requireLicenseAcceptance>false</requireLicenseAcceptance>

<license type="expression">MIT</license>

<projectUrl> Link to the project in DevOps </projectUrl>

<description> Description </description>

<releaseNotes> Why this release (e.g. Initial release) </releaseNotes>

<copyright>Copyright © 2021</copyright>

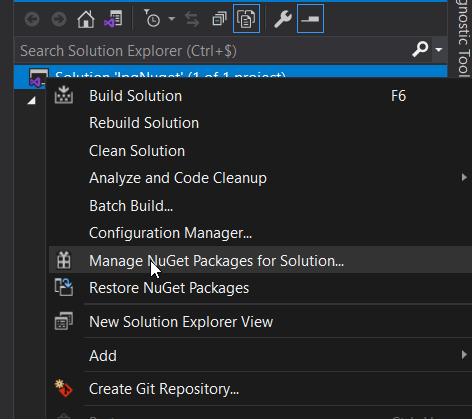
<tags>Tag1 Tag2</tags>

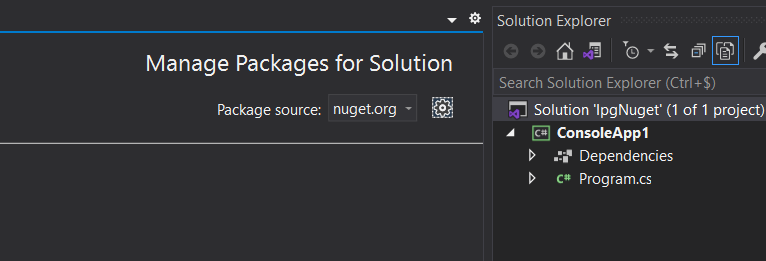
</metadata>

</package>

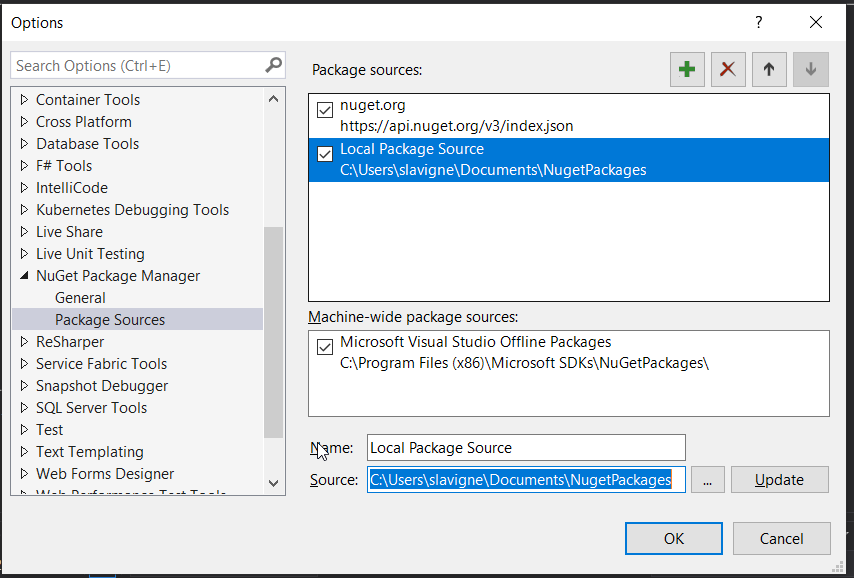
* Run command ‘nuget pack’ and a file will be created with the name ‘eu.ipggroup. <-projectname->.1.0.0.nupkg’
* Add the package to the local source
  + Open cmd or bash and navigate to the folder where the package lives
  + Type ‘nuget add <eu.ipggroup.client.feature.x.y.z.nupkg > -source "//p01VTFS01/Nuget"’ (double quotes around the network share and forward slashes!)

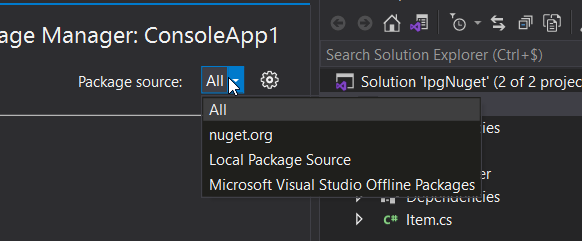
# Using the created package

* Add Source to Visual Studio
  + Open the package manager
  + Go to settings



* + Enter the source path and give it a suitable name. The source can also be a URL to a different Nuget server. For example: your (company’s) local Nuget server.



* + Change the source for the Nuget Package manager to ‘All’ or the newly created one. Disclaimer: when one of the sources is not available, the Package Manager does not work, nor will the command ‘Restore Nuget Packages’! This is the case even if you only choose the source ‘Nuget.org’. In the settings, disable the unavailable package
  + Use Nuget as you would for any other package

# Local Nuget server

The easiest way to install a local Nuget server is to use a docker image. Steps to perform

* Create a folder on the machine where you want to host the docker container, named something like ‘Nuget’
* Add a folder named ‘baget-data;C’. I know, strange name, but it seems there is a small bug in the Baget image. It should be ‘baget-data’
* Add a file, named baget.env to that folder. And add following content to it:
  + ## The API key can be any string

ApiKey= <NUGET-SERVER-API-KEY>

Storage\_\_Type=FileSystem

Storage\_\_Path=/var/baget/packages

Database\_\_Type=Sqlite

Database\_\_ConnectionString=Data Source=/var/baget/baget.db

Search\_\_Type=Database

* In a command line, cd into the Nuget folder and run the following:
  + docker pull loicsharma/baget
  + docker run --name nuget-server -p 5555:80 --env-file baget.env -v "$(pwd)/baget-data;C:/var/baget" loicsharma/baget:latest
    - You can add –rm after ‘docker run’ in order to remove the container when it is stopt. The data is persisted to the baget-data;C folder  
      But I don’t like to do it, so the container stays in the list of containers in Docker Desktop
  + Browse to <http://localhost:5555/> to see a list of available packages.
* For using a package: see chapter 2
* For uploading a package, use the following command
  + dotnet nuget push -s http://localhost:5000/v3/index.json -k <NUGET-SERVER-API-KEY> <Package name>.1.0.0.nupkg
    - Of course, the <NUGET-SERVER-API-KEY> has to be the same as the key in baget.env

# 