CLI .NET Core

CHEAT SHEET

Vincent BOURDON

Août 2018

Cabinet de conseil IT et Agilité | SOAT

CLI .NET CORE

La CLI .NET Core (Command-line interface, en français, interface en ligne de commande) est un nouvel utilitaire multiplateforme Windows, MacOs et Linux qui permet la gestion, la compilation et l'exécution de projets .NET Core.

Les IDEs et les chaînes d'intégration continue peuvent alors s'appuyer dessus pour orchestrer l'automatisation des systèmes de build.

UTILISATION

La CLI .NET Core est composée de quatre parties :

- Le driver `dotnet` qui permet l'exécution d'une application .Net ou l'exécution d'une commande.
- La commande qui exécute une action. Par exemple, `build`.
- Les arguments de la commande. Par exemple, le chemin vers le projet à exécuter.
- Les options de la commande. Par exemple l'option `--output` de la commande `publish`.

OPTIONS GÉNÉRALES

Afficher l'aide pour une commande	`dotnet build -h`
Afficher la version du SDK utilisée, les informations sur l'environnement et la liste des SDK et Runtime installés	`dotnet --info`
Afficher la version du SDK .NET Core en cours d'utilisation	`dotnet --version`
Afficher la liste des SDK installés	`dotnet --list-sdks`
Afficher la liste des runtimes installés	`dotnet --list-runtime`

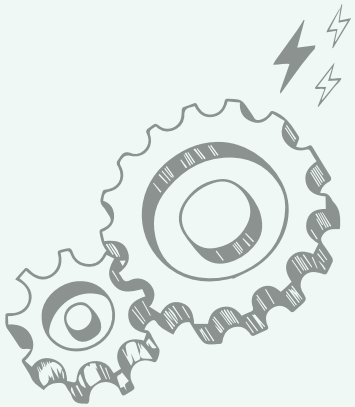
INITIALISER DES PROJETS ET DES FICHIERS

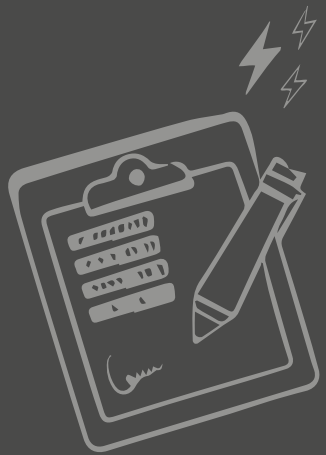
La commande new permet de générer des projets ou des fichiers à partir de modèles préinstallés avec le SDK .NET Core ou à installer depuis des packages NuGet.

Lister les modèles installés	`dotnet new -l`
Créer une solution vide	`dotnet new sln`
Créer un projet console en c#	`dotnet new console`
Créer un projet de librairie en VB .NET	`dotnet new classlib -lang VB`
Créer un projet ASP.NET Core Web API en F# dans le dossier fsharp-API	`dotnet new webapi -lang F# -o ./fsharp-API`
Créer un projet de tests avec Xunit nommé 'tests'	`dotnet new xunit -n tests`
Installer un nouveau modèle (SAFE) depuis NuGet	`dotnet new -i SAFE.Template`

GESTION DE SOLUTION

Ajouter un projet à une solution	`dotnet sln Awesome.sln add WebAPI.csproj`
Lister les projets d'une solution	`dotnet sln Awesome.sln list`
Supprimer un projet d'une solution	`dotnet sln Awesome.sln remove WebAPI.cspro`





GESTION DES PACKAGES

Depuis .NET Core 2.0 et dans des scénarios de restauration par défaut, il n'est plus nécessaire d'exécuter `dotnet restore`, la commande est exécutée implicitement par toutes les commandes qui nécessitent une restauration.

Restaurer les dépendances spécifiées dans le/les projet(s) .Net	`dotnet restore`
Ajouter un package au projet	`dotnet add package Microsoft.AspNet.SignalR`
Ajouter un package en spécifiant la version	`dotnet add package Newtonsoft.Json --version 11.0.2`
Ajouter un package sans effectuer de restauration	`dotnet add package Newtonsoft.Json -n`
Restaurer le package dans le répertoire spécifié	`dotnet add package OpenCover --package-directory ./build-tools`
Utiliser une source de package NuGet spécifique	`dotnet add package AspNet.Security.OAuth.Discord -s https://www.myget.org/F/aspnet-contrib/api/v3/index.json`
Supprimer un package	`dotnet remove package Newtonsoft.Json`

GESTION DES RÉFÉRENCES

Ajouter une référence de projet à un projet	`dotnet add WebApi.csproj reference DomainModel.csproj`
Répertorier des références de projet	`dotnet list WebApi.csproj reference`
Supprimer une référence d'un projet	`dotnet remove WebApi.csproj reference DomainModel.csproj`

Prérequis

Créez un compte gratuit sur nuget.org ou utilisez un compte Microsoft.

Créer un package sur le serveur NuGet	`dotnet pack`
Publier un package sur le serveur NuGet	`dotnet nuget push foo.nupkg`
Supprimer ou retirer un package du serveur	`dotnet nuget delete Microsoft.AspNetCore.Mvc 1.`
Effacer tous les packages de tous les répertoires du cache local	`dotnet nuget locals --clear all`

COMPILATION ET PUBLICATION

CLI .NET Core

Générer un projet .Net et ses dépendances	`dotnet build`
Publier un projet .Net à des fins de déploiement, sur un environnement où le runtime est installé	`dotnet publish`
Publier un projet .Net à des fins de déploiement runtime MacOs inclus	`dotnet publish --runtime sx.10.11-x64`
Exécuter Microsoft Build Engine (MSBuild) pour compiler en release en utilisant tous les processeurs	`dotnet msbuild /p:Configuration=Release /m`
Nettoyer les sorties de build (obj et bin)	`dotnet clean`
Stocker les assemblys du projet *foo* dans le magasin de runtimes 2.1.0	`dotnet store -m foo.csproj -f 2.1.0`

EXÉCUTION

Compiler et exécuter immédiatement un projet .Net à partir du code source	`dotnet run -p WepApi.csproj`
Exécuter sans compiler	`dotnet run --no-build`
Exécuter la configuration release	`dotnet run -c Release`
Spécifier la version du runtime .Net Core à utiliser pour exécuter l'application	`dotnet run --fx-version 2.0`
Définir le niveau de verbosité en sortie de la commande. Les valeurs autorisées sont q[uiet], m[inimal], n[ormal], d[etailed] et diag[nostic].	`dotnet run -v d`
Exécuter un assembly déjà compilé (avec build ou publish)	`dotnet WebApi.dll`

TESTS

- Exécuter des tests unitaires à l'aide du programme *Test Runner* spécifié dans le projet

```
dotnet test
```

- Installer *Xunit* comme framework de test
- Et en tant que *Test Runner*

```
dotnet add package xunit

dotnet add package xunit.runner.visualstudio
```

- Obtenir la couverture de test avec *coverlet*

```
dotnet add package coverlet.msbuild
dotnet test /p:CollectCoverage=true
/p:CoverletOutputFormat=lcov
```

Tips : Utilisez *Coverage Gutters* dans VS code pour afficher la couverture

Plus d'informations sur https://github.com/tonerdo/coverlet/

CLI EXTENSIBILITÉ ET OUTILS (1/3)

Utilisation des outils par projet

Ajouter un élément <DotNetCliToolReference> qui référence l'outil à utiliser. Puis exécuter un dotnet restore.

```
<ItemGroup>
       <DotNetCliToolReference Include="Amazon.ECS.Tools" Version="1.0.0" />
</ItemGroup>
```

Msbuild Targets

La CLI reposant sur MSbuild, il est désormais possible d'utiliser des targets MSbuild.

Paket utilise ce principe pour s'intégrer avec la commande dotnet restore.

```
<Project Sdk="Microsoft.NET.Sdk">
  ...
  <Import Project="..\.paket\Paket.Restore.targets" />
  ...
</Project>
```

CLI EXTENSIBILITÉ ET OUTILS (2/3)

Outil global

Depuis le framework 2.1.300, il est possible d'installer des outils complémentaires à la CLI.

- Installer une commande qui lance un serveur HTTP

```
dotnet tool install --global dotnet-serve
dotnet serve -o
```

- Lister tous les outils installés de façon globale

```
dotnet tool list -g-
```

- Désinstaller un outil

```
dotnet tool uninstall dotnet-serve -g
```

- Mettre à jour un outil

```
dotnet tool update dotnet-serve -g
```

CLI EXTENSIBILITÉ ET OUTILS (3/3)

Outils déjà intégrés

Certains outils sont désormais intégrés de base à la CLI.

- **dotnet-watch :** Lance une commande lorsque qu'une modification de fichier est détectée

```
dotnet watch run
```

- **dotnet-user-secrets :** Gère les *user-secrets*, cette commande est utilisée pour stocker des mots de passe et des chaînes de connexion.

```
dotnet user-secrets set "Movies:ConnectionString" "Data
Source=.;Initial Catalog=MoviesDB;Integrated Security=True;"
```

- **dotnet-dev-certs :** Génère des certificats utilisés par ASP.NET Core pour le serveur web de développement.

```
dotnet dev-certs https
```

- **dotnet-sql-cache :** Met en place un cache distribué en utilisant une base de données SQL Server

```
dotnet sql-cache create "Data Source=(localdb)\v11.0;Initial
Catalog=DistCache;Integrated Security=True;" dbo TestCache
```

- **dotnet-ef :** Outil supplémentaire pour travailler avec Entity Framework Core.

Voir la Cheet Sheat Entity Framework CLI pour plus d'informations

NOTES

Vincent BOURDON
Développeur Senior - SOAT

Vincent est passionné par tout ce qui est high-tech et design. Expert du web, domaine sur lequel il a débuté avec l'ASP classic et Visual Studio InterDev, il est naturellement passé sur les technologies .Net en 2001 (VB.Net puis c#). Avant tout développeur, Vincent a effectué de nombreuses missions en tant que consultant, architecte, expert, formateur et même coach Craftsman.

SOAT
89 quai Panhard et Levassor 75013 Paris
tél. : + (33) 1.44.75.42.55
contact@soat.fr - www.soat.fr

Retrouvez-nous sur

Cabinet de conseil IT et Agilité | SOAT

Conception graphique et réalisation : MAP ADVERTISING