# TUTORIEL PROFILAGE C#

## Introduction :

Dans ce tutoriel, nous allons effectuer des tests de performance C# avec Visual Studio 2012. Ainsi nous allons pouvoir constater les temps d’exécution machine, le pourcentage d’unité centrale (UC) utilisé par chaque méthode à l’intérieur même du programme principal.

## Pré-requis :

Vous allez récupérer les sources et lancer la solution Visual Studio qui contient déjà les trois projets de méthodes de Tri.

Chaque projet implémente une méthode de tri différente :

* Tri à bulles
* Tri par sélection
* Tri par insertions

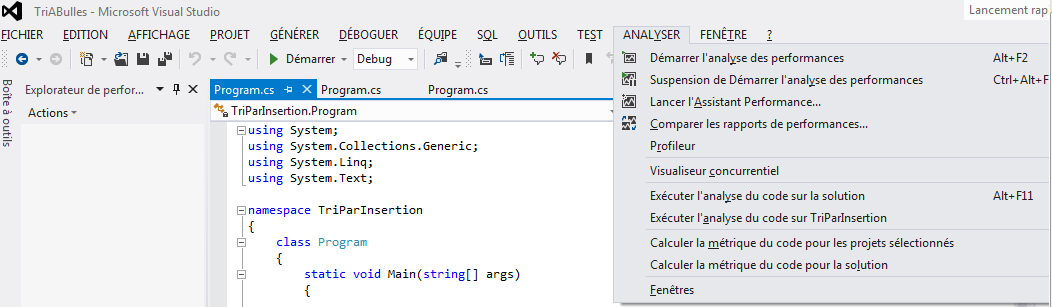
## Avant le profilage :

Si vous le souhaitez, vous pouvez tester les projets individuellement pour observer le comportement du code. Pour cela, il suffit de dé commenter les lignes  en fin de méthode main() de chaque fichier program.cs. Ensuite vous pouvez les lancer/debugger.

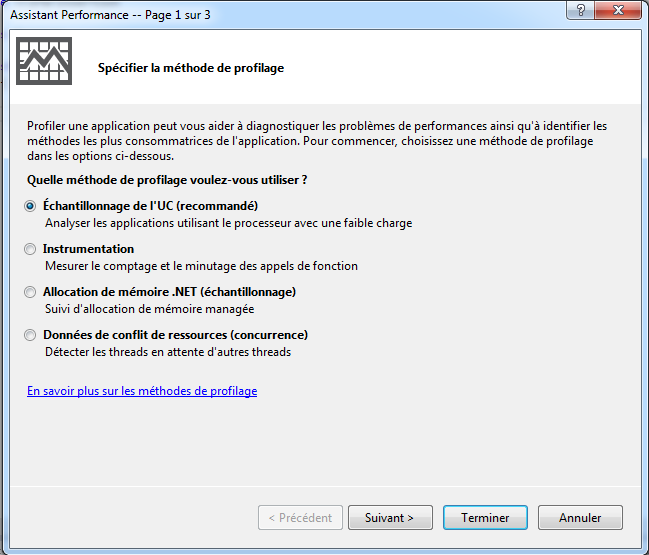
## Le profilage :

*Passons aux choses sérieuses…*

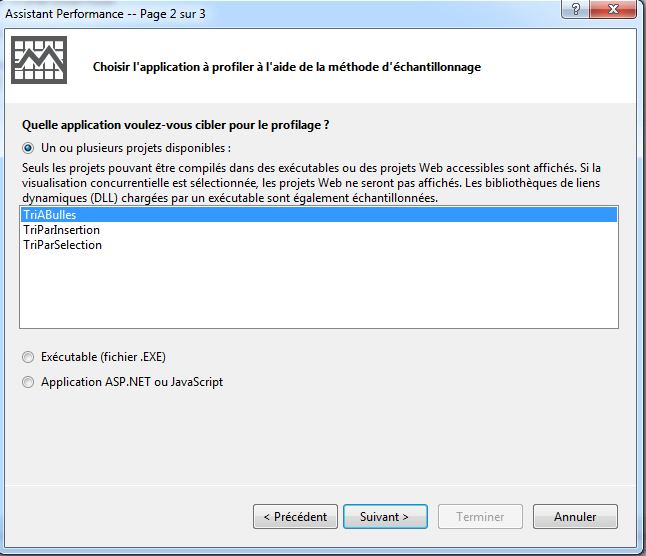
Pour lancer une session de profilage de performance Visual Studio, cliquer sur le menu Analyser, puis Lancer l’Assistant Performance…



Ensuite, nous allons choisir la méthode de profilage. La méthode la plus intéressante et la seule que nous allons lancer est l’Echantillonnage de l’UC.



Cliquer sur Suivant pour choisir le projet à tester. Choisir UN seul des projets car nous allons comparer chaque rapport individuellement.

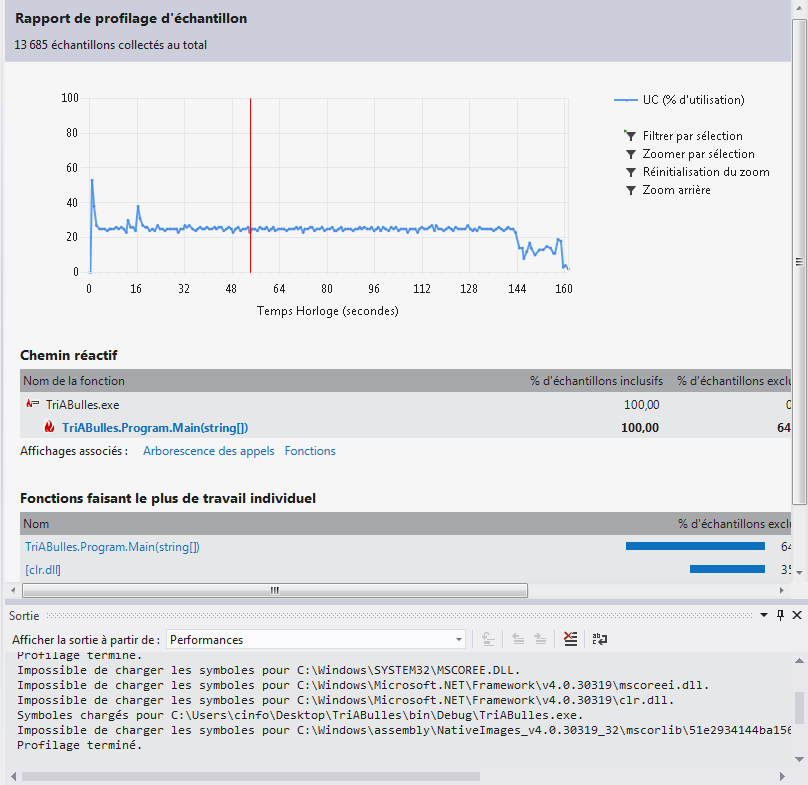


Cliquer sur Suivant, laisser cocher le démarrage du profilage et cliquer sur Terminer.

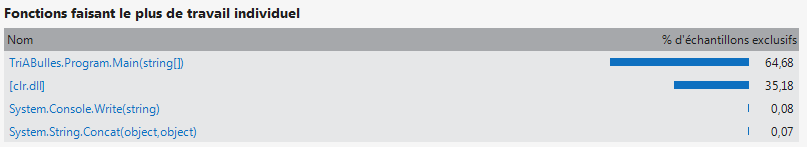
Ensuite, la session de performance va se lancer. Attendre la fin de l’exécution de la tâche de profilage.

Nous pouvons désormais analyser le rapport généré par Visual Studio.

* On peut observer un graphique de l’évolution du pourcentage d’utilisation de l’UC en fonction du temps Horloge exprimé en secondes
* En dessous, on peut voir les fonctions les plus coûteuses en utilisation UC avec le pourcentage des échantillons. Echantillons inclusifs correspond au travail effectué par la fonction ainsi que toutes celles appelées par celle-ci. Echantillons exclusifs indiquent la quantité de travail effectué par le code dans le corps de la fonction à l’exception des fonctions appelées dans ce code.
* Le dernier tableau présente l’échantillonnage exclusif des fonctions. C’est-à-dire le pourcentage du code présent dans cette fonction.



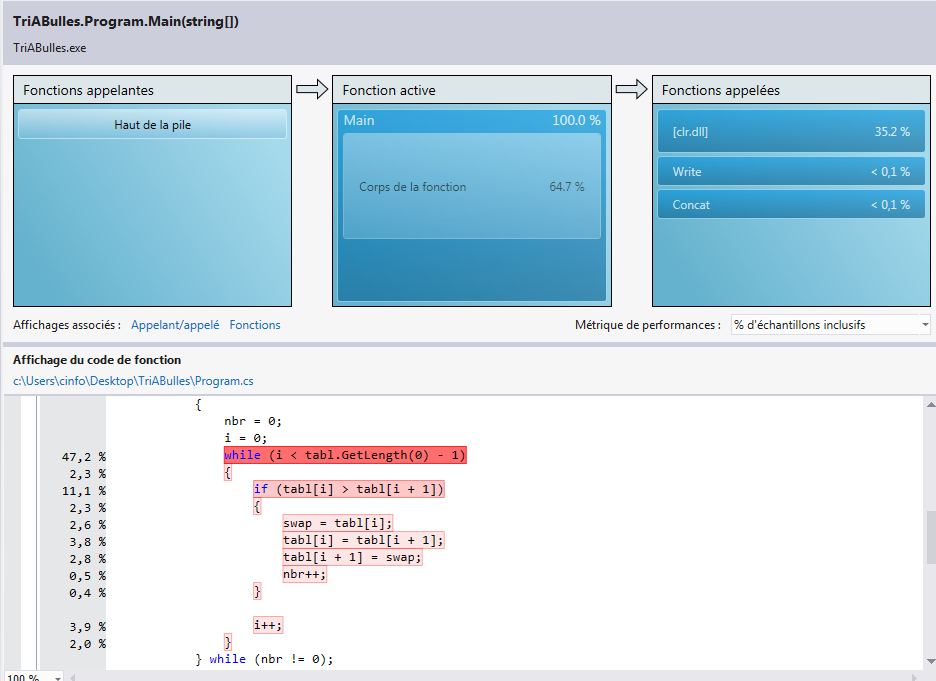
Ce tableau est intéressant que on peut voir quelles fonctions ont été appelées et exécutées dans le code du programme. Ainsi que le pourcentage du travail effectué par cette fonction par rapport au travail total d’exécution.



Si on clique sur une des fonctions (la méthode Main() ici), on a le détail du travail effectué par celle-ci :

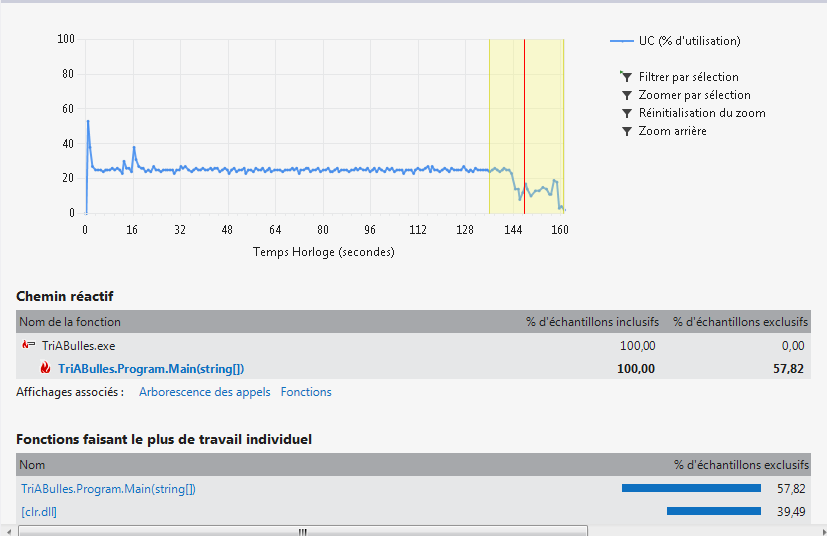
* Fonction appelante
* Travail du corps de la fonction
* Fonctions appelées

En dessous, on trouve aussi une analyse détaillée de la charge de travail sur chaque portion de code. Ici, on peut constater que le parcours du tableau contenant les valeurs à trier est la portion de code qui constitue la charge de travail la plus importante pour l’UC.



Si vous revenez à l’écran précédent, avec la petite flèche en haut à gauche, vous pouvez obtenir l’analyse détaillée du travail de l’UC sur une partie du travail, sur le premier graphique. Pour cela, sélectionner la portion du graphique en cliquant sur 2 points de celui-ci et cliquer sur filtrer par sélection.

Vous pouvez répéter l’opération pour comparer le travail de l’UC sur 2 portions différentes (par exemple : début et fin de l’exécution du programme).



Vous venez donc de réaliser une analyse de performance avec l’outil de profilage Visual Studio.

## Etape suivante

Répéter cette opération pour chaque projet, afin d’obtenir un rapport de performance par méthode de tri et de pouvoir les comparer.

Il suffit pour cela de choisir un projet différent du premier lors du lancement de l’assistant de l’outil de performance.

## Comparaison

Vous venez donc de générer 3 rapports de performances correspondant au travail de l’UC, nous allons donc pouvoir comparer ces rapports. La comparaison va permettre de déterminer quelle méthode de tri est la méthode optimale pour trier un grand tableau de données (ici des entiers).

L’outil de comparaison fournit par Visual Studio n’est pas très intéressant car il fournit uniquement un différentiel des chiffres d’échantillonnage entre 2 rapports, sans apporter de détails.

Pour comparer donc 2 rapports, il suffit d’ouvrir les rapports et de constater la différence du temps horloge, du travail de l’UC et du détail de ce travail pour chaque fonction. Vous constaterez des différences sur les chiffres…

Alors, quelle méthode est la plus optimale ?