Créer un modèle C# pour AutoCAD avec Visual Studio 2015

L'objectif de ce tutoriel est de montrer comment créer un modèle de démarrage d'un nouveau projet C# pour AutoCAD dans Visual Studio 2015, modèle permettant de lancer automatiquement AutoCAD en chargeant la DLL depuis Visual Studio en mode Debug.

L'exemple montré utilise Visual Studio 2015 Community mais il est facilement transposable pour d'autres versions.

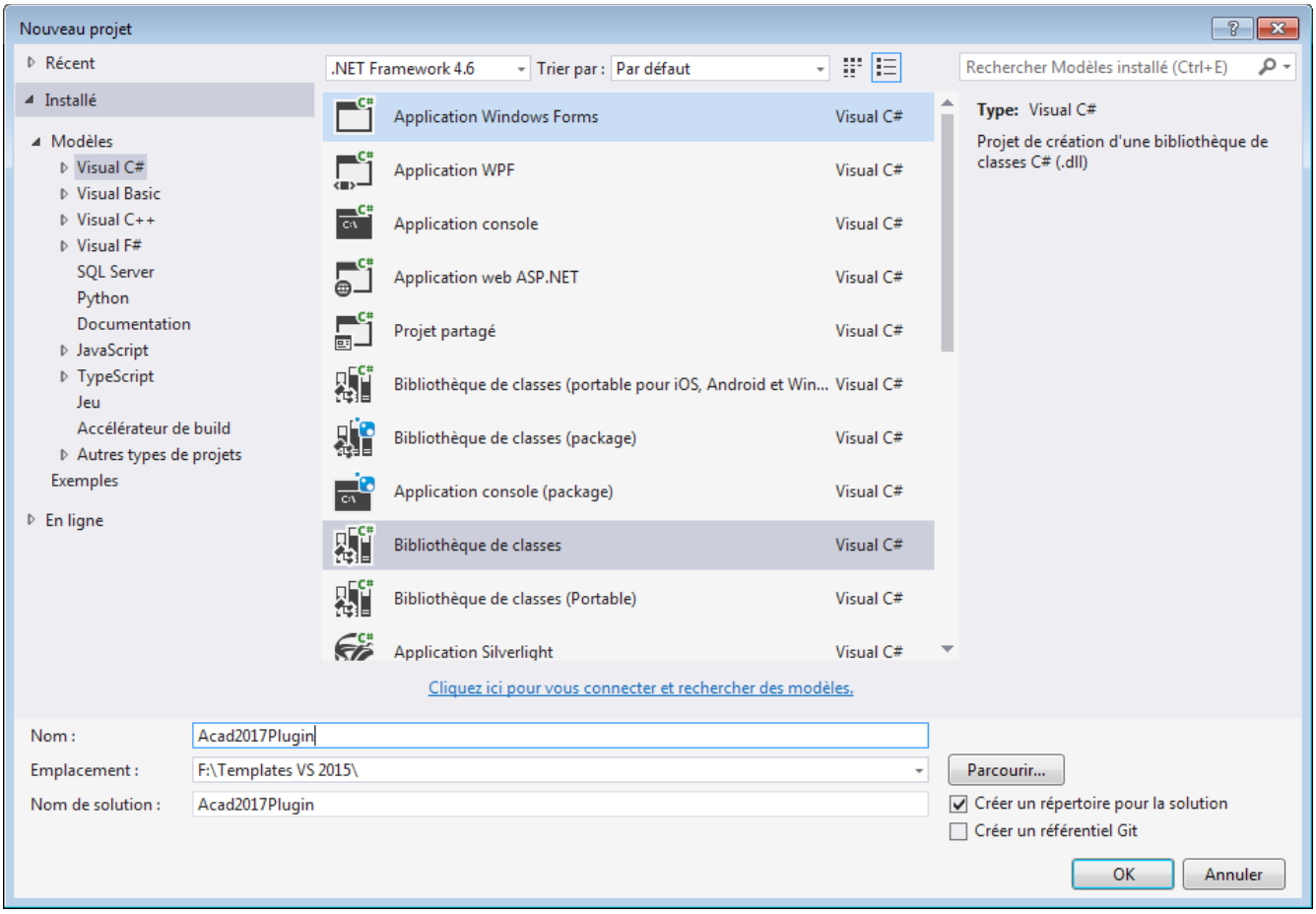
Il y sera créé un modèle pour AutoCAD 2017 mais là encore il est possible de transposer pour cibler d'autres versions d'AutoCAD.

Le chemin du répertoire de sortie pour le mode Debug utilisé dans l'exemple est celui par défaut (.\bin\Debug).

Démarrer un nouveau projet

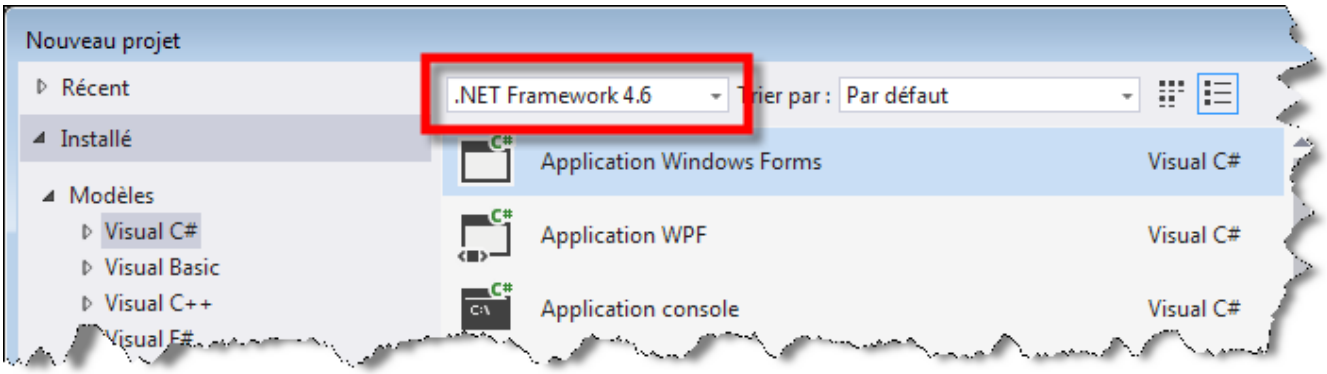
Dans Visual Studio, choisir le langage *Visual C#* et le type de projet : *Bibliothèque de classes*.

Renommer le projet : Acad2017Plugin par exemple, et changer le chemin de l'emplacement comme désiré.



Spécifier le Framework ciblé

Dans la boite de dialogue *Nouveau projet*, choisir le Framework cible dans la liste déroulante.



Il est préférable de cibler la version du Framework correspondant à celle installée par la version d'AutoCAD pour la quelle le modèle est créé :

AutoCAD Version	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
Release	R17.0	R17.1	R17.2	R18.0	R18.1	R18.2	R19.0	R19.1	R20.0	R20.1	R21.0	R22.0
DWG Format	AC1021	AC1021	AC1021	AC1024	AC1024	AC1024	AC1027	AC1027	AC1027	AC1027	AC1027	AC1032
Installed .NET Framework	2.0	2.0	2.0	3.5	3.5	4.0	4.0	4.0	4.5	4.5	4.6	4.6
AcCoreMgd.dll required												

Visual Studio 2005 (8.0)												
Visual Studio 2008 (9.0)												
Visual Studio 2010 (10.0)												
Visual Studio 2012 (11.0)												
Visual Studio 2013 (12.0)												
Visual Studio 2015 (14.0)												
Visual Studio 2017 (15.0)												

Ajouter les bibliothèques AutoCAD

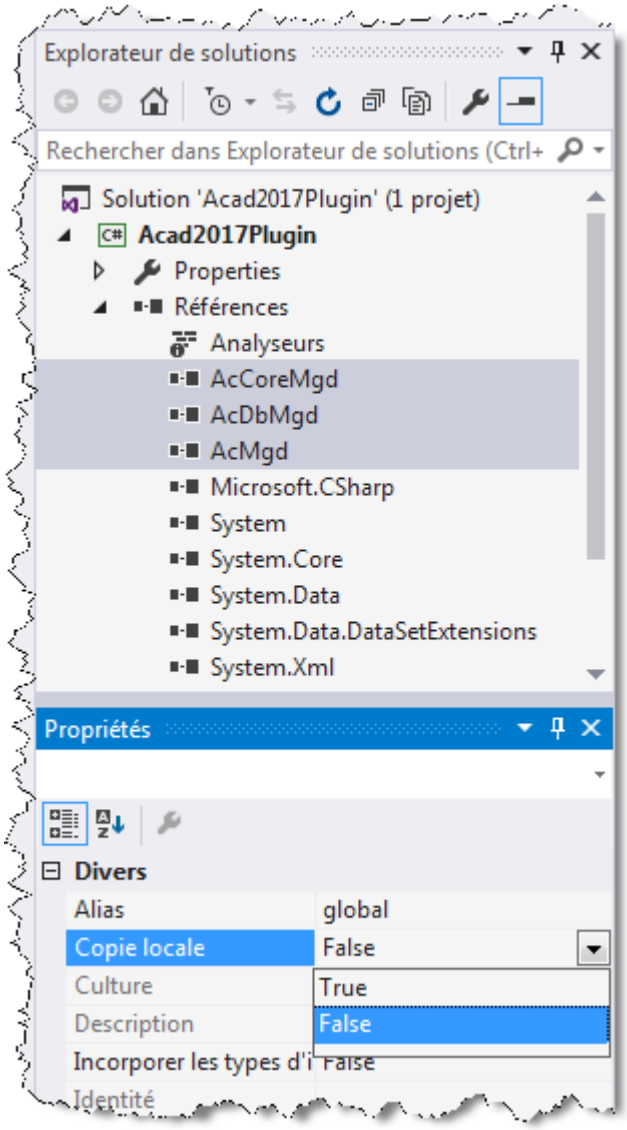
Dans l'explorateur de solutions, sélectionner *Références* puis clic droit et *Ajouter une référence...*
Dans la boite de dialogue *Gestionnaire des références*, cliquer sur *Parcourir...*

Les références à ajouter se trouvent dans le répertoire d'installation de la version ciblée d'AutoCAD ou, mieux, dans le dossier ObjectARX20##\inc correspondant à la version d'AutoCAD ciblée (inc-win32 ou inc-x64 suivant les versions d'AutoCAD et la plateforme du poste utilisé).
On peut télécharger les dernières versions d'ObjectARX sur cette page :
http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=773204

Choisir les bibliothèques les plus couramment utilisées (il sera facile d'en ajouter d'autres de la même manière si la création du nouveau projet le requiert).

- AutoCAD 2007 à 2012 : AcDbMgd.dll et AcMgd.dll
- AutoCAD 2013/2017 : AcCoreMgd.dll, AcDbMgd.dll et AcMgd.dll

Mettre la propriété *Copie locale* de ces DLLs sur *False* :



Ajouter une ébauche de code

Renommer la classe créée par Visual Studio : *Class1* avec un nom plus explicite, *Commands* par exemple, depuis l'explorateur de solution. Une boite de dialogue propose de renommer toutes les références à *Class1* dans le projet, répondre *Oui*.

Ouvrir la classe *Commands* dans l'éditeur de code par double clic sur *Commands* dans l'explorateur de solution.

Ajouter les instructions *using* pour importer les espaces de nom AutoCAD les plus couramment utilisés et, éventuellement, un alias pour la classe Autodesk.AutoCAD.ApplicationServices.Application.

Ajouter quelques lignes de codes usuelles pour définir une commande AutoCAD :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Autodesk.AutoCAD.ApplicationServices;
using Autodesk.AutoCAD.DatabaseServices;
using Autodesk.AutoCAD.EditorInput;
using Autodesk.AutoCAD.Geometry;
using Autodesk.AutoCAD.Runtime;

using AcAp = Autodesk.AutoCAD.ApplicationServices.Application;

namespace Acad2017Plugin
{
    public class Commands
    {
        [CommandMethod("TEST")]
        public void Test()
        {
            var doc = AcAp.DocumentManager.MdiActiveDocument;
            var db = doc.Database;
            var ed = doc.Editor;

            using (var tr = db.TransactionManager.StartTransaction())
            {

                tr.Commit();
            }
        }
    }
}
```

Générer la solution pour contrôler l'absence d'erreur (F6).

Ajouter un script pour charger l'application au démarrage d'AutoCAD

Depuis l'explorateur de solutions, faire un clic droit sur le projet Acad2017Plugin puis *Ajouter* et *Nouvel élément...*(Ctrl+Maj+A).

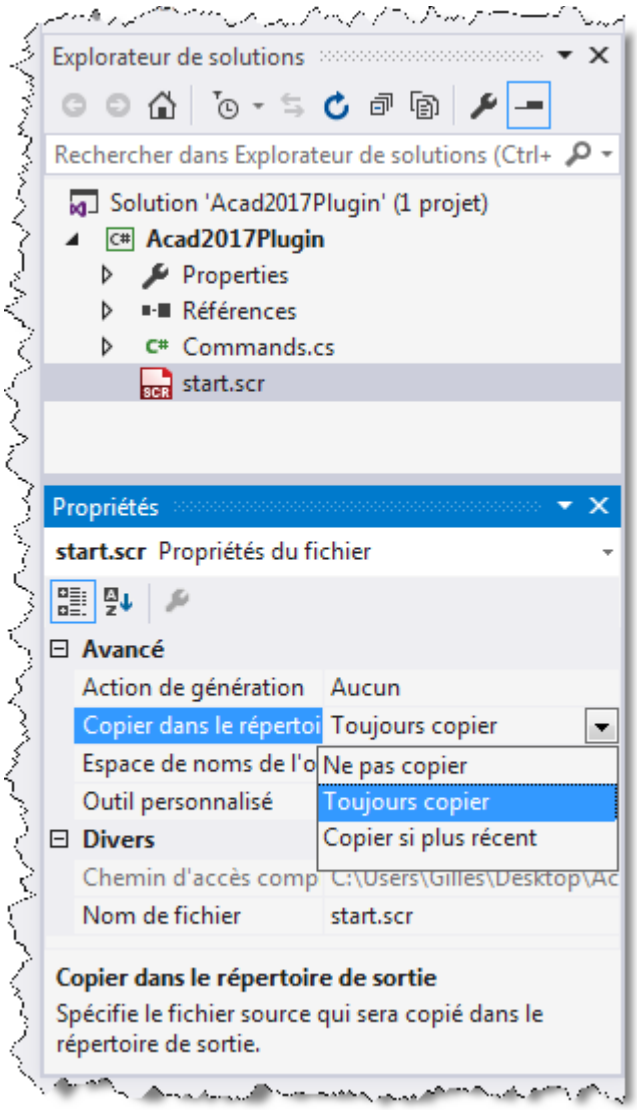
Dans la boite dialogue *Ajouter un nouvel élément*, choisir : *Fichier texte*. Renommer le fichier : start.scr et cliquer sur *Ajouter.*

Le fichier s'ouvre dans la fenêtre d'édition, ajouter l'instruction suivante sans oublier une espace à la fin.

```
netload "Acad2017Plugin.dll"
```

Enregistrer le fichier start.scr.

Dans les propriétés de start.scr (sélectionner start.scr dans l'explorateur de solutions), mettre la propriété *Copier dans le répertoire* sur *Toujours copier.*



Note : avec AutoCAD 2016 et 2017, il est impératif que la variable système LEGACYCODESEARCH soit à 1 pour que AutoCAD intègre le répertoire de sortie (.\ Acad2017Plugin\bin\Debug) dans les chemins de recherche et ainsi touve la DLL et le fichier de script.

Modifier le fichier de projet MSBuild pour lancer AutoCAD au débogage

Les fichiers de projet MSBuild (.csproj) sont des fichiers xml qui décrivent et contrôlent le processus de génération des applications.
C'est dans ce fichier que doivent être ajoutées les instructions qui permettront de lancer AutoCAD depuis Visual Studio en mode débogage et de charger la DLL au démarrage d'AutoCAD.

Générer la solution et fermer Visual Studio pour pouvoir modifier le fichier Acad2017Plugin.csproj.

Ouvrir le fichier .\Acad2017Plugin\Acad2017Plugin\Acad2017Plugin.csproj avec un éditeur de texte (bloc-note, notepad++, etc.).

Ajouter les nœuds suivants à la fin du nœud *PropertyGroup* correspondant à la génération en mode Debug (le deuxième nœud *PropertyGroup* du fichier), après avoir modifié, si nécessaire, le chemin du fichier acad.exe

correspondant à la version d'AutoCAD ciblée et celui du fichier start.scr (le répertoire de sortie pour le mode Debug).

```
<StartAction>Program</StartAction>
<StartProgram>C:\Program Files\Autodesk\AutoCAD 2017\acad.exe</StartProgram>
<StartArguments>/nologo /b "start.scr"</StartArguments>
```

StartAction indique que la génération du projet doit lancer un programme (ce qui n'est généralement pas le cas avec une DLL).
StartProgram spécifie le programme à démarrer, ici AutoCAD.
StartArguments contient les arguments qui vont lancer le script qui chargera la DLL.

Le nœud *PropertyGroup* devrait ressembler à ça :

```
<PropertyGroup Condition=" '$(Configuration)|$(Platform)' == 'Debug|AnyCPU' ">
  <DebugSymbols>true</DebugSymbols>
  <DebugType>full</DebugType>
  <Optimize>false</Optimize>
  <OutputPath>bin\Debug\</OutputPath>
  <DefineConstants>DEBUG;TRACE</DefineConstants>
  <ErrorReport>prompt</ErrorReport>
  <WarningLevel>4</WarningLevel>
  <StartAction>Program</StartAction>
  <StartProgram>C:\Program Files\Autodesk\AutoCAD 2017\acad.exe</StartProgram>
  <StartArguments>/nologo /b "start.scr"</StartArguments>
</PropertyGroup>
```

Dans le nœud *ItemGroup* se trouvent les références ajoutées au projet.

Si le répertoire de la solution est sur le même lecteur que celui dans le quel se trouvent les DLLs référencées (le répertoire d'installation d'AutoCAD ou celui d'ObjectARX 20##), les chemins enregistrés sont relatifs.
Comme le modèle sera exporté dans un répertoire différent de celui de la solution, il faut remplacer ces chemins relatifs par les chemins absolus.

Par exemple, remplacer les chemins relatifs suivant :

```
<Reference Include="AcCoreMgd">
  <HintPath>..\..\..\..\..\..\..\ObjectARX 2017\inc\AcCoreMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
<Reference Include="AcDbMgd">
  <HintPath>..\..\..\..\..\..\..\ObjectARX 2017\inc\AcDbMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
<Reference Include="AcMgd">
  <HintPath>..\..\..\..\..\..\..\ObjectARX 2017\inc\AcMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
```

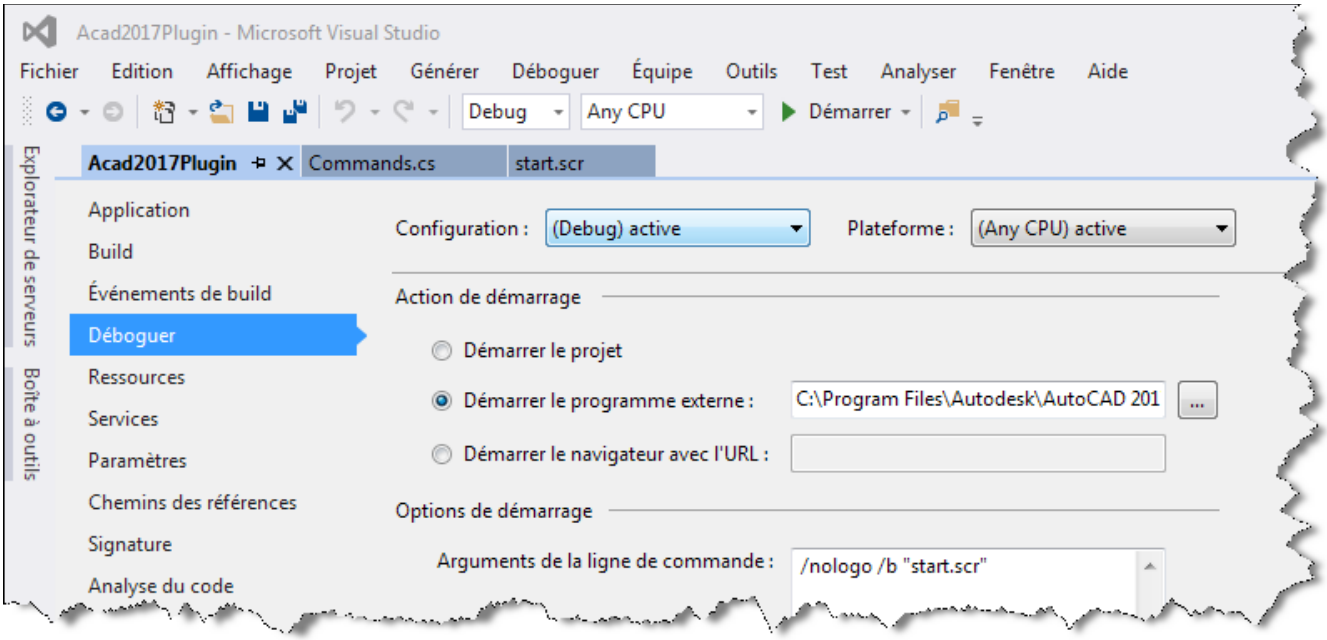
Par ceux-ci :

```
<Reference Include="AcCoreMgd">
  <HintPath>C:\ObjectARX 2017\inc\AcCoreMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
<Reference Include="AcDbMgd">
  <HintPath>C:\ObjectARX 2017\inc\AcDbMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
```

```
<Reference Include="AcMgd">
  <HintPath>C:\ObjectARX 2017\inc\AcMgd.dll</HintPath>
  <Private>False</Private>
</Reference>
```

Enregistrer le fichier Acad2017Plugin.csproj.

Ces changements devraient apparaître dans Visual Studio, dans l'onglet *Déboguer* des propriétés du projet.



Exporter le modèle

Ré-ouvrir la solution Acad2017Plugin.sln dans Visual Studio.
Lancer un débogage (F5) pour contrôler le bon fonctionnement. AutoCAD devrait s'ouvrir et afficher en ligne de commande :

```
Commande: netload Nom du fichier d'assemblage: "Acad2017Plugin.dll"
```

Le message d'erreur ci-dessous indique soit un nom de DLL erroné dans le fichier start.scr ; soit que le fichier start.scr n'a pas été copié dans le répertoire de sortie (CF propriété *Copier dans le répertoire de sortie*).

```
Commande: netload Nom du fichier d'assemblage: "Acad2017Plugin.dll" Impossible de
charger l'assemblage. Détails de l'erreur: System.IO.FileNotFoundException:
Impossible de charger le fichier ou l'assembly...
```

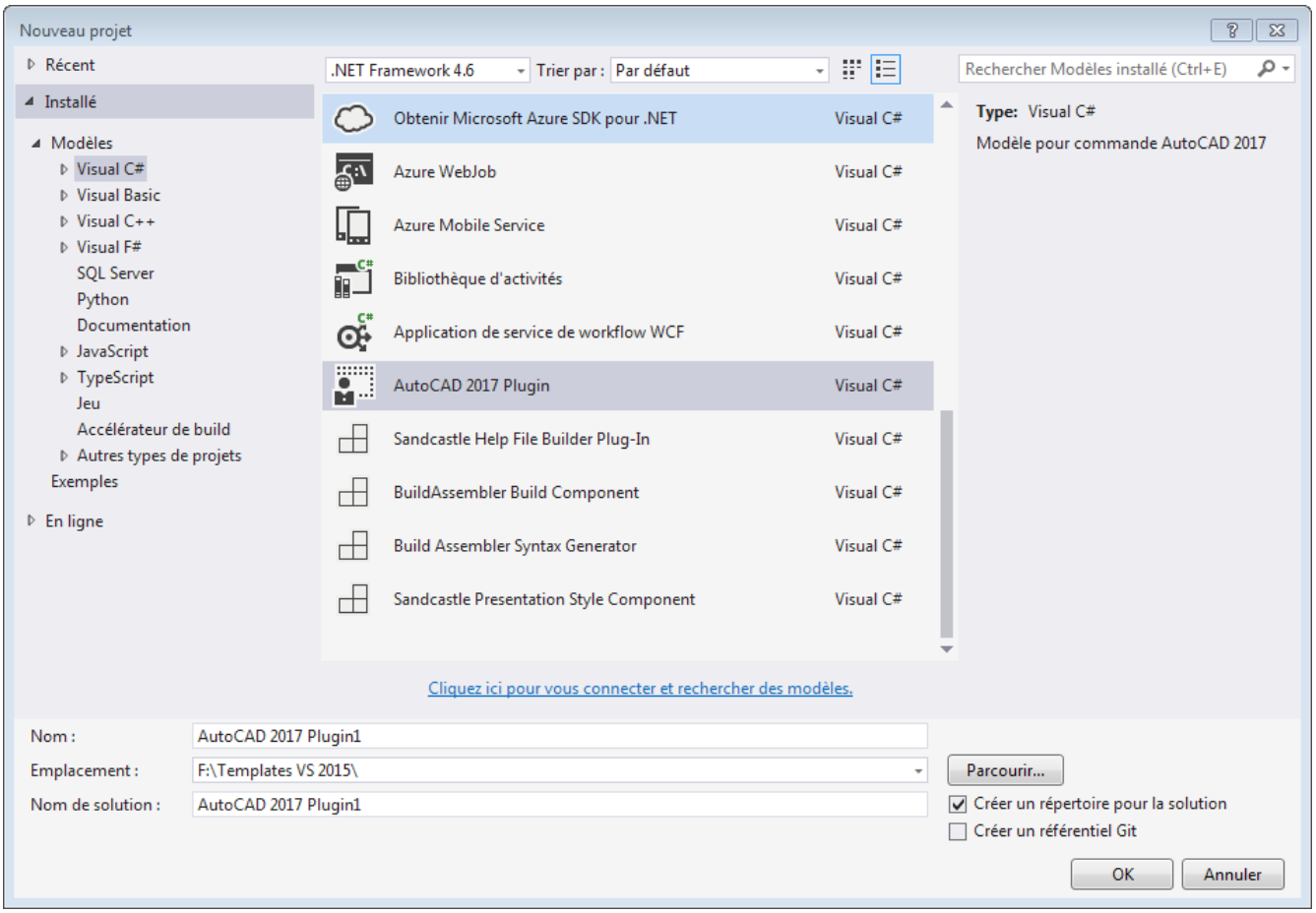
Si tout fonctionne correctement, il est temps de générer le modèle.

Dans le menu *Fichier*, choisir Exporter le modèle...
Dans la boite de dialogue *Assistant Exportation de modèle*, laisser coché *Modèle de projet* et faire *Suivant*.
Changer le nom du modèle comme désiré, *AutoCAD 2017 Plugin*, par exemple et ajouter éventuellement une description : *Modèle pour commande AutoCAD 2017*.
Terminer.

Avec les options par défaut, le modèle est exporté sous forme de fichier ZIP dans les dossiers :
Visual Studio 2017\Templates\ProjectTemplates et *Visual Studio 2017\My Exported Templates*
du répertoire Documents ou Mes documents.

Visual Studio utilise les modèles situés dans *ProjectTemplates*, *My Exported Templates* sert plutôt de sauvegarde.

Ce modèle sera désormais proposé par Visual Studio au démarrage d'un nouveau projet.



En l'état, renommer le projet dans la boite de dialogue sera automatiquement répercuté dans le projet généré pour l'espace de nom par défaut et le nom du projet (DLL), par contre, le nom de la DLL à charger dans le script demeurera inchangé.
Pour que le nom du projet soit automatiquement répercuté dans le script, il va falloir modifier le fichier : *start.scr* et le fichier de configuration du modèle : *My Template.vstemplate* qui se trouvent dans : *Visual Studio 2015\Templates\ProjectTemplates\AutoCAD 2017 Plugin.zip.*

Extraire le contenu de l'archive *AutoCAD 2017 Plugin.zip* et ouvrir *start.scr* dans un éditeur de texte.

Remplacer :

```
netload "Acad2017Plugin.dll"
```

par :

```
netload "$projectname$.dll"
```

Ne pas oublier une espace pour validation à la fin du script, enregistrer les modifications,

Ouvrir le fichier *My Template.vstemplate* dans un éditeur de texte (il s'agit d'un fichier XML) et remplacer :
```
<ProjectItem ReplaceParameters="false" TargetFileName="start.scr">start.scr</ProjectItem>
```
par :
```
<ProjectItem ReplaceParameters="true" TargetFileName="start.scr">start.scr</ProjectItem>
```

On peut aussi ajouter la version du Framework requise de manière à ce que Visual Studio ne propose ce modèle que lorsque cette version du Framework est sélectionnée, ajouter :
```
<RequiredFrameworkVersion>4.6</RequiredFrameworkVersion>
```
après la balise : `<Description>.`

Le contenu du fichier devrait finalement ressembler à ça :

```
<VSTemplate Version="3.0.0" xmlns="http://schemas.microsoft.com/developer/vstemplate/2005"
Type="Project">
  <TemplateData>
    <Name>AutoCAD 2017 Plugin</Name>
    <Description>Modèle pour commande AutoCAD2017</Description>
    <RequiredFrameworkVersion>4.6</RequiredFrameworkVersion>
    <ProjectType>CSharp</ProjectType>
    <ProjectSubType>
    </ProjectSubType>
    <SortOrder>1000</SortOrder>
    <CreateNewFolder>true</CreateNewFolder>
    <DefaultName>AutoCAD 2017 Plugin</DefaultName>
    <ProvideDefaultName>true</ProvideDefaultName>
    <LocationField>Enabled</LocationField>
    <EnableLocationBrowseButton>true</EnableLocationBrowseButton>
    <Icon>__TemplateIcon.ico</Icon>
    </TemplateData>
    <TemplateContent>
    <Project TargetFileName="Acad2017Plugin.csproj" File="Acad2017Plugin.csproj"
ReplaceParameters="true">
    <ProjectItem ReplaceParameters="true" TargetFileName="Commands.cs">Commands.cs</ProjectItem>
    <Folder Name="Properties" TargetFolderName="Properties">
    <ProjectItem ReplaceParameters="true"
TargetFileName="AssemblyInfo.cs">AssemblyInfo.cs</ProjectItem>
    </Folder>
    <ProjectItem ReplaceParameters="true" TargetFileName="start.scr">start.scr</ProjectItem>
    </Project>
  </TemplateContent>
</VSTemplate>
```

Enregistrer les modifications et recréer l'archive *AutoCAD 2017 Plugin.zip* avec les fichiers modifiés. Remplacer l'ancienne archive dans *Visual Studio 2015\Templates\ProjectTemplates.*

Conclusion

La même procédure peut être utilisée pour créer d'autres modèles, ciblant d'autres versions d'AutoCAD et/ou du Framework, ou encore pour d'autres types de projets (fonctions LISP, bibliothèques de classe, extensions d'application, ...).

Il est possible de rajouter des instructions dans le fichier start.scr comme le lancement d'une commande.

Il est aussi possible de spécifier un fichier DWG à ouvrir dans les *Arguments de la ligne de commande* (onglet Déboguer des propriétés du projet) avec un chemin complet ou relatif (la racine étant le répertoire de sortie).

Exemple :

```
"Dessin1.dwg" /nologo /b "start.scr"
```