

**ANG-450** *POC Document*

|  |  |
| --- | --- |
| POC Title: | **.NET template & nuget package simplified** |
| Created by: | Md. Sahidul Islam, Jr. Software Engineer |
| Reviewed by: | Md. Mahedee Hasan, Head of Soft. Dev Eng |
| Approved by: | Md. Mahedee Hasan, Head of Soft. Dev Eng |
| Confidentiality level: | Internal |

**DOCUMENT CHANGE HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Description of change** |
| 21-Sept-2022 | 1.0 | Md. Sahidul Islam | Initial Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**.NET Template & NuGet Package Simplified**

Contents

[Creating an item template 1](#_Toc114645500)

[Creating a project template 2](#_Toc114645501)

[Creating a template package 3](#_Toc114645502)

[Creating a dependency package 3](#_Toc114645503)

[Publish the packages into Gitlab Package Registry 4](#_Toc114645504)

[Access the packages from Gitlab Package Registry 4](#_Toc114645505)

# 

# Creating an item template

* Go to your desired folder and create the item file (.cs or anything)
* Create a sub-folder named .template.config and create a json file named template.json
* Paste the below text into template.json

{

"$schema": "http://json.schemastore.org/template",

"author": "Sahidul",

"classifications": [ "Common", "Code" ],

"identity": "ExampleTemplate.StringExtensions",

"name": "Example templates: string extensions",

"shortName": "item",

"tags": {

"language": "C#",

"type": "item"

}

}

* Now, open command prompt and run dotnet new -–install . to install the template.
* To test, run dotnet new item
* To uninstall, run dotnet new –-uninstall .

# Creating a project template

* Go to your desired folder and create a project (your desired template). You can run dotnet new console to a new console app.
* Create a sub-folder named .template.config and create a json file named template.json
* Paste the below text into template.json

{

"$schema": "http://json.schemastore.org/template",

"author": "Me",

"classifications": [ "Common", "Console" ],

"identity": "ExampleTemplate.ConsoleProject",

"name": "Example templates: console project",

"shortName": "project",

"tags": {

"language": "C#",

"type": "project"

}

}

* You can replace texts by appending:

{

"$schema": "http://json.schemastore.org/template",

"author": "Me",

"classifications": [ "Common", "Console" ],

"identity": "ExampleTemplate.AsyncProject",

"name": "Example templates: async project",

"shortName": "consoleasync",

"tags": {

"language": "C#",

"type": "project"

},

"symbols": {

"FrameWork": {

"type": "parameter",

"description": "Framework replace korte chai",

"datatype": "choice",

"choices": [

{

"choice": "net6.0"

},

{

"choice": "net5.0"

}

],

"defaultValue": "net6.0",

"replaces": "{TargetFramework}"

}

}

}

* Now, open command prompt and run dotnet new –install . to install the template.
* To test run dotnet new consoleasync
* To uninstall, run dotnet new –-uninstall .

# Creating a template package

* Create a template project by running dotnet new console –n templatepack –o .
* We only need the .csproj file and ofcourse our templates
* Next, open the templatepack.csproj file in your favorite editor and replace the content with the following XML:

<Project Sdk="Microsoft.NET.Sdk">

<PropertyGroup>

<PackageType>Template</PackageType>

<PackageVersion>1.0</PackageVersion>

<PackageId>AdatumCorporation.Utility.Templates</PackageId>

<Title>AdatumCorporation Templates</Title>

<Authors>Me</Authors>

<Description>Templates to use when creating an application for Adatum Corporation.</Description>

<PackageTags>dotnet-new;templates;contoso</PackageTags>

<TargetFramework>netstandard2.0</TargetFramework>

<IncludeContentInPack>true</IncludeContentInPack>

<IncludeBuildOutput>false</IncludeBuildOutput>

<ContentTargetFolders>content</ContentTargetFolders>

<NoWarn>$(NoWarn);NU5128</NoWarn>

<NoDefaultExcludes>true</NoDefaultExcludes>

</PropertyGroup>

<ItemGroup>

<Content Include="templates\\*\*\\*" Exclude="templates\\*\*\bin\\*\*;templates\\*\*\obj\\*\*" />

<Compile Remove="\*\*\\*" />

</ItemGroup>

</Project>

* Now run **dotnet pack** to pack the templates, you will find the nuget template package in bin folder.
* To install, **dotnet new –install <path-to-package>**

# Creating a dependency package

* Open a directory and run **dotnet new classlib** to create a library project.
* Open your project file (.csproj, .fsproj or .vbproj depending on the language you're using) and add the following minimal properties inside the existing <PropertyGroup> tag, changing the values as appropriate:

<PackageId>AppLogger</PackageId>

<Version>1.0.0</Version>

<Authors>your\_name</Authors>

<Company>your\_company</Company>

* Run **dotnet pack** to pack the library into a package, output will be in bin/debug folder.

# Publish the packages into Gitlab Package Registry

* Add the source first

dotnet nuget add source "https://gitlab.example.com/api/v4/projects/<your\_project\_id>/packages/nuget/index.json" --name <source\_name> --username <gitlab\_username or deploy\_token\_username> --password <gitlab\_personal\_access\_token or deploy\_token>

Example: **dotnet nuget add source https://gitlab.com/api/v4/projects/39130081/packages/nuget/index.json --name testSource --username sahidul2866 –-password glpat-cs2szjpDsjyTjuNAZ4Zq**

* Push the package

Dotnet nuget push <package\_file> --source <source\_name>

Example: **dotnet nuget push ASAI.Sahidul.DepPack.1.0.0.nupkg --source testSource**

# Access the packages from Gitlab Package Registry

* Simply install the package by **dotnet new --install <package-id>**
* To add dependency reference to your project:
  + **Dotnet add package <package-id>**
* **If it is a template package then use dotnet new <template-short-name>**