eBook Gratuit

APPRENEZ Visual Studio

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#visual-studio

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: visual-studio

It is an unofficial and free Visual Studio ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Visual Studio.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec Visual Studio

Remarques

Visual Studio est un environnement de développement intégré (IDE) de Microsoft. Il permet au développeur de travailler avec différents types de projets, y compris Windows Forms, les applications de console, les plug-ins Office et les applications Windows universelles.

L'EDI prend en charge plusieurs langages de programmation, les plus courants étant Visual C #, Visual Basic, Visual F # et Visual C ++.

Il existe plusieurs éditions de Visual Studio: Communauté (gratuite), Express (gratuite), Professionnel, Entreprise et Intégrale (Cependant, toutes ne sont pas disponibles pour toutes les versions).

Versions

Version	Nom de code	Numéro de version	.NET pris en charge Versions du framework	Rendez-vous amoureux
97	Boston	5.0	N / A	1997-02-01
6,0	Tremble	6,0	N / A	1998-06-01
.NET 2002	Rainier	7.0	1.0	2002-02-13
.NET 2003	Everett	7.1	1.1	2003-04-24
2005	Whidbey	8.0	2.0, 3.0	2005-11-07
2008	Les orques	9.0	2.0, 3.0, 3.5	2007-11-19
2010	Dev10 / Rosario	10.0	2.0 - 4.0	2010-04-12
2012	Dev11	11.0	2.0 - 4.5.2	2012-09-12
2013	Dev12	12,0	2.0 - 4.5.2	2013-10-17
2013.1 (mise à jour 1)				2014-01-20
2013.2 (mise à jour 2)				2014-05-12

Version	Nom de code	Numéro de version	.NET pris en charge Versions du framework	Rendez-vous amoureux
2013.3 (mise à jour 3)				2014-08-04
2013.4 (mise à jour 4)				2014-11-12
2013.5 (mise à jour 5)				2015-07-20
2015	Dev14	14.0	2,0 - 4,6	2015-07-20
2015.1 (mise à jour 1)				2015-11-30
2015.2 (mise à jour 2)				2016-03-30
2015.3 (mise à jour 3)				2016-06-27
"15" Aperçu	Dev15	15.0	2,0 - 4,6,2; Core 1.0	2016-03-30
"15" Aperçu 2				2016-05-10
"15" Aperçu 3				2016-07-07
"15" Aperçu 4				2016-08-22
"15" Aperçu 5				2016-10-05
2017	Dev15	15,0 - 15,2	3,5 - 4,7; Core 1.0 - 1.1	2017-03-07

Examples

Installation ou configuration

Visual Studio peut être téléchargé et installé gratuitement dans l' *édition Comunity à* partir du site Microsoft et peut également être trouvé dans différentes versions . Cliquez simplement sur le bouton Télécharger et lancez l'exécutable, puis suivez les instructions.

Lire Démarrer avec Visual Studio en ligne: https://riptutorial.com/fr/visual-studio/topic/972/demarrer-avec-visual-studio

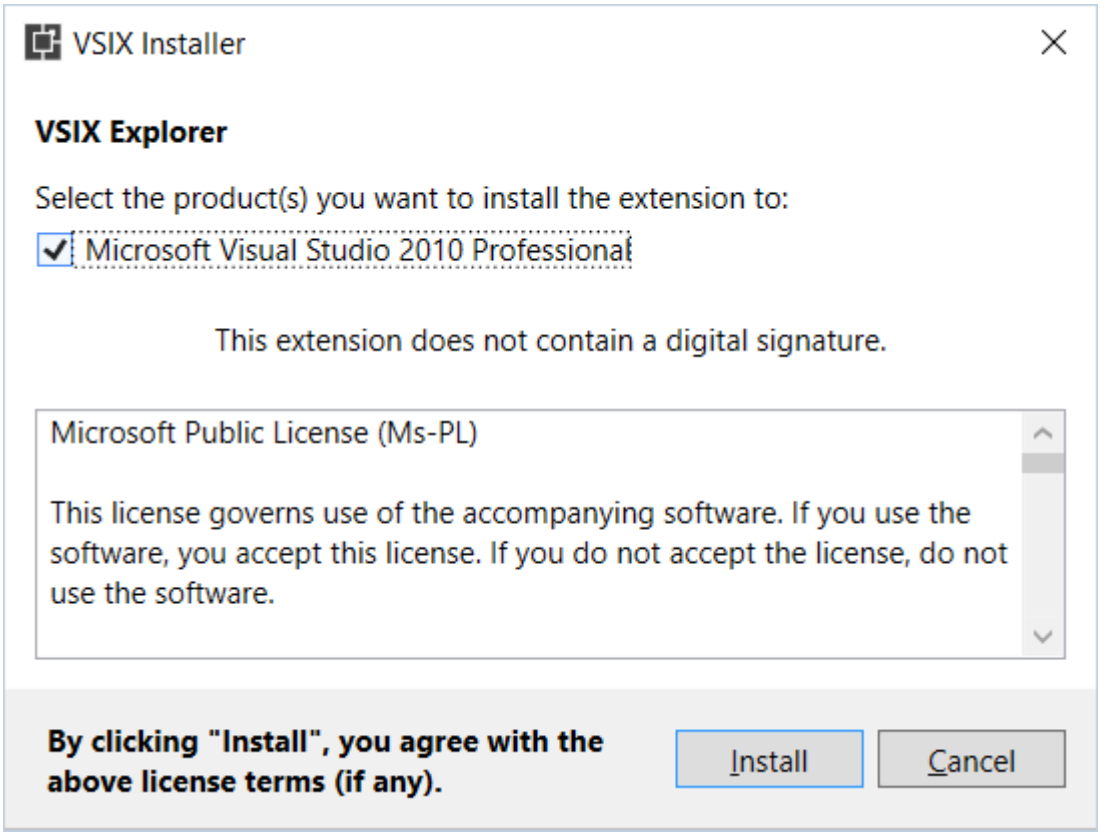
Chapitre 2: Ajouter une extension

Examples

Ajout d'une extension à Visual Studio en utilisant un fichier `VSIX`

Si vous avez un fichier `vsix` , vous pouvez l'installer en exécutant le fichier.

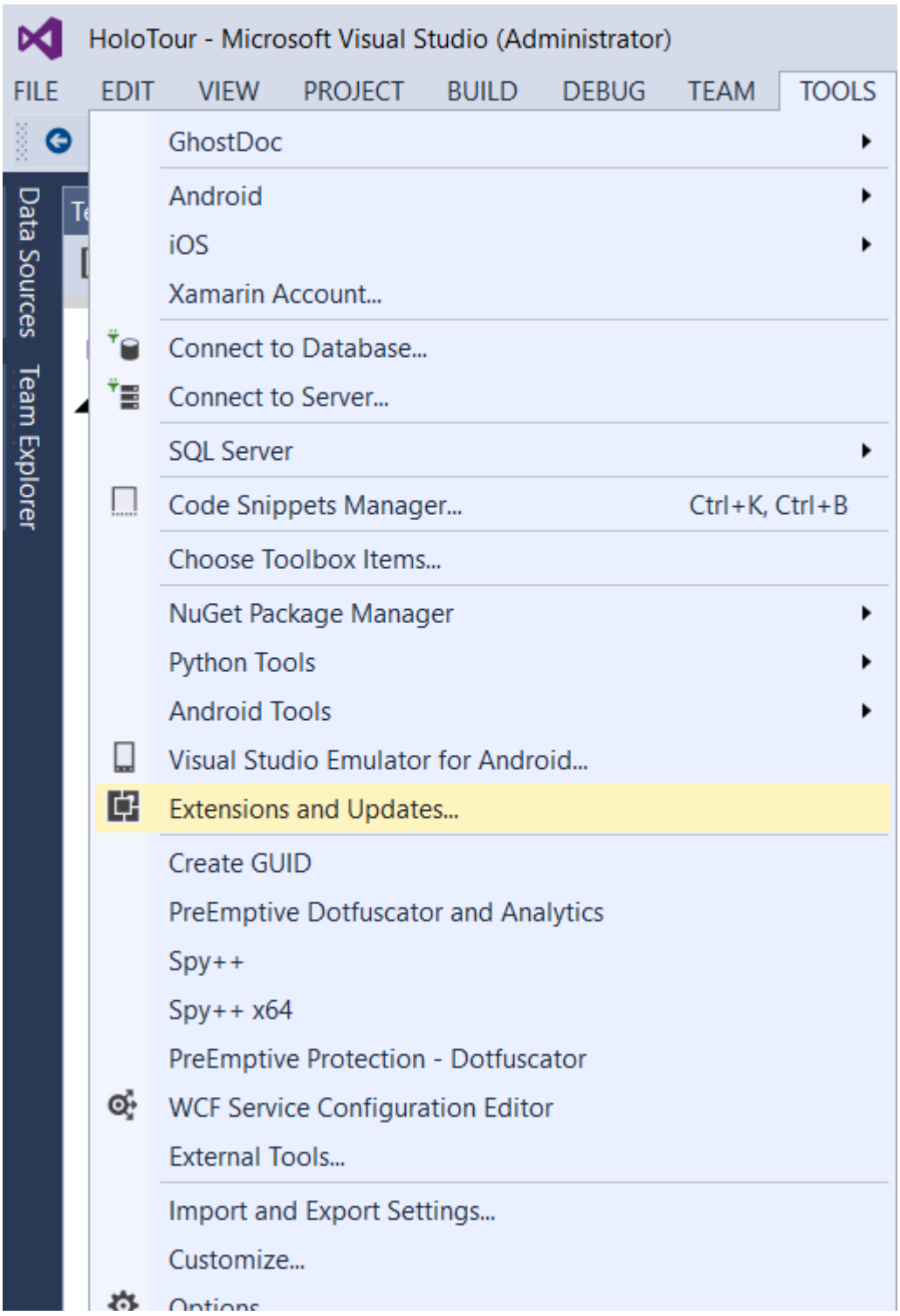
1. Obtenez le fichier `vsix` (il s'agit du programme d'installation de l'extension)
2. Exécutez le fichier.
3. Dans la fenêtre qui s'ouvre, confirmez l'installation.

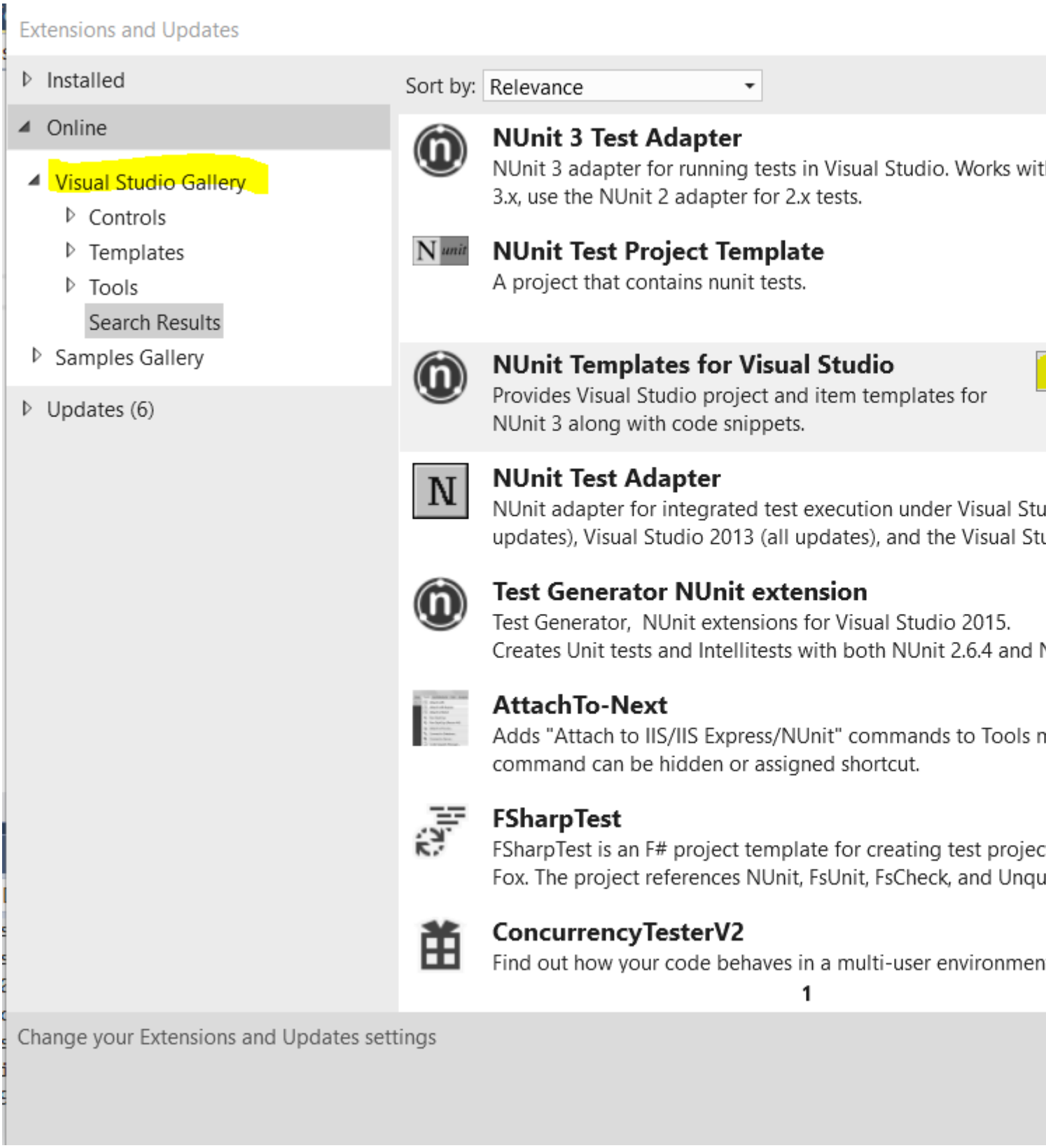


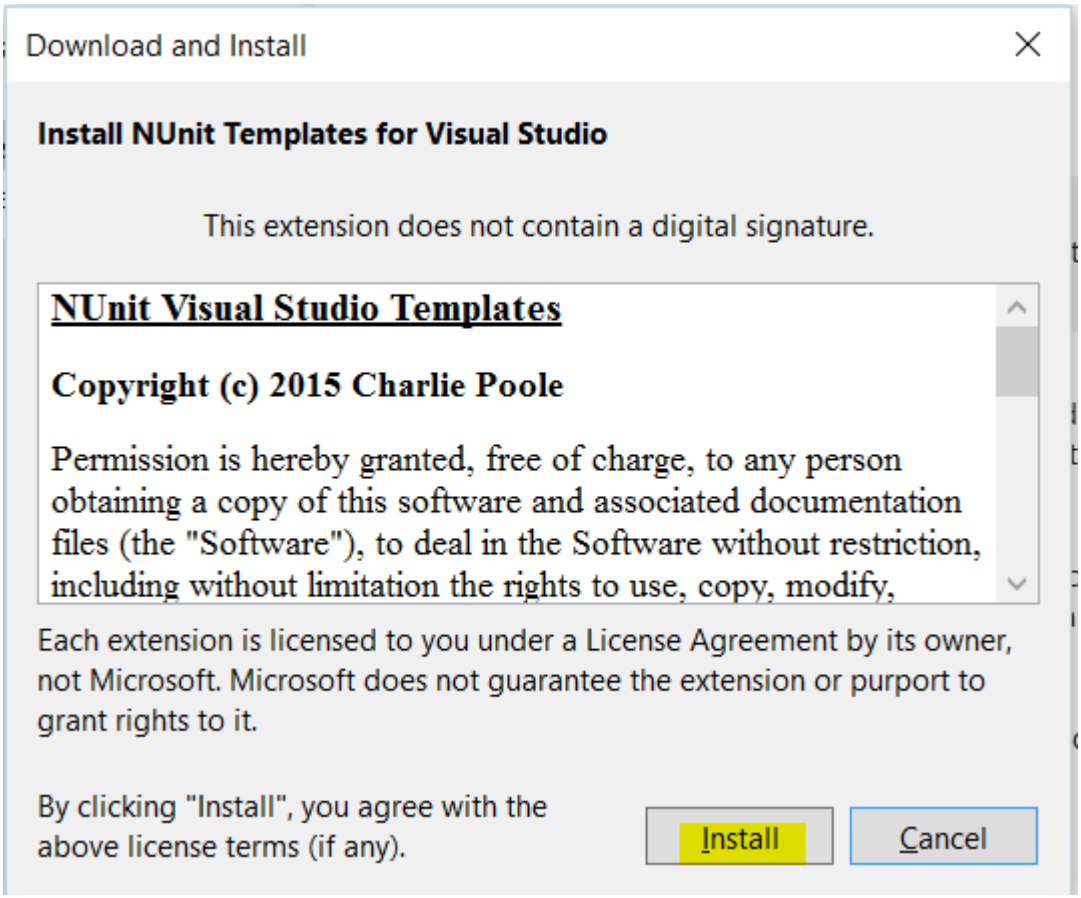
Ajout d'une extension à Visual Studio à partir de Visual Studio Gallery

En studio visuel

- allez dans **Outils** > **Extensions et mises à jour ...**
- Dans la fenêtre qui s'ouvre, allez en ligne
- Sélectionnez Visual Studio Gallery
- Vous pouvez rechercher une extension dans le champ de recherche situé dans le coin supérieur droit
- Sélectionnez l'extension que vous souhaitez ajouter
- Cliquez sur le téléchargement.
- Une fois le téléchargement terminé, cliquez sur le bouton **Installer** de la fenêtre qui s'ouvre.
- Pour utiliser l'extension, vous pouvez être invité à redémarrer Visual Studio.


HoloTour - Microsoft Visual Studio (Administrator)
FILE
EDIT
VIEW
PROJECT
BUILD
DEBUG
TEAM
TOOLS
Data Sources
Team Explorer
GhostDoc
Android
iOS
Xamarin Account...
Connect to Database...
Connect to Server...
SQL Server
Code Snippets Manager... Ctrl+K, Ctrl+B
Choose Toolbox Items...
NuGet Package Manager
Python Tools
Android Tools
Visual Studio Emulator for Android...
Extensions and Updates...
Create GUID
PreEmptive Dotfuscator and Analytics
Spy++
Spy++ x64
PreEmptive Protection - Dotfuscator
WCF Service Configuration Editor
External Tools...
Import and Export Settings...
Customize...
Options


Extensions and Updates
Installed
Online
Visual Studio Gallery
Controls
Templates
Tools
Search Results
Samples Gallery
Updates (6)
Sort by: Relevance
NUnit 3 Test Adapter
NUnit 3 adapter for running tests in Visual Studio. Works wit
3.x, use the NUnit 2 adapter for 2.x tests.
NUnit Test Project Template
A project that contains nunit tests.
NUnit Templates for Visual Studio
Provides Visual Studio project and item templates for
NUnit 3 along with code snippets.
NUnit Test Adapter
NUnit adapter for integrated test execution under Visual Stu
updates), Visual Studio 2013 (all updates), and the Visual Stu
Test Generator NUnit extension
Test Generator, NUnit extensions for Visual Studio 2015.
Creates Unit tests and Intellitests with both NUnit 2.6.4 and
AttachTo-Next
Adds "Attach to IIS/IIS Express/NUnit" commands to Tools m
command can be hidden or assigned shortcut.
FSharpTest
FSharpTest is an F# project template for creating test projec
Fox. The project references NUnit, FsUnit, FsCheck, and Unqu
ConcurrencyTesterV2
Find out how your code behaves in a multi-user environmen
1
Change your Extensions and Updates settings



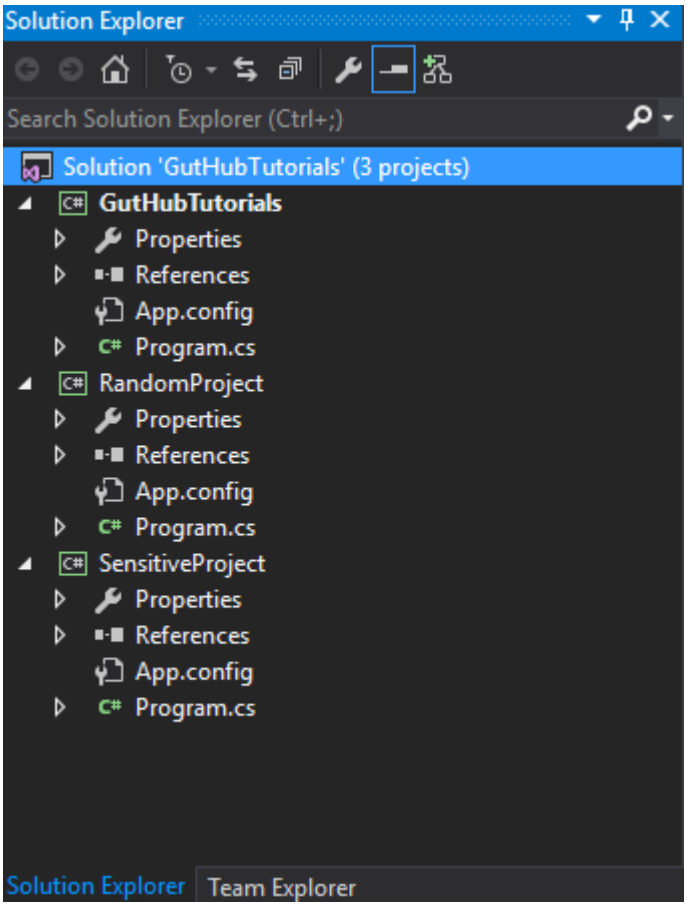
Lire Ajouter une extension en ligne: https://riptutorial.com/fr/visual-studio/topic/2257/ajouter-une-extension

Chapitre 3: Connecter votre projet de studio visuel à Github

Examples

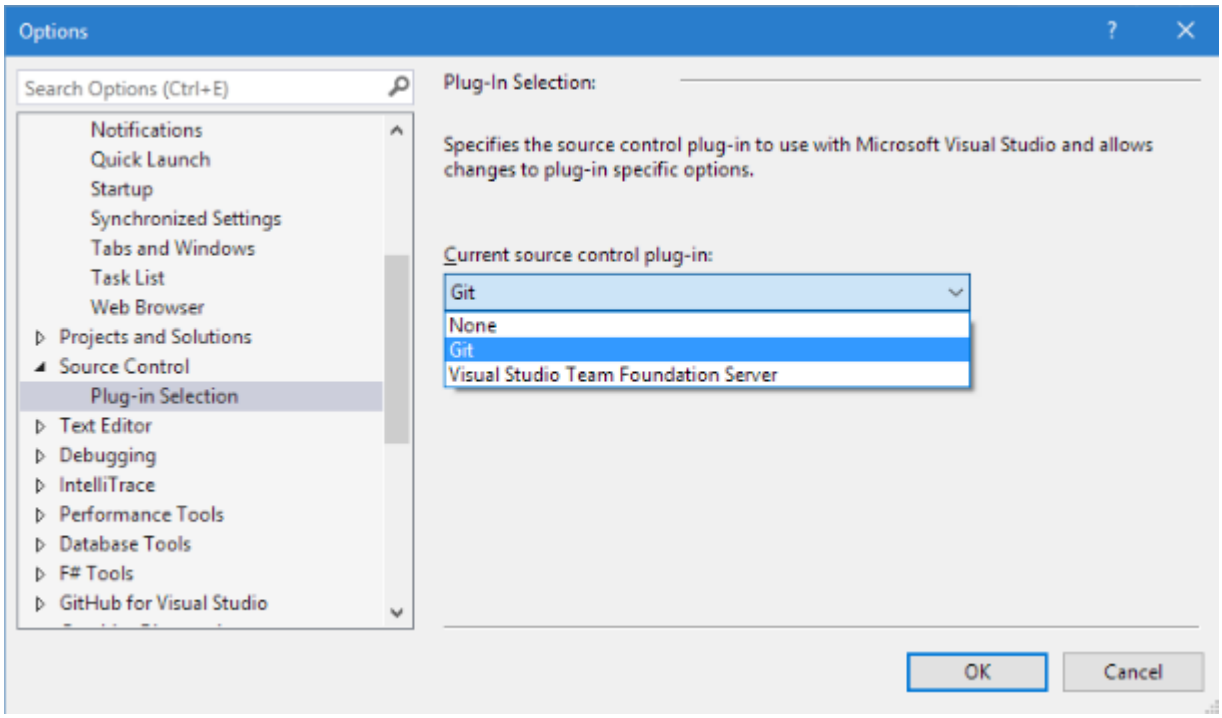
Publication de votre projet dans un référentiel github supprimant les données sensibles

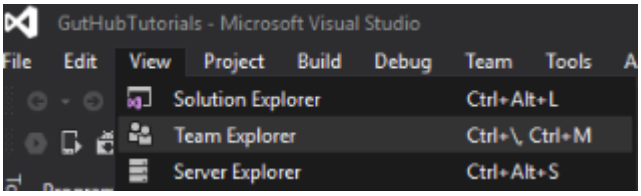
les étapes de cet exemple utiliseront la structure de projet suivante comme démonstration



et nous avons l'intention de l'exporter dans le dépôt "GHTuts" [Notez que le Repo n'existe pas encore sur github] mais laissez le "SensitiveProject" sans publier car il contient des mots de passe, des clés, etc.

1. Tout d'abord, nous nous assurons que le plug-in de contrôle de source est réglé sur "Git" dans "Outils> Options> Sélection de plug-in"



2. Si vous ne pouvez pas voir l'onglet "Team Explorer", affichez-le dans Visual Studio comme ceci 

---

1. Allez dans votre dossier de solution local et créez un nouveau fichier appelé ".gitignore.txt" [Note] cette étape est importante uniquement si vous avez des informations sensibles dans votre projet, sinon, laissez Visual Studio le créer pour vous

2. Maintenant, ouvrez le fichier ".gitignore.txt" et collez-le, ceci est un modèle pour ignorer les fichiers visuels communs de studio (vérifiez les liens ci-dessous)

```
## Ignore Visual Studio temporary files, build results, and
## files generated by popular Visual Studio add-ons.

# User-specific files
*.suo
*.user
*.userosscache
*.sln.docstates

# User-specific files (MonoDevelop/Xamarin Studio)
*.userprefs

# Build results
[Dd]ebug/
[Dd]ebugPublic/
[Rr]elease/
[Rr]eleases/
x64/
x86/
bld/
```

```
[Bb]in/
[Oo]bj/
[Ll]og/

# Visual Studio 2015 cache/options directory
.vs/
# Uncomment if you have tasks that create the project's static files in wwwroot
#wwwroot/

# MSTest test Results
[Tt]est[Rr]esult*/
[Bb]uild[Ll]og.*

# NUNIT
*.VisualState.xml
TestResult.xml

# Build Results of an ATL Project
[Dd]ebugPS/
[Rr]eleasePS/
dlldata.c

# DNX
project.lock.json
project.fragment.lock.json
artifacts/

*_i.c
*_p.c
*_i.h
*.ilk
*.meta
*.obj
*.pch
*.pdb
*.pgc
*.pgd
*.rsp
*.sbr
*.tlb
*.tli
*.tlh
*.tmp
*.tmp_proj
*.log
*.vspscc
*.vssscc
.builds
*.pidb
*.svclog
*.scc

# Chutzpah Test files
_Chutzpah*

# Visual C++ cache files
ipch/
*.aps
*.ncb
*.opendb
*.opensdf
```

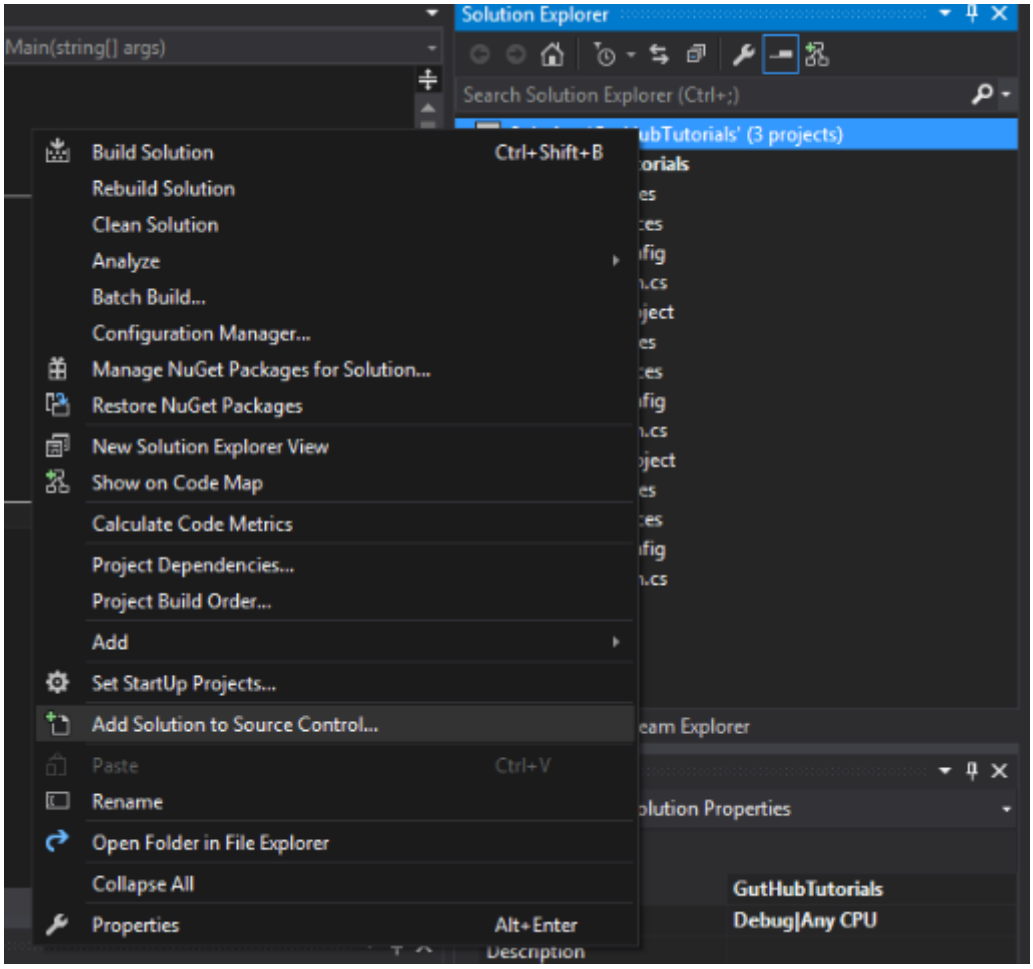
```
*.sdf
*.cachefile
*.VC.db
*.VC.VC.opendb

# Visual Studio profiler
*.psess
*.vsp
*.vspx
*.sap

# TFS 2012 Local Workspace
$tf/

# Guidance Automation Toolkit
*.gpState

# ReSharper is a .NET coding add-in
_ReSharper*/
*.[Rr]e[Ss]harper
*.DotSettings.user

# JustCode is a .NET coding add-in
.JustCode

# TeamCity is a build add-in
_TeamCity*

# DotCover is a Code Coverage Tool
*.dotCover

# NCrunch
_NCrunch_*
.*crunch*.local.xml
nCrunchTemp_*

# MightyMoose
*.mm.*
AutoTest.Net/

# Web workbench (sass)
.sass-cache/

# Installshield output folder
[Ee]xpress/

# DocProject is a documentation generator add-in
DocProject/buildhelp/
DocProject/Help/*.HxT
DocProject/Help/*.HxC
DocProject/Help/*.hhc
DocProject/Help/*.hhk
DocProject/Help/*.hhp
DocProject/Help/Html2
DocProject/Help/html

# Click-Once directory
publish/

# Publish Web Output
*.[Pp]ublish.xml
```

```
*.azurePubxml
# TODO: Comment the next line if you want to checkin your web deploy settings
# but database connection strings (with potential passwords) will be unencrypted
*.pubxml
*.publishproj

# Microsoft Azure Web App publish settings. Comment the next line if you want to
# checkin your Azure Web App publish settings, but sensitive information contained
# in these scripts will be unencrypted
PublishScripts/

# NuGet Packages
*.nupkg
# The packages folder can be ignored because of Package Restore
**/packages/*
# except build/, which is used as an MSBuild target.
!**/packages/build/
# Uncomment if necessary however generally it will be regenerated when needed
#!**/packages/repositories.config
# NuGet v3's project.json files produces more ignoreable files
*.nuget.props
*.nuget.targets

# Microsoft Azure Build Output
csx/
*.build.csdef

# Microsoft Azure Emulator
ecf/
rcf/

# Windows Store app package directories and files
AppPackages/
BundleArtifacts/
Package.StoreAssociation.xml
_pkginfo.txt

# Visual Studio cache files
# files ending in .cache can be ignored
*.[Cc]ache
# but keep track of directories ending in .cache
!*.[Cc]ache/

# Others
ClientBin/
~$*
*~
*.dbmdl
*.dbproj.schemaview
*.pfx
*.publishsettings
node_modules/
orleans.codegen.cs

# Since there are multiple workflows, uncomment next line to ignore bower_components
# (https://github.com/github/gitignore/pull/1529#issuecomment-104372622)
#bower_components/

# RIA/Silverlight projects
Generated_Code/
```

```
# Backup & report files from converting an old project file
# to a newer Visual Studio version. Backup files are not needed,
# because we have git ;-)
_UpgradeReport_Files/
Backup*/
UpgradeLog*.XML
UpgradeLog*.htm

# SQL Server files
*.mdf
*.ldf

# Business Intelligence projects
*.rdl.data
*.bim.layout
*.bim_*.settings

# Microsoft Fakes
FakesAssemblies/

# GhostDoc plugin setting file
*.GhostDoc.xml

# Node.js Tools for Visual Studio
.ntvs_analysis.dat

# Visual Studio 6 build log
*.plg

# Visual Studio 6 workspace options file
*.opt

# Visual Studio LightSwitch build output
**/*.HTMLClient/GeneratedArtifacts
**/*.DesktopClient/GeneratedArtifacts
**/*.DesktopClient/ModelManifest.xml
**/*.Server/GeneratedArtifacts
**/*.Server/ModelManifest.xml
_Pvt_Extensions

# Paket dependency manager
.paket/paket.exe
paket-files/

# FAKE - F# Make
.fake/

# JetBrains Rider
.idea/
*.sln.iml
```

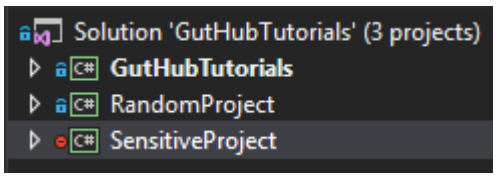
3. Ajoutez maintenant votre dossier de projet sensible au fichier ".gitignore.txt" à n'importe quelle ligne ne contenant pas de `#` , ajoutez-le donc à la toute fin, et cela devrait ressembler à ceci

```
# Paket dependency manager
.paket/paket.exe

# FAKE - F# Make
.fake/

# User-Ignored projects
SensitiveProject/
```

4. Faites un clic droit sur la solution et choisissez "Ajouter une solution au contrôle de code source ..." 

[Note] il pourrait vous être demandé de sauvegarder la solution avant de continuer

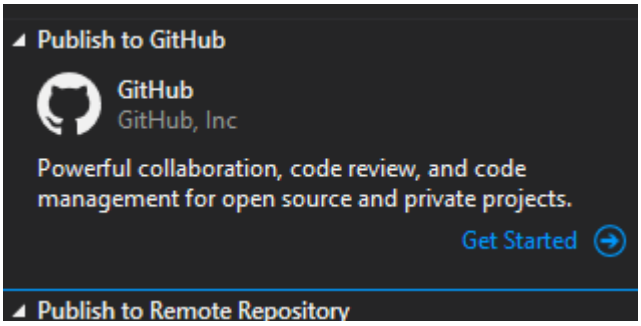
5. Maintenant, vous avez un repo git "LOCAL" sur votre PC, à partir duquel VS va lire, mais sans un repo github, et vous verrez une petite icône de verrou bleu à côté de chaque fichier de la solution ajoutée à git et un cercle rouge au projet ignoré

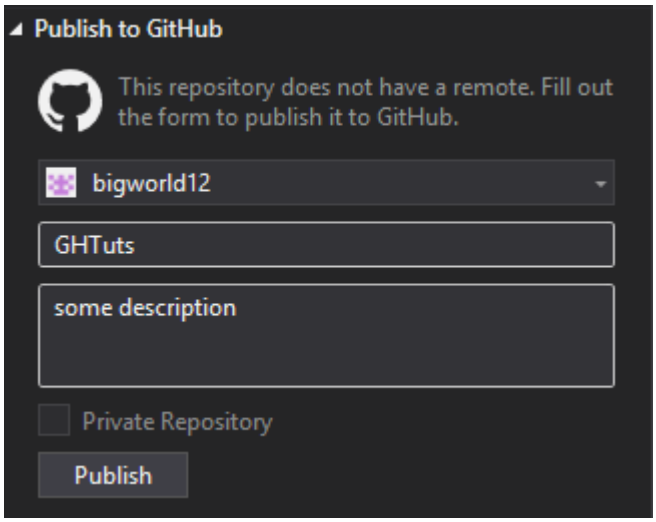


[Remarque]
pour plus d'informations sur le fichier .gitignore, vérifiez ces liens

- https://help.github.com/articles/ignoring-files/
- https://github.com/github/gitignore

7. Allez dans l'onglet "Team Explorer" puis "Sync"

8. Maintenant, nous créons un repo de vs en github comme celui-ci, appuyez sur le bouton "Get Started" 

9. Maintenant, remplissez vos informations dans github pour le nouveau dépôt, puis cliquez sur "Publier" 

10. Maintenant, quand nous allons à github, nous voyons que notre repo local a été publié sur github sans notre projet sensible [Note]
    l'url du repo ressemblera à ceci
    `https://github.com/<user name>/<repo name>`

Lire Connecter votre projet de studio visuel à Github en ligne: https://riptutorial.com/fr/visual-studio/topic/3826/connecter-votre-projet-de-studio-visuel-a-github

Chapitre 4: Contrats de code

Remarques

Pour bénéficier pleinement des contrats de code, vous devez installer l' extension pour Visual Studio. Il y a aussi un manuel d'utilisation des contrats de code .

Examples

Condition préalable standard

```
using System.Diagnostics.Contracts;

public int DivideNumbers(int numerator, int denominator)
{
    Contract.Requires(denominator != 0);

    return numerator / denominator;
}
```

Condition préalable qui lance une exception spécifique

```
using System.Diagnostics.Contracts;

public int DivideNumbers(int numerator, int denominator)
{
    Contract.Requires<ArgumentOutOfRangeException>(denominator != 0);

    return numerator / denominator;
}
```

Pré et postconditions

```
using System.Diagnostics.Contracts;

public int IncrementByRandomAmount(int input)
{
    Contract.Requires<ArgumentNullException>(input != null); // Don't allow null parameter.
    Contract.Requires<ArgumentOutOfRangeException>(input < int.MaxValue); // We can't do
anything if we're given int.MaxValue.
    Contract.Ensures(Contract.Result<int>() > input); // Return value will be greater than
input value.

    Random rnd = new Random();
    input += rnd.Next(1, 13); // Creates a number between 1 and 12 and adds it to input.

    return input;
}
```

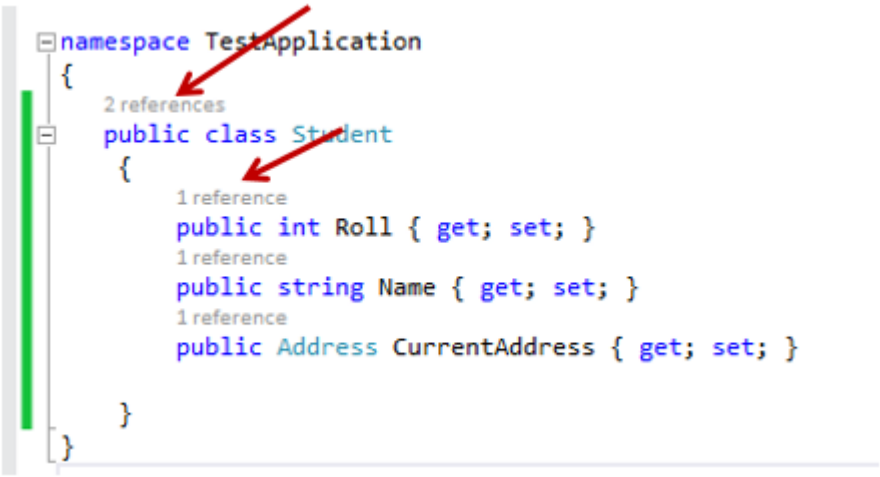
Lire Contrats de code en ligne: https://riptutorial.com/fr/visual-studio/topic/6311/contrats-de-code

Chapitre 5: Outils Visual Studio

Examples

Lentille de code

La lentille de code est un moyen simple de savoir ce qui se passe avec le code. Ici, vous pouvez trouver une image avec le nombre de références d'une méthode ou d'une classe.



Si vous ne pouvez pas voir l'objectif du code, veuillez vous reporter à la question suivante: les références manquantes de CodeLens comptent dans l'édition 2015 de la communauté VS

Des extraits

Intoduction

Depuis Visual Studio 2005, vous pouvez créer des extraits de code Intellisense. Cela vous permet de générer du code en tapant simplement un mot-clé et en appuyant deux fois sur la touche de `tabulation` .

En utilisant le code

Le code XML dont vous avez besoin pour créer un extrait de code Intellisense figure ci-dessous:

```
<?xml version="1.0" encoding="utf-8"?>

<CodeSnippets xmlns="http://schemas.microsoft.com/VisualStudio/CodeSnippet">
  <CodeSnippet Format="1.0.0"> <!-- format attribute is required -->

    <Header> <!-- 1 -->
      <Title></Title>
      <Author></Author>
```

```
      <Shortcut></Shortcut>
      <Description></Description>
      <Keywords>
        <Keyword>abc<Keyword>
        <Keyword>def<Keyword>
      </keywords>
    </Header>

    <Snippet> <!-- 2 -->

      <Imports> <!-- 2.1 -->
        <Import>
          <Namespace>System</Namespace>
        </Import>
      </Imports>

      <Declarations> <!-- 2.2 -->
        <Literal Editable="true/false"> <!-- 2.2.1 -->
          <ID>example</ID>
          <Type>System.String</Type>
          <ToolTip>A tip you can show</ToolTip>
          <Default>default value</Default>
          <Function></Function> <!-- 2.2.2 -->
        </Literal>

        <Object> <!-- 2.2.1 -->
          <ID>example</ID>
          <Type>System.String</Type>
          <ToolTip>A tip you can show</ToolTip>
          <Default>default value</Default>
          <Function></Function> <!-- 2.2.2 -->
        </Object>
      </Declarations>

      <References> <!-- 2.3 -->
        <Reference>
          <Assembly>System.Data.dll</Assembly>
        </Reference>
      </References>

      <Code Language=""> <!-- 2.4 -->
        <![CDATA[
            <!-- your code here if you use literals use dollar chars  -->
        ]]>
      </Code>

    </Snippet>

  </CodeSnippet>
</CodeSnippets>
```

Dans la balise d'extrait de code, vous avez deux balises obligatoires nommées En-tête et Snippet. Vous pouvez trouver plus d'informations dans les titres suivants. Le numéro près du nom correspond aux numéros du code ci-dessus.

Il peut y avoir zéro ou plusieurs éléments CodeSnippet ajoutés dans l'élément CodeSnippets.

1. en-tête

Dans la balise d'en-tête, vous pouvez placer des informations spécifiques sur l'extrait de code et ce qu'il fait. Les balises importantes que vous pouvez utiliser dans cette balise sont les suivantes:

Élément	La description
Titre	Le titre de l'extrait. Cet attribut est requis.
Auteur	L'auteur de l'extrait.
Raccourci	Est le raccourci, vous pouvez utiliser pour générer le code. Notez que cela ne peut contenir que des lettres et des chiffres et doit commencer par une lettre. **Remarque:** N'oubliez pas également de donner à l'extrait de code un nom et un raccourci appropriés. Sinon, vous rencontrerez des problèmes lorsque vous importerez l'extrait de code dans Visual Studio.
La description	Donne plus d'informations sur l'extrait de code si vous en avez besoin.
HelpUrl	Une URL pour une page d'aide sur Internet.
Mots clés	Groupe un ou plusieurs éléments de mot clé.
SnippetTypes	Groupes d'éléments `SnippetType` . Cet élément contient une valeur de texte et doit être l'une des valeurs suivantes. Les types d'extraits sont fusionnés avec une barre oblique. • `SurroundsWith` : Permet à l'extrait de code d'être placé autour d'un morceau de code sélectionné. • `Expansion` : Permet à l'extrait de code d'être inséré au niveau du curseur. • `Refactoring` : spécifie que l'extrait de code est utilisé lors du refactoring Visual C #. Le refactoring ne peut pas être utilisé dans des extraits de code personnalisés. Liste de sources: msdn.microsoft.com

Table source (mais modifications): msdn.microsoft.com

2. extrait

Dans l'étiquette de l'extrait de code, vous pouvez utiliser trois balises différentes. Cela peut être:

- Importations
- Déclarations
- Code (obligatoire)

- Les références

Celles-ci sont expliquées ci-dessous.

2.1 Importations

`Imports` contiennent les espaces de noms nécessaires au code. Utilisez la balise import dans cette balise et vous pouvez placer ici les espaces de noms nécessaires avec l'étiquette `Namespace` .

2.2 Déclarations

`Declarations` peuvent être utilisées pour déclarer certains littéraux ou objets dans votre code dans le `Code` -tag. Les enfants sont des littéraux et des objets.

2.2.1 Littéraux et objets

Les littéraux et les objets définissent les littéraux et les objets de l'extrait de code que vous pouvez modifier. Les fonctionnalités sont des littéraux et les objets sont les mêmes, mais ils ont une contrainte de type supplémentaire.

Le littéral et l'objet-tag peuvent contenir les enfants suivants:

- **`ID`** : l'ID du littéral (obligatoire)
- **`Type`** : le type de cet objet, y compris l'espace de noms et la classe (requis par les objets)
- **`ToolTip`** - **`ToolTip`** : donne un conseil
- Valeur par **`Default`** : une valeur par défaut de cet objet (obligatoire)
- **`Functions`**

Dans les extraits, il existe des littéraux prédéfinis. Ils sont énumérés ci-dessous:

Littéral	Détails
`$end$`	Marque l'emplacement où placer le curseur après l'insertion de l'extrait de code.

`$selected$`	Représente le texte sélectionné dans le document à insérer dans l'extrait de code lorsqu'il est appelé. Exemple, si vous avez:
	`A $selected$ is an object that I like.`
	et le mot était voiture sélectionnée lorsque vous avez invoqué le modèle, vous obtiendrez:
	`A car is an object that I like.`

2.2.2 Fonctions

Fonctions dans la balise Literal- ou Object signifie que vous pouvez utiliser une fonction pour

générer du code en fonction d'un autre élément. Il y a trois fonctions que je connais:

Fonction	La description	La langue
`GenerateSwitchCases (EnumerationLiteral)`	Génère une instruction switch et un ensemble d'instructions de cas pour les membres de l'énumération spécifiée par le paramètre EnumerationLiteral. Le paramètre EnumerationLiteral doit être une référence à un littéral d'énumération ou à un type d'énumération.	Visual C # et Visual J # [1]
`ClassName()`	Renvoie le nom de la classe qui contient l'extrait de code inséré.	Visual C # et Visual J # [1]
`SimpleTypeName(TypeName)`	Réduit le paramètre TypeName à sa forme la plus simple dans le contexte dans lequel l'extrait de code a été appelé.	Visual C #

[1] uniquement disponible dans Visual Studio 2005.

Table source: msdn.microsoft.com

Attributs pour les éléments littéraux et objets

Les balises Literal et Object peuvent avoir des attributs facultatifs.

Attribut	La description	Type
Éditable	Indique si vous pouvez ou non modifier le littéral après l'insertion de l'extrait de code. La valeur par défaut de cet attribut est true.	Booléen

Table source: msdn.microsoft.com

2.3 Références

Groupes d'éléments de référence contenant des informations sur les références d'assembly pour l'extrait de code. Cela peut contenir les éléments suivants:

- **Assembly:** contient le nom de l'assembly par l'extrait de code (requis)
- **URL:** contient un site Web qui fournit plus d'informations sur l'assembly

2.4 Code

Le code est le code que vous allez générer entre `<![CDATA[` et `]]>` . Placez l' `ID` de votre littéral

entre les caractères dollar et Visual Studio vous demandera de modifier ces valeurs par défaut si les déclarations sont remplies. Ici, vous avez un exemple pour C # et VB pour le raccourci.

```
<!-- ... Other code ... -->
<Declarations>
  <Literal>
    <Id>variablename</Id>
    <Default>_myproperty</Default>
  </Literal>

  <Literal>
    <Id>propertytype</Id>
    <Default>int</Default>
  </Literal>

  <Literal>
    <Id>propertyname</Id>
    <Default>myproperty</Default>
  </Literal>
</Declarations>

<Code Language="CSharp">
  <![CDATA[
    private $propertyvalue$ $variablename$;

    public $propertyvalue$ $propertyname$
    {
        get { return $variablename$; }
        set { $Variablename$ = Value; }
    }
  ]]>
</Code>

<!-- ... Other code ... -->

<Declarations>
  <Literal>
    <Id>variablename</Id>
    <Default>_myproperty</Default>
  </Literal>

  <Literal>
    <Id>propertytype</Id>
    <Default>int</Default>
  </Literal>

  <Literal>
    <Id>propertyname</Id>
    <Default>myproperty</Default>
  </Literal>
</Declarations>

<Code Language="VB">
  <![CDATA[
    Private $variablename$ As $propertyvalue$

    Public Property $propertyname$ As $propertyvalue$
        Get
            Return $variablename$
        End Get
```

```
        Set (ByVal value As $propertyvalue$)
            $variablename$ = value
        End Set
    End Property
  ]]>
</Code>

<!-- ... Other code ... -->
```

Dans l'attribut de langue requis, vous pouvez définir la langue dans laquelle vous créez l'extrait de code. Vous pouvez trouver les langues que vous pouvez utiliser dans le tableau suivant.

La langue	Mot-clé	Disponible dans les prochaines versions
Visual C #	CSharp	2005, 2010, 2012 et plus tard
Visual Basic	VB	2005, 2010, 2012 et plus tard
XML	XML	2005, 2010, 2012 et plus tard
Visual J #	VJSharp	2005, 2012 et plus tard
C ++	CPP	2012 et plus tard
JavaScript	JavaScript	2012 et plus tard
JScript	JScript	2012 et plus tard
SQL	SQL	2012 et plus tard
HTML	HTML	2012 et plus tard
CSS	CSS	2012 et plus tard
XAML	XAML	2012 et plus tard

Les autres attributs facultatifs sont les suivants:

Attribut	La description
Délimiteur	Spécifie le délimiteur utilisé pour décrire les littéraux et les objets dans le code. Par défaut, le délimiteur est `$` .
Gentil	Spécifie le type de code que contient l'extrait de code et, par conséquent, l'emplacement auquel un fragment de code doit être inséré pour que l'extrait de code soit compilé.

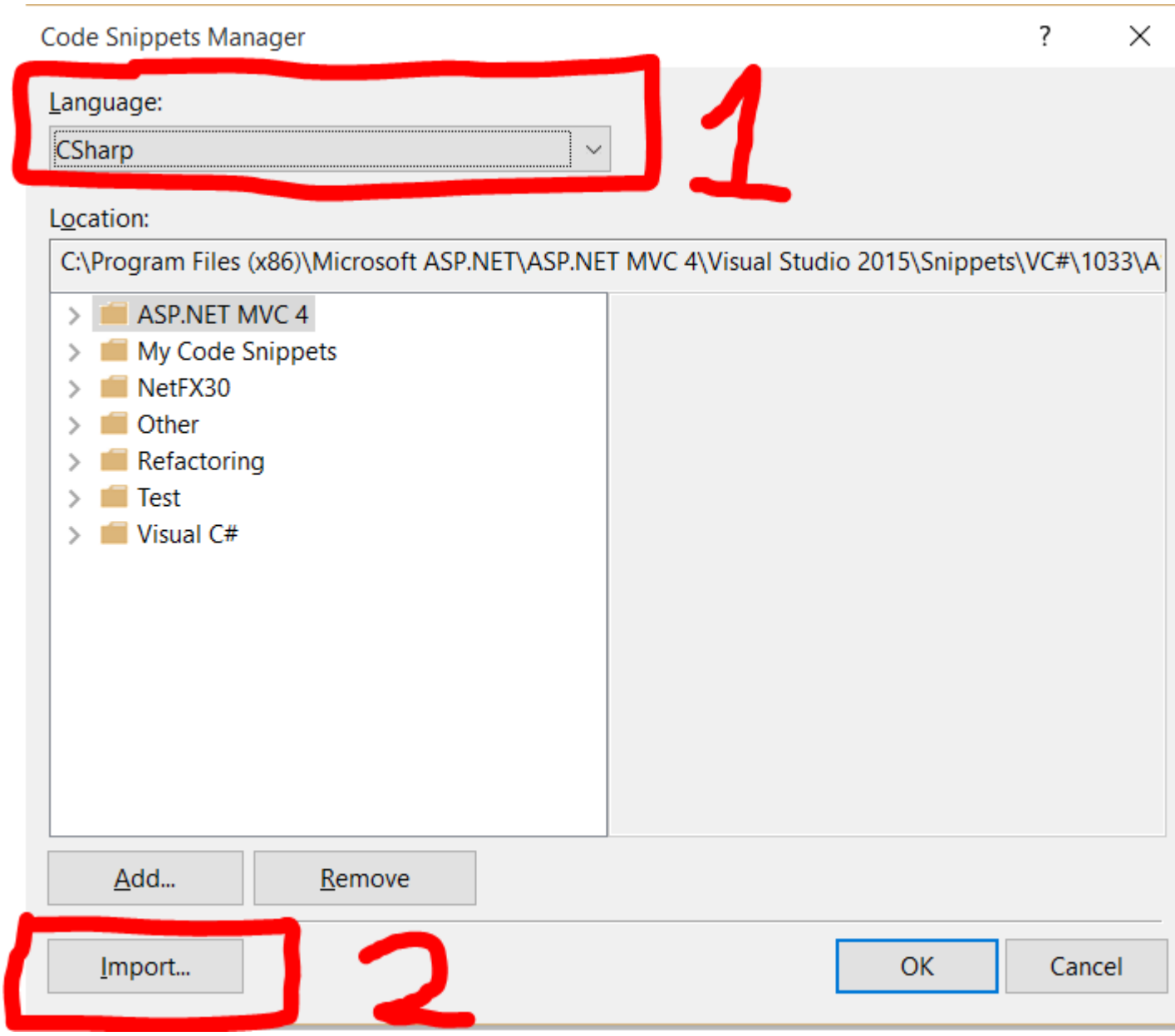
Les valeurs valides pour la variable kind sont les suivantes:

Valeur	La description
corps de méthode	Indique que l'extrait de code est un corps de méthode et doit donc être inséré dans une déclaration de méthode.
méthode décl	Indique que l'extrait de code est une méthode et doit donc être inséré dans une classe ou un module.
type décl	Indique que l'extrait de code est un type et doit donc être inséré dans une classe, un module ou un espace de noms.
fichier	Spécifie que l'extrait de code est un fichier de code complet. Ces extraits de code peuvent être insérés seuls dans un fichier de code ou dans un espace de noms.
tout	Spécifie que l'extrait de code peut être inséré n'importe où. Cette balise est utilisée pour les fragments de code indépendants du contexte, tels que les commentaires.

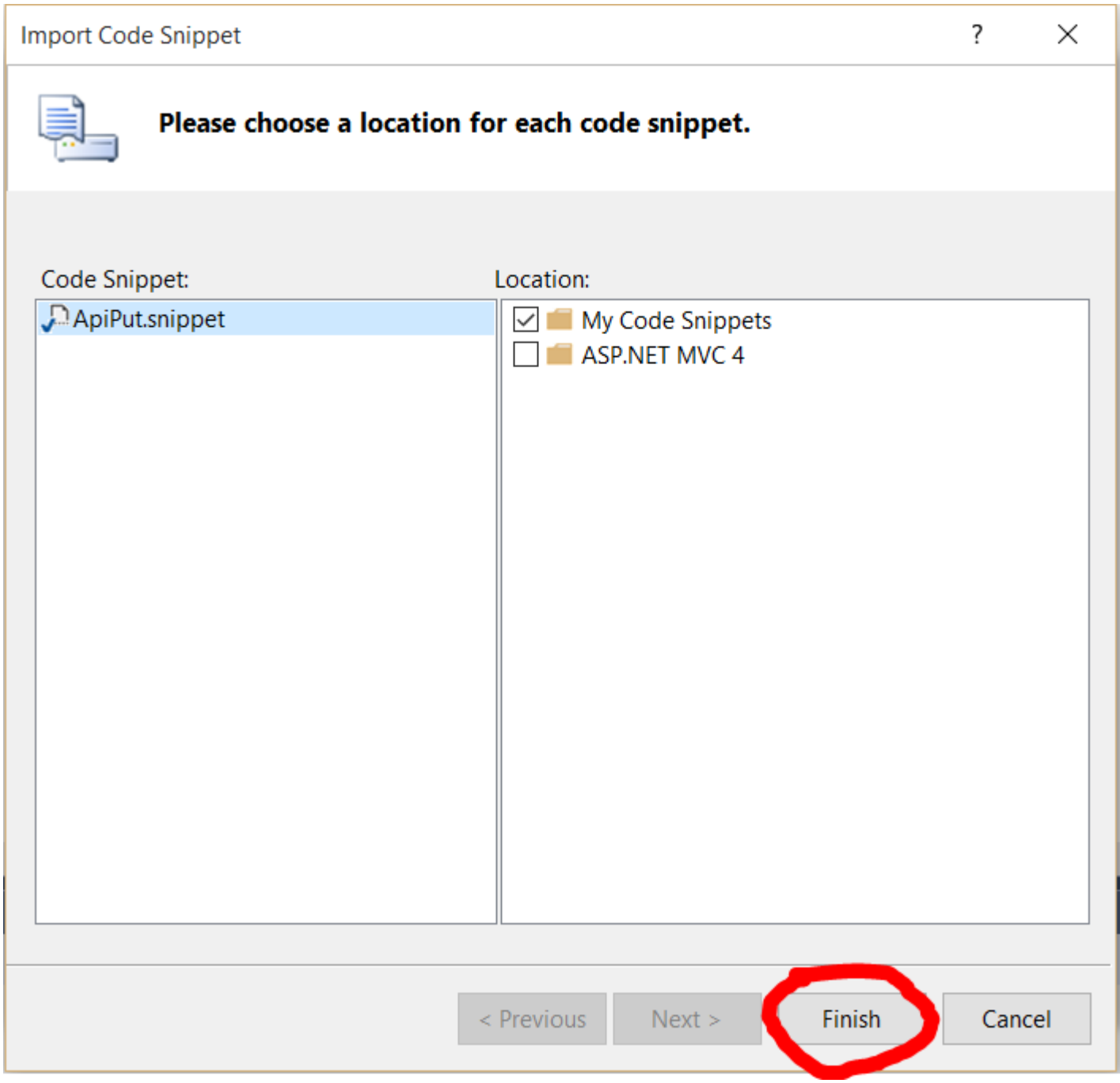
Tables source: msdn.microsoft.com

Importer l'extrait dans Visual Studio

1. Enregistrez le code XML et donnez-lui l'extension `.snippet` .
2. Vous pouvez ajouter le nouvel extrait dans Visual Studio en appuyant sur `Ctrl` + `K` , `Ctrl` + `B` ou aller dans *"Outils"* → *"Gestionnaire de fragments de code ..."* . Cette prochaine fenêtre ouverte:



3. Choisissez la langue dans la liste déroulante pour laquelle vous avez créé le fragment. Cliquez sur *"Importer ..."* et choisissez le fichier que vous avez créé.



4. Cliquez sur `"Finish"` . Si le nom de fichier a déjà été utilisé, Visual Studio va demander à remplacer le fichier existant. Vous avez trois options:

    - **Écraser:** écrase le fichier. Vous pouvez utiliser cette option si vous modifiez un ancien extrait de code.
    - **Renommer: permet** de renommer le fichier en un nom unique.
    - **Ignorer:** annule l'importation. Renomme le fichier en un nom unique.

Vous pouvez également ajouter un nouvel emplacement avec tous les extraits que vous avez créés en cliquant sur le bouton *"Ajouter ..."* de la première fenêtre et en sélectionnant le dossier dans la *fenêtre "Sélectionner un dossier"* . L'avantage est que lorsqu'un nouveau fragment de code valide est ajouté dans ce dossier, vous pouvez l'utiliser directement dans Visual Studio.

Remarque: testez après avoir importé votre extrait de code pour détecter les erreurs afin de ne

pas avoir de problème lorsque vous utilisez l'extrait de code. Vous pouvez toujours supprimer ou écraser l'extrait de code en cas d'erreur.

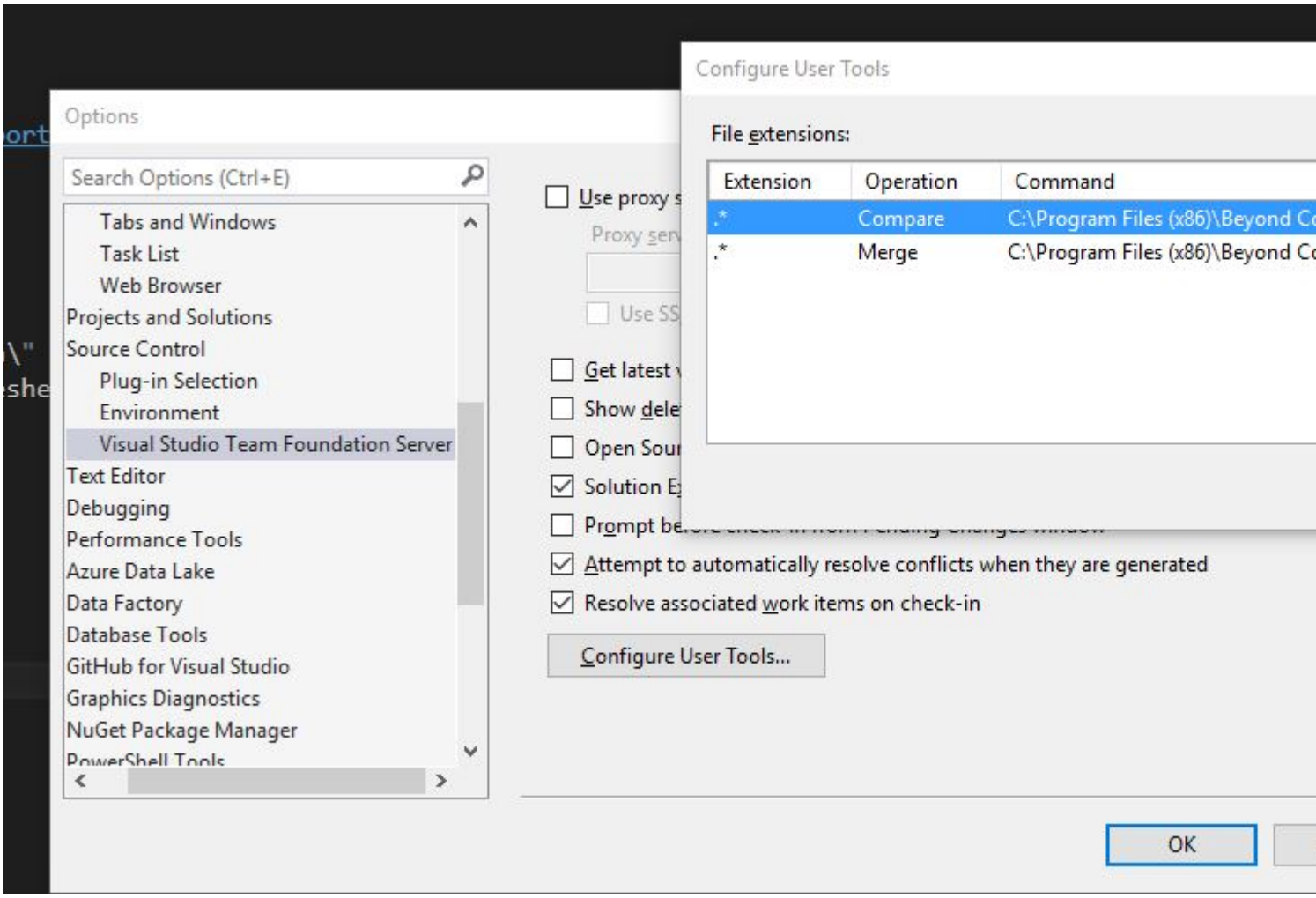
Point d'Intrest

Vous pouvez également consulter la documentation sur MSDN pour plus d'informations.

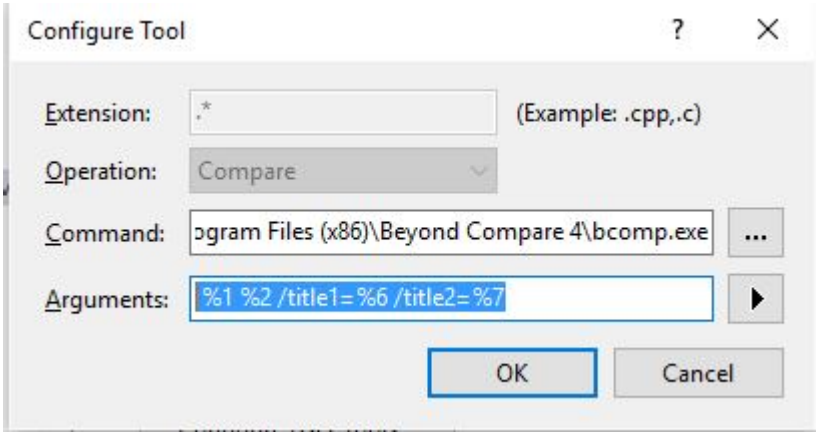
Remplacer les outils de fusion / comparaison

Got to Tools | Options | Contrôle de source | Visual Studio Team Foundation Server

cliquez sur Configurer les outils utilisateur:



Vous pouvez ajouter des remplacements séparés pour les opérations de comparaison et de fusion. Cliquez sur Ajouter et sélectionnez l'opération que vous souhaitez remplacer. Vous devez saisir le chemin d'accès à l'outil que vous utilisez et les arguments exacts attendus par votre outil. Par exemple, pour utiliser BeyondCompare, ajoutez les arguments suivants "% 1% 2 / title1 =% 6 / title2 =% 7":



Pour fusionner avec BeyondCompare, utilisez les arguments "% 1% 2% 3% 4 / title1 =% 6 / title2 =% 7 / title3 =% 8 / title4 =% 9"

Dans un article publié en 2006 sur un blog , James Manning, employé de MS, a analysé les arguments comme prévu avec divers outils: WinDiff, DiffDoc, WinMerge, Beyond Compare, KDiff3, Araxis, Compare It !, SourceGear DiffMerge, TortoiseMerge et Visual SlickEdit. La publication est un bon point de départ, mais assurez-vous de vérifier la documentation à jour de votre outil.

Il est fortement recommandé de *ne* pas utiliser pour les outils de fusion incapables de fusionner à trois (par exemple, WinMerge 2.x).

Cadre d'entité

> Entity Framework (EF) est un mappeur objet-relationnel qui permet aux développeurs .NET de travailler avec des données relationnelles à l'aide d'objets spécifiques à un domaine. Cela élimine le besoin de la plupart du code d'accès aux données que les développeurs doivent généralement écrire.
>
> Entity Framework vous permet de créer un modèle en écrivant du code ou en utilisant des boîtes et des lignes dans EF Designer. Ces deux approches peuvent être utilisées pour cibler une base de données existante ou créer une nouvelle base de données.

Source et plus d'informations: Documentation Entity Framework

Lire Outils Visual Studio en ligne: https://riptutorial.com/fr/visual-studio/topic/2398/outils-visual-studio

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec Visual Studio	Almir Vuk, Avi Turner, Cody Gray, Community, Derpcode, Fruchtzwerg, H. Pauwelyn, meJustAndrew, Misaz, Nikita, Richard Banks
2	Ajouter une extension	Avi Turner
3	Connecter votre projet de studio visuel à Github	bigworld12
4	Contrats de code	Disk Crasher
5	Outils Visual Studio	dove, H. Pauwelyn, Ofek Shilon