**Se familiariser avec les fonctionnalités du débogueur intégré de Visual Studio Community 2022**

-Pour comprendre vos erreurs, Visual Studio vous permet d’utiliser le débogueur. Il permet de voir de plus près comment fonctionne une variable au cas vous aurez aussi la possibilité qu’uneerreur ne soit pas déclenchée mais le résultat attendu n’est pas le bon (erreur logique).

Vous allez ainsi tracer ou surveiller votre code.

**I. Démarrer le débogueur et atteindre des points d’arrêt**

-Créer un nouveau projet Tutoriel9DebogueurVisualStudio

-Dans *Program.cs*, remplacez tout le code par défaut par le code suivant :

**using System;**

**class ArrayExample**

**{**

**static void Main()**

**{**

**char[] letters = { 'f', 'r', 'e', 'd', ' ', 's', 'm', 'i', 't', 'h' };**

**string name = "";**

**int[] a = new int[10];**

**for (int i = 0; i < letters.Length; i++)**

**{**

**name += letters[i];**

**a[i] = i + 1;**

**SendMessage(name, a[i]);**

**}**

**Console.ReadKey();**

**}**

**static void SendMessage(string name, int msg)**

**{**

**Console.WriteLine("Hello, " + name + "! Count to " + msg);**

**}**

**}**

-Démarrer le débogueur :**Pour démarrer le débogueur**, sélectionnez F5 (pour Windows) ou choisissez le bouton Cible de débogage dans la barre d’outils Standard, ou choisissez le bouton Démarrer le débogage dans la barre d’outils Déboguer, ou choisissez Déboguer>Démarrer le débogage dans la barre de menus. L’application s’exécute jusqu’à l’achèvement et vous voyez la sortie de la console :Une image contenant texte

Description générée automatiquement

- **Pour arrêter le débogueur**, sélectionnez Maj+F5, ou choisissez le bouton Arrêter le débogage dans la barre d’outils Déboguer, ou choisissez Déboguer>Arrêter le débogage dans la barre de menus.

**-Définir un point d’arrêt et démarrer le débogueur :**

Vous pouvez définir des points d’arrêt là où vous voulez que Visual Studio interrompe l’exécution de votre code afin d’examiner les valeurs des variables ou le comportement de la mémoire ou encore la bonne exécution ou non d’une partie du code.

1. Dans la boucle for de la fonction Main, définissez un point d’arrêt en cliquant dans la marge gauche de la ligne de code suivante :

name += letters[i];

Un cercle rouge apparaît là où vous avez défini le point d’arrêt.



1. Pour commencer le débogage, sélectionnez **F5** ou choisissez le bouton Cible de débogage dans la barre d’outils Standard, ou choisissez **le bouton Démarrer le débogage dans la barre d’outils Déboguer, ou choisissez Déboguer>Démarrer le débogage dans la barre de menus.** L’application démarre et le débogueur s’exécute jusqu’à la ligne de code où vous avez défini le point d’arrêt.



La flèche jaune pointe vers l’instruction sur laquelle le débogueur s’est arrêté. L’exécution de l’application est suspendue au même point, l’instruction n’étant pas encore exécutée.

Lorsque l’application n’est pas en cours d’exécution, **F5** démarre le débogueur, qui l’exécute jusqu’à ce qu’elle atteigne le premier point d’arrêt. Si l’application est suspendue à un point d’arrêt, **F5** continue d’exécuter l’application jusqu’à ce qu’elle atteigne le point d’arrêt suivant.

Les points d’arrêt sont une fonctionnalité pratique quand vous savez quelle ligne ou section de code vous voulez examiner en détail.

**II. Naviguer dans le code et inspecter les données à l’aide de conseils sur les données**

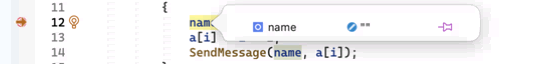
1-Pendant la pause sur l’instruction name += letters[i], pointez sur la variable letters pour afficher une info-bulle de données montrant la taille du tableau et le type d’élément, char[10].

2-Développez la variable letters pour afficher tous ses éléments de tableau et leurs valeurs :

Une image contenant table

Description générée automatiquement

3- Pointez sur la variable name pour afficher sa valeur actuelle, qui est une chaîne vide.



4-Pour faire avancer le débogueur jusqu’à l’instruction suivante, sélectionnez **F10**, ou **cliquez sur le bouton Pas à pas principal dans la barre d’outils Déboguer**, ou choisissez **Déboguer>Pas à pas principal dans la barre de menus. Sélectionnez F10 deux fois de plus pour passer l’appel de méthode SendMessage.**

5- Pour itérer dans la boucle for plusieurs fois, sélectionnez **F10** à plusieurs reprises. Pendant chaque itération de boucle, effectuez une pause au point d’arrêt, puis pointez sur la variable name pour vérifier sa valeur dans l’info-bulle de données.

Une image contenant texte

Description générée automatiquement

La valeur de la variable change à chaque itération de la boucle for, en affichant les valeurs de f, puis fr, puis fre, et ainsi de suite. Pour faire avancer le débogueur dans la boucle plus rapidement, sélectionnez plutôt **F5**, qui progresse jusqu’à votre point d’arrêt au lieu de l’instruction suivante.

6- Lorsque la boucle for de la méthode Main est suspendue, sélectionnez F11, ou choisissez le bouton effectuer un pas à pas détaillé dans la barre d’outils Déboguer, ou choisissez Déboguer>effectuer un pas à pas détaillé dans la barre de menus, jusqu’à ce que vous atteigniez l’appel de méthode SendMessage.

Le débogueur doit être suspendu à cette ligne de code :

SendMessage(name, a[i]);

**II. Parcourir le code avec exécuter jusqu’au clic :**

1-Sélectionnez F5 pour passer à nouveau au point d’arrêt.

2-Dans l’éditeur de code, placez le curseur sur l’appel de méthode Console.WriteLine dans la méthode SendMessage.

Clic droit –executer jusqu au clic (run to cursor)

Le débogueur avance jusqu’à l’appel de méthode Console.WriteLine.

Resultat à la console : Hello, f! Count to 1

L’utilisation du bouton Exécuter jusqu’au clic est similaire à la définition d’un point d’arrêt temporaire. Cela permet de se déplacer rapidement dans une zone visible du code de votre application dans un fichier ouvert.

**-Redémarrer rapidement votre application :**

Pour réexécuter votre application depuis le début dans le débogueur, sélectionnez **Ctrl+Maj+F5**, ou choisissez le bouton **Redémarrer dans la barre d’outils Déboguer**, ou choisissez **Déboguer>Redémarrer** dans la barre de menus.



La commande **Redémarrer arrête le débogueur, puis le redémarre en une seule étape**. Lorsque le débogueur redémarre, il s’exécute jusqu’au premier point d’arrêt, qui est le point d’arrêt que vous avez précédemment défini dans la boucle for, puis s’interrompt.

- **Actions de points d’arrêt et points de trace**

Un *point de trace* est un point d’arrêt qui affiche un message dans la fenêtre **Sortie**. Un point de trace peut agir comme une instruction de trace temporaire dans le langage de programmation et ne met pas en pause l’exécution du code. Vous créez un point de trace en définissant une action spéciale dans la fenêtre **Paramètres du point d’arrêt**.

- **Conditions de point d’arrêt**

Vous pouvez contrôler quand et où un point d’arrêt s’exécute en définissant des conditions. La condition peut être n’importe quelle expression valide reconnue par le débogueur.

**IV. Inspecter des variables dans des bulles d’informations et dans les fenêtres du débogueur**

Les fenêtres Autos et Locals affichent les valeurs des variables lorsque vous déboguez. Les fenêtres ne sont disponibles que pendant une session de débogage. La fenêtre Autos affiche les variables utilisées sur la ligne active à laquelle se trouve le débogueur et sur la ligne précédente. La fenêtre Locals affiche les variables définies dans l’étendue locale, qui est généralement la fonction ou la méthode actuelle.

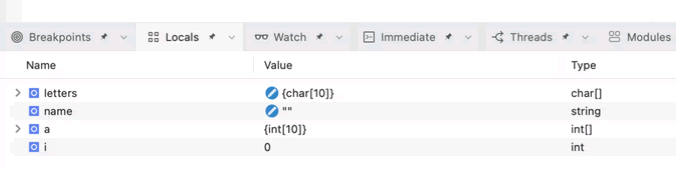
1-Pendant que le débogueur est suspendu, consultez la fenêtre Autos en bas de l’éditeur de code.

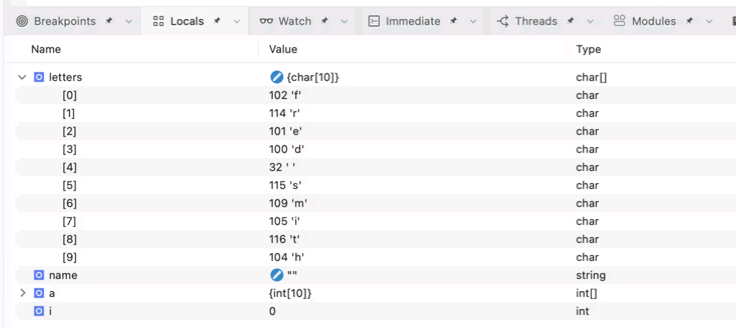
Si la fenêtre Autos est fermée, sélectionnez Ctrl+D, A ou choisissez Déboguer>Fenêtres>Autos dans la barre de menus.

2-Pendant que le débogueur est toujours suspendu, consultez la fenêtre Locals, dans un onglet à côté de la fenêtre Autos.

Si la fenêtre Locals est fermée, sélectionnez Ctrl+D, L ou choisissez Déboguer>Fenêtres>Locals.

3-Dans la fenêtre Locals, développez la variable letters pour voir ses éléments de tableau et leurs valeurs.



**-** Pour les détails de la variable letters : ****

- Pour modifier les valeurs de la plupart des variables dans les fenêtres **Autos** ou **Locals**, double-cliquez sur la valeur et entrez la nouvelle valeur.

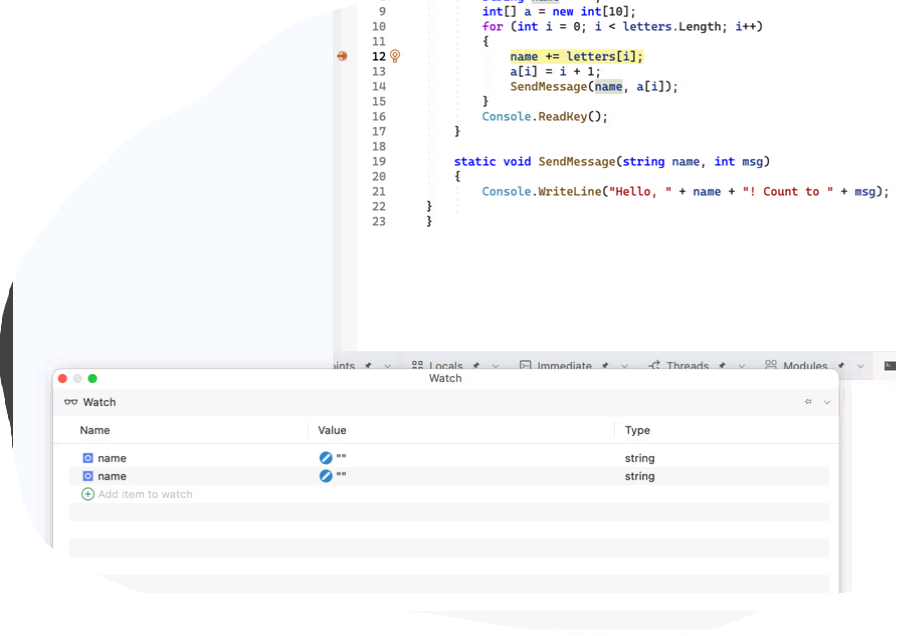
Vous pouvez entrer une expression pour une valeur, par exemple a + b. Le débogueur accepte la plupart des expressions de langage valides.

**V. Définir un espion (Watch) :**

Vous pouvez spécifier une variable, ou une expression, sur laquelle vous souhaitez garder un œil lorsque vous parcourez le code, en l’ajoutant à la fenêtre Espion.

1. Pendant que le débogueur est suspendu, cliquez avec le bouton droit sur la variable name et choisissez **Ajouter un espion**.

La fenêtre **Espion** s’ouvre par défaut en bas de l’éditeur de code.



1. Maintenant que vous avez défini un espion sur la variable name, parcourez votre code pour voir la valeur de la variable name changer à chaque itération de la boucle for.

Contrairement à d’autres fenêtres de variables, la fenêtre **Espion** montre toujours les variables que vous observez, mais elles apparaissent en grisé quand elles sont en dehors de l’étendue.

* 1. **Examiner la pile des appels (Call Stack)**

La pile des appels peut vous aider à comprendre le flux d’exécution de votre application, en montrant l’ordre dans lequel les méthodes et les fonctions sont appelées.

1-Pendant que le débogueur est suspendu dans la boucle for, consultez la fenêtre Pile des appels, qui s’ouvre par défaut dans le volet inférieur droit de l’éditeur de code.

Si la fenêtre Pile des appels est fermée, sélectionnez Ctrl+D, C ou choisissez Déboguer>Fenêtres>Pile des appels dans la barre de menus.

Dans la fenêtre Pile des appels, vous verrez le pointeur jaune au niveau de la méthode Main active.

2-Sélectionnez F11 plusieurs fois, jusqu’à ce que le débogueur place l’application en pause dans la méthode SendMessage.

La ligne supérieure de la fenêtre Pile des appels affiche la fonction active, qui est la méthode SendMessage. La deuxième ligne montre que la méthode SendMessage a été appelée à partir de la méthode Main.

Références :

-Tutoriel : Apprendre à déboguer le code C# avec Visual Studio : <https://learn.microsoft.com/fr-ca/visualstudio/get-started/csharp/tutorial-debugger?view=vs-2022>

-Inspectez les variables dans les fenêtres Autos et Locals: <https://learn.microsoft.com/fr-fr/visualstudio/debugger/autos-and-locals-windows?view=vs-2022>

- Utiliser les points d’arrêt dans le débogueur Visual Studio : <https://learn.microsoft.com/fr-fr/visualstudio/debugger/using-breakpoints?view=vs-2022>