

**Prise en main de Visual Studio 2022**

**Les professeurs d’informatique**

*Informatique de gestion*

**HENALLUX – Catégorie économique – 2022-2018**

### Qu’est-ce que Visual Studio ?

Visual Studio est un ensemble complet d'outils de développement permettant de construire divers types d’applications complexes (applications Web ASP.NET, web services XML, applications Windows, applications liées à des bases de données, applications mobiles…). C’est un environnement de développement intégré qui repose sur des bibliothèques communes accessibles via divers langages (comme C#, Visual C++, F# ou encore Visual Basic).

***Bon à savoir…***Un **EDI** (Environnement de Développement Intégré) ou **IDE** (Integrated Development Environment) est une application qui intègre divers outils facilitant le développement de programmes (généralement, un éditeur de texte, un compilateur et un débuggeur).

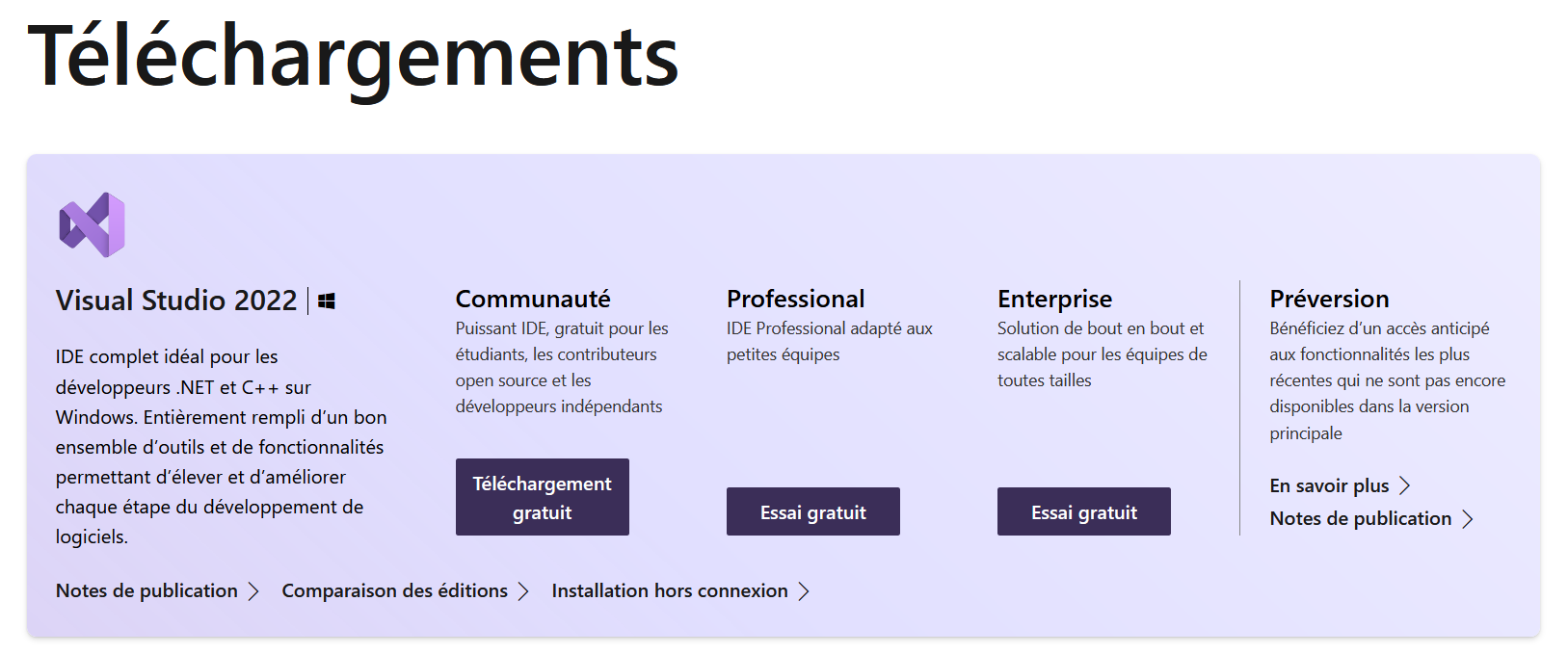
### Pour travailler chez soi

Plusieurs solutions s’offrent à vous :

* Vous avez normalement déjà installé Visual Studio pour accéder à la ligne de commande lors du cours de « Langage de programmation : bases ».
* Visual Studio se décline en plusieurs versions (Professional, Enterprise…), dont une version gratuite appelée Visual Studio Community[[1]](#footnote-1). Cette version convient parfaitement aux besoins des cours de la section.

### Installation de Visual Studio 2022

La première étape est de télécharger la version de Visual Studio que vous désirez :

****

Ensuite, exécuter l’installeur et après avoir accepté les termes du contrat de licence Microsoft, le logiciel ouvre la fenêtre d’installation.

Une image contenant texte

Description générée automatiquement

### Installation de Visual Studio Community 2022

La fenêtre d’installation est affichée et Visual Studio vous demande de configurer votre environnement de travail en choisissant les éléments à installer.

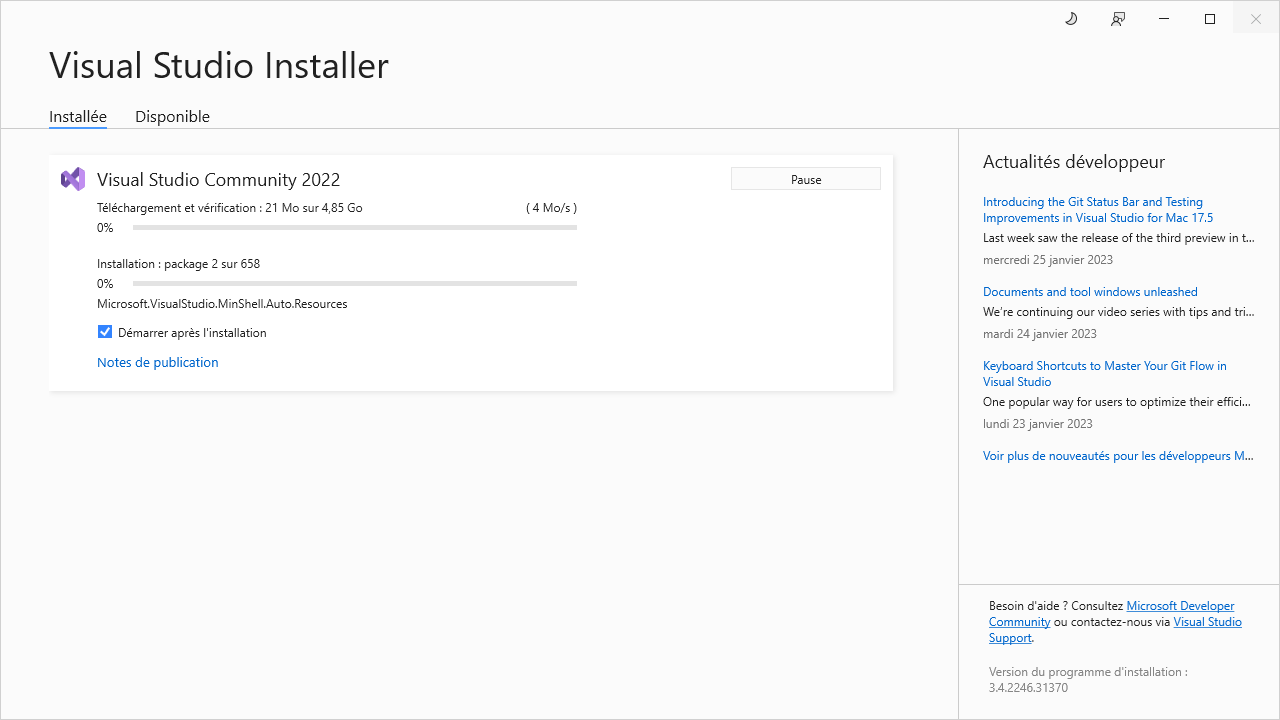
Vous devez choisir les « **Charges de travail** » qui sont nécessaires à la compilation de vos futurs projets en C. Scrollez un peu dans cet onglet afin de choisir les deux charges mises en évidence ci-dessous.

Une image contenant texte

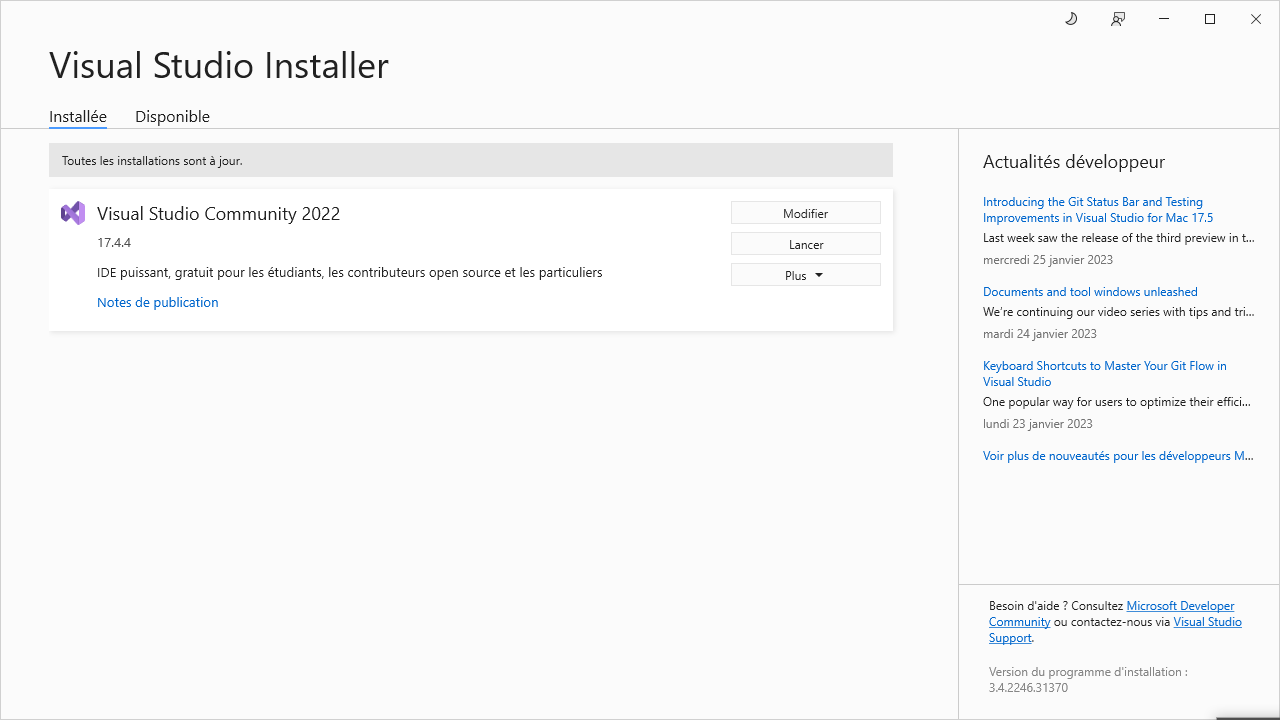
Description générée automatiquement

La langue d'installation est, par défaut, celle du système installé. Il est possible d'en changer via l’onglet « **Modules linguistiques** » ou de le faire par la suite.

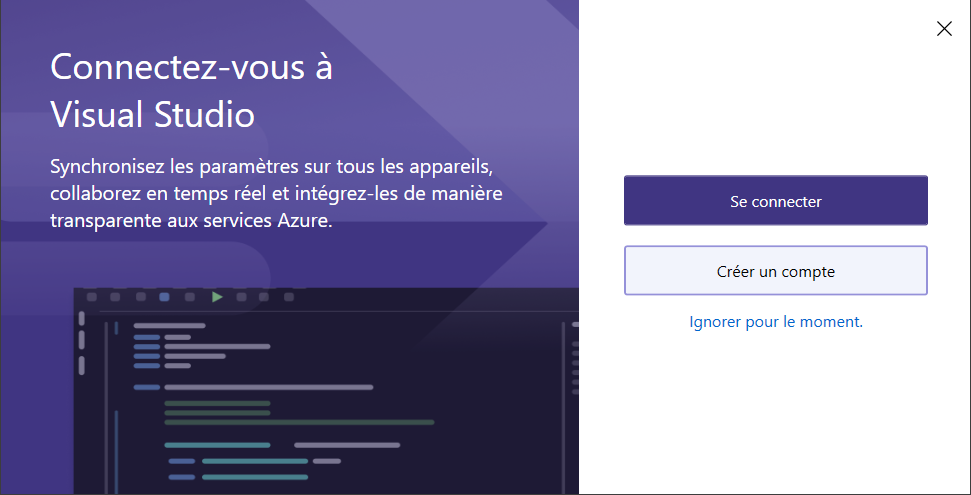
Cliquez sur le bouton « **Installer** » et les différents éléments demandés sont installés.



Lorsque l’installation est terminée, la fenêtre devient… Il vous suffit de lancer !



Lors de la première exécution, il faut vous connecter à votre compte Microsoft (login et mot de passe de l’HENALLUX).



Voici l'écran d'accueil de Visual Studio :

Une image contenant texte

Description générée automatiquement

### Création d’un projet Visual Studio

***Solutions, projets, fichiers…***Une application peut nécessiter plusieurs fichiers (des programmes, des pages web, des feuilles de style CSS…). Sous Visual Studio, les fichiers sont rassemblés en **projets**.  
De même, on peut avoir envie de rassembler plusieurs projets apparentés (par exemple, la gestion des ventes en ligne, la gestion des stocks et la gestion du fichier client). On parle alors de **solution**.

Avant de pouvoir se mettre à programmer en Visual Studio, il faut créer un projet qui va rassembler tous les fichiers de code. Vos premiers travaux ne comporteront sans doute qu’un seul fichier mais, plus tard, vous pourriez être amenés à morceler votre travail en plusieurs fichiers. Ce projet sera automatiquement placé dans une solution, à laquelle vous pourrez éventuellement ajouter d’autres projets par la suite.

Voici la marche à suivre pour créer un projet Visual Studio.

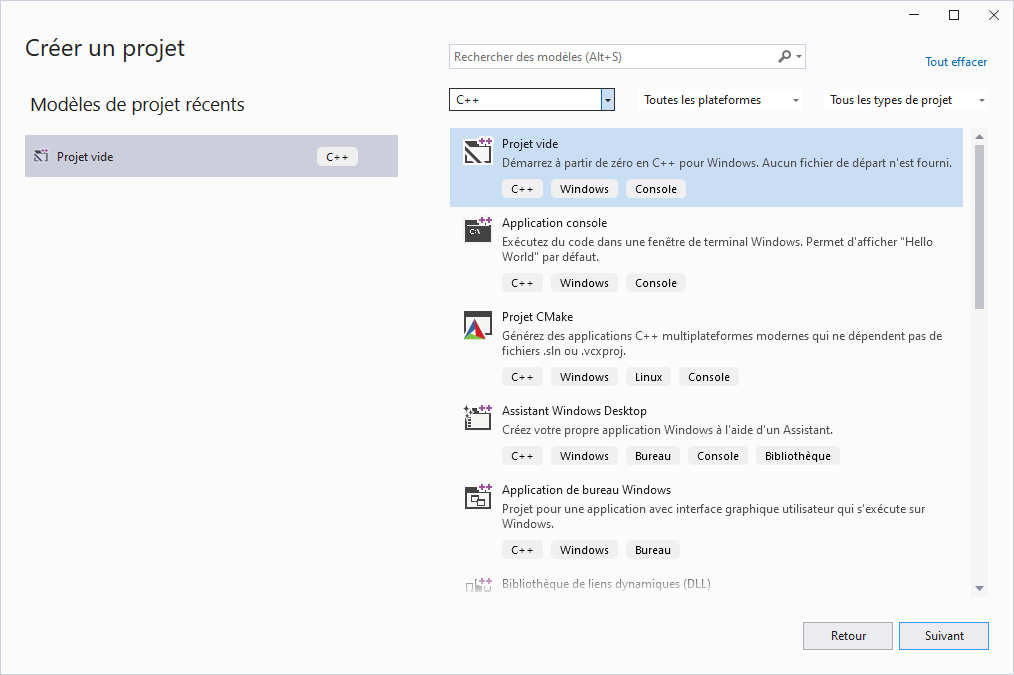
1. *Cliquez sur Créer un projet*

Une image contenant texte

Description générée automatiquement

*Remarque : il est possible de continuer sans code.*

1. *Sélectionnez «* ***C++*** *» comme langage dans l’onglet prévu à cet effet. Choisissez «* ***Projet vide*** *» comme type de projet et cliquez sur «* ***Suivant*** *».*



1. *Choisissez un nom pour le projet, un emplacement (par exemple U:\LabosC\Serie1) où sauvegarder les fichiers correspondants et un nom pour la solution.*

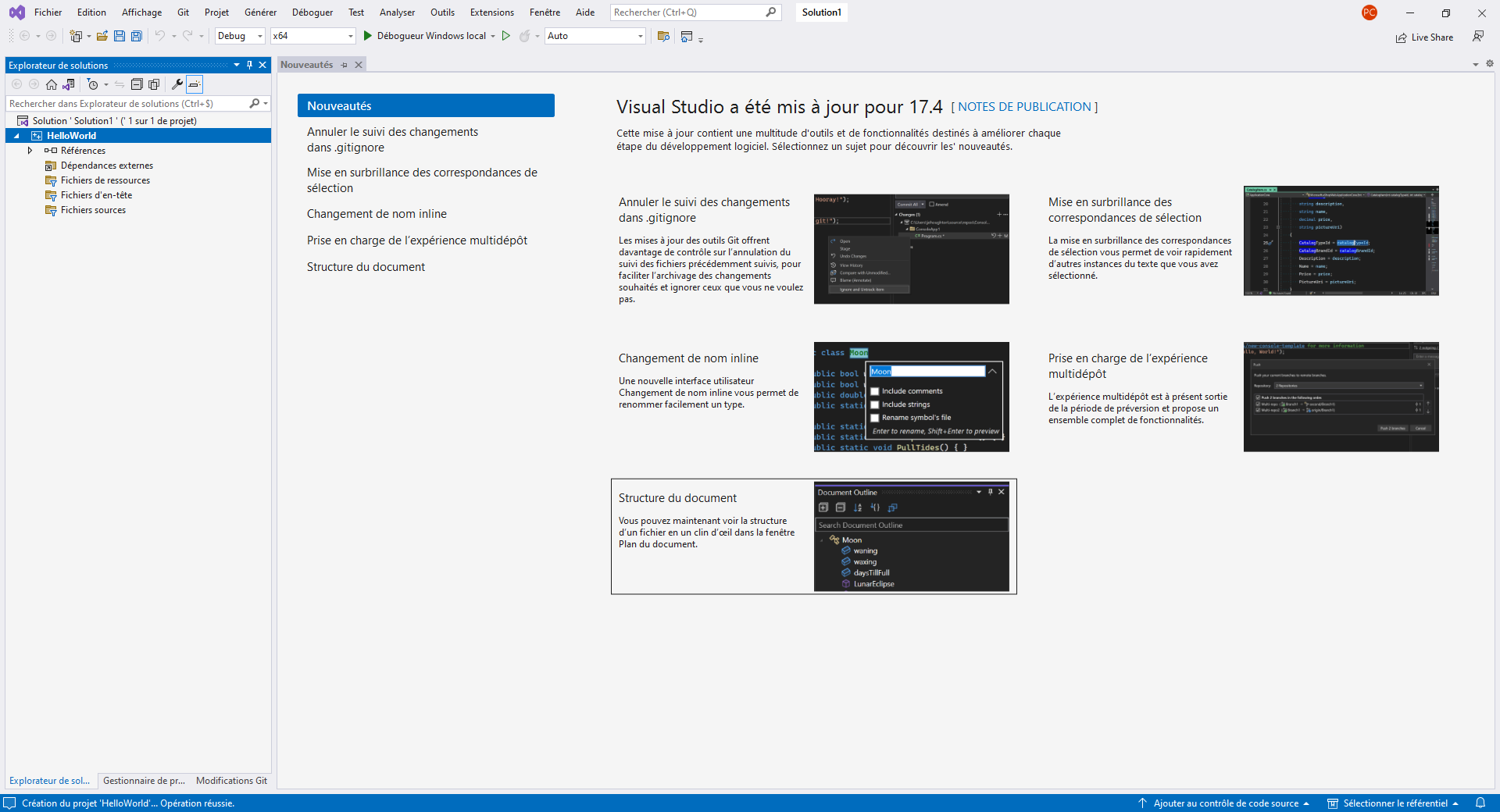
Une image contenant texte

Description générée automatiquement

*Cela fait, vous pouvez cliquer sur «* ***Créer*** *».*

***Travail dans les coulisses…***Visual Studio crée un répertoire (du nom de votre projet) à l’emplacement indiqué. C’est là qu’on place non seulement le code que vous allez écrire mais aussi sa version compilée et les divers fichiers utilisés par Visual Studio pour analyser vos programmes.  
Dès que vous créez un projet, Visual Studio le place dans une solution. Par défaut, celle-ci porte le même nom que le projet.  
Comme vous pourrez par la suite ajouter des projets dans la solution, il est conseillé de donner un nom à la solution qui regroupe les divers projets.  
Exemple : *Module1* pour la solution et *Atelier1* pour le premier projet de ce module concernant l’atelier, puis *Alelier2*… et ensuite *Exercice1* pour le premier projet concernant la série d’exercices associée, puis *Exercice2*…

*La fenêtre suivante est affichée.*

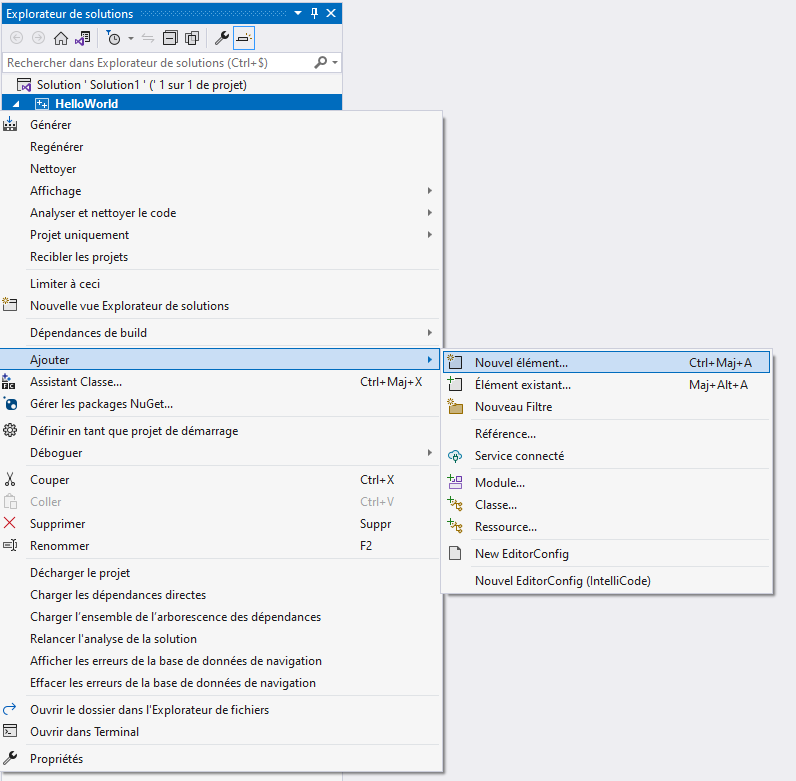
**

1. *Votre projet est créé, ainsi que des répertoires prêts à recevoir les fichiers nécessaires.*

*Une image contenant texte

Description générée automatiquement*

1. *Effectuez un clic droit sur le nom du projet puis choisissez «* ***Ajouter*** *» et «* ***Nouvel élément…*** *» pour ajouter un fichier de code.*



*Si votre solution contient plusieurs projets, celui dont le nom est écrit en gras est celui qui est « actif » (celui qui sera exécuté lorsque vous l’ordonnerez). Vous pouvez changer le projet actif en passant par le menu contextuel : cliquez droit sur le nom du projet que vous voulez rendre actif et sélectionnez «* ***Définir comme projet de démarrage*** *».*

*La fenêtre suivante est affichée.*

*Une image contenant texte

Description générée automatiquement*

1. *Assurez-vous d’être dans le langage «* ***Visual C++*** *». Choisissez l’option «* ***Fichier C++ (.cpp)*** *» dans le menu puis donnez un nom à votre fichier. Par défaut, Visual Studio propose l’extension .cpp (pour C++).* ***(Important)*** *Modifiez l’extension en «* ***.c*** *» pour vous assurer que le fichier soit compilé comme du C plutôt que comme du C++ !  
   Cliquez sur «* ***Ajouter*** *» et le nouveau fichier est ouvert, prêt à recevoir votre code.*

*Une image contenant texte

Description générée automatiquement*

### Éditer son code sous Visual Studio

Au fur et à mesure que Visual Studio reconnaît les mots-clefs et les éléments du code, ceux-ci se coloreront. Pour information, le processus qui permet d’analyser le code au fur et à mesure que vous le tapez, à le colorer, à l’indenter et éventuellement à indiquer des erreurs s’appelle *Intellisense*; son nom est parfois mentionné dans les messages d’erreur.

Une fois le code entré, vous pouvez remarquer deux choses :

* Un astérisque est apparu dans l’onglet, juste à côté du nom du fichier.
* À gauche du code, une longue barre verticale jaune est apparue.

Ces deux indices visuels indiquent la même chose : que le code affiché a été modifié depuis la dernière sauvegarde. Pour enregistrer la nouvelle version du code, vous pouvez passer par le menu et utiliser l’option **File 🡪 Save** ou bien encore vous servir du raccourci **Ctrl-S**.

**L’astérisque indique que le fichier a été modifié depuis la dernière sauvegarde**

Une image contenant texte

Description générée automatiquement

**Les numéros de ligne sont pratiques pour retrouver rapidement à quelle instruction correspond un message d’erreur.**

**La ligne jaune précise quelles parties du code ont été modifiées (ici, tout).**

Une fois la sauvegarde effectuée, l’astérisque disparaît et la barre devient entièrement verte.

Une image contenant texte

Description générée automatiquement

Si, par la suite, vous éditez certaines parties du programme, des portions de la barre verte redeviennent jaunes (uniquement les portions qui sont situées en face des lignes altérées) et l’astérisque réapparaît…

### Compiler et exécuter son code sous Visual Studio

Pour **compiler** le code, il suffit d’appuyer sur **F7** (le raccourci est F6 sous certaines versions de Visual Studio) ou de passer par le menu **Build 🡪 Build Solution**. La compilation prend un certain temps ; vous pouvez suivre son déroulement dans la fenêtre Output. Si tout se passe bien, la dernière ligne indiquera « succeeded ».

**Après compilation, un exécutable a été créé pour le projet.**

Une image contenant texte

Description générée automatiquement

**Tout s’est bien passé !**

Si la compilation a échoué, la fenêtre « Sortie » est remplacée par la liste des erreurs. Visual Studio précise à quel fichier et à quelle ligne de code chacune des erreurs se rapporte.

Une image contenant texte

Description générée automatiquement

En cas de compilation réussie, vous pouvez exécuter le programme en utilisant le raccourci clavier **F5** ou en passant par le menu **Debug 🡪** **Start Debugging.** Une fenêtre console s’ouvrira alors et le programme s’exécutera. Comme la fenêtre console se ferme automatiquement dès la fin de l’exécution, il peut être intéressant d’ajouter une instruction telle que system("pause").

Une image contenant texte

Description générée automatiquement

**Note.** Pour aller plus vite, vous pouvez directement demander l’exécution du code sans demander explicitement la compilation. Dans ce cas-là, Visual Studio vérifiera de lui-même si le code a été modifié depuis la dernière exécution.

**Note (2).** Chaque fois que vous demandez la compilation (ou l’exécution) de votre code, Visual Studio le sauve automatiquement.

### Quelques informations pratiques

**La chasse aux bugs.** Lorsque vous écrivez du code, gardez toujours un œil attentif à la fenêtre des erreurs. IntelliSense repère certaines erreurs (surtout les erreurs de syntaxe) avant la compilation.

**Corriger les bugs.** Pour chaque erreur, Visual Studio indique un numéro de ligne… mais, attention, il s’agit du numéro de la ligne où il s’est rendu compte qu’il y avait une erreur, pas forcément la ligne qui contient l’erreur. Par exemple, si vous oubliez un point-virgule à la ligne 12, l’erreur ne sera détectée qu’à la ligne 13 ! Lors de la chasse aux bugs, ne regardez donc pas seulement la ligne indiquée mais aussi ce qu’il y a plus haut.

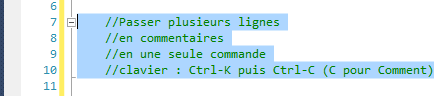
**Comment s’organiser pour les exercices.** Au cours de chaque séance de laboratoire, vous serez invités à résoudre plusieurs exercices différents… comment s’organiser au mieux ? Plusieurs fichiers dans un seul projet ? Plusieurs projets dans une même solution ? Ou carrément plusieurs solutions ? Aucune de ces réponses n’est parfaite… mais voici quelques indications qui vous permettront de choisir l’option qui vous convient le mieux.

* Visual Studio ne peut ouvrir qu’une seule solution à la fois. Si vous créez une solution par exercice, vous devrez donc chaque fois fermer un exercice pour pouvoir en ouvrir un autre.
* Lors de l’exécution d’un projet, c’est toujours la fonction « main » qui est utilisée. Vous ne pouvez donc avoir qu’une seule fonction « main » par projet ! Si vous choisissez de faire tous les exercices dans le même projet, vous devrez soit créer des fonctions de noms différents (et les renommer à chaque fois avant l’exécution), soit faire plusieurs fonctions « main » mais les transformer toutes (sauf une) en commentaire avant chaque exécution.
* Quand vous ordonnez une compilation, Visual Studio compile tous les projets de la solution. Cela signifie que, si un des projets contient un code incorrect, la compilation pourrait ne pas aboutir. Cela peut être problématique si vous n’arrivez pas à terminer un exercice et que vous voulez passer au suivant en attendant (dans ce cas-là, une solution consiste à mettre en commentaires le code incorrect).

**Comment s’y retrouver s’il y a plusieurs projets dans une même solution.** Tout d’abord, pour créer plusieurs projets dans une même solution, une fois celle-ci ouverte, il faut passer par le menu   
«**Fichier 🡪 Ajouter 🡪 Nouveau projet…**». Comme indiqué plus haut, s’il y a plusieurs projets dans une solution, c’est celui dont le nom est écrit en gras qui est actif (cela signifie que c’est la méthode « main » de ce projet qui sera appelée lors de l’exécution). Pour rendre actif un autre projet, il suffit de faire un clic droit sur son nom puis d’utiliser le menu contextuel (option « **Définir comme projet de démarrage** »). Notez que, si l’exécution utilise la méthode main du projet « de démarrage », la compilation, elle, porte à chaque fois sur toute la solution. Cela signifie entre autres que si l’un de vos projets contient un fichier source qui comporte des erreurs, l’exécution ne fonctionnera pas (même si le projet que vous tentez d’exécuter, lui, n’a pas d’erreur) !

**On ne voit rien sur l’écran, c’est trop petit !** Pas de panique : Visual Studio permet d’agrandir la taille des caractères du code en appuyant sur la touche Ctrl puis en utilisant la roulette de la souris (vers l’avant pour augmenter la taille, vers l’arrière pour la diminuer).

**Comment indenter automatiquement un groupe de ligne (pour « l’emballer » dans un bloc marqué par des accolades par exemple) ?** Visual Studio propose un raccourci clavier pour automatiquement indenter (repousser vers la droite) ou dés-indenter (repousser vers la gauche) un ensemble de lignes : il suffit de sélectionner toutes les lignes puis d’appuyer sur TAB pour indenter ou sur Shift-TAB pour dés-indenter.

**Comment commenter tout un groupe de lignes ?** Visual Studio propose un raccourci clavier pour transformer un groupe de lignes en commentaires (en ajoutant // au début de chaque ligne), ainsi qu’un autre raccourci pour « dé-commenter » un groupe de lignes ainsi transformées. Pour transformer en commentaires, il faut sélectionner toutes les lignes concernées puis appuyer sur Ctrl-K suivi de Ctrl-C (C pour « comment »). Pour « dé-commenter » les lignes sélectionnées, on peut appuyer sur Ctrl-K suivi de Ctrl-U (U pour « uncomment »). L’illustration ci-contre montre le résultat obtenu après avoir « commenté » un groupe de lignes.

**Comment trouver les fichiers associés à mon projet ?** Dans l’explorateur de solution (là où les noms des projets et leurs fichiers sont cités), effectuez un clic droit sur le nom de la solution ou d’un projet puis choisissez « **Ouvrir le dossier dans l’Explorateur de fichier** ».

**J’ai inclus la bibliothèque math.h mais il ne reconnaît pas la constante M\_PI !** En fait, même si la constante M\_PI est définie dans la plupart des implémentations du C, elle ne fait pas partie des éléments standards. C’est pour cela que, sous Visual Studio, il ne suffit pas d’ajouter un #include <math.h> pour pouvoir l’utiliser. Par souci de compatibilité, elle se trouve tout de même dans math.h mais sa définition n’est pas automatiquement prise en compte. Pour qu’elle le soit, il faut ajouter la ligne suivante avant l’include de math.h. La ligne en question est :

#define \_USE\_MATH\_DEFINES

**La compilation ne fonctionne pas, et je ne vois pas du tout pourquoi !** Parfois, Visual Studio s’empêtre dans les fichiers temporaires et réutilise de l’ancien code plutôt que de le recoder. Pour « nettoyer » ces fichiers temporaires et le forcer à reprendre la compilation à zéro, utilisez la commande « **Générer 🡪 Nettoyer la solution »** puis relancer une nouvelle compilation.

La compilation pourrait aussi ne pas fonctionner parce que l’un des projets de la solution courante comporte des erreurs. Chaque fois que Visual Studio effectue une compilation, il le fait pour tous les projets de la solution courante, pas seulement le projet « de démarrage » (celui dont le nom est écrit en gras). Si un des autres projets de la solution courante ne compile pas, la compilation globale échouera.

**Je ne parviens pas à fermer la fenêtre console où mon programme s’exécute !** Si votre programme se lance dans une boucle infinie ou tourne sans fin pour une autre raison, vous pouvez interrompre son exécution à partir de Visual Studio en utilisant le raccourci clavier **Shift-F5** ou encore la commande « **Déboguer 🡪 Arrêter le débogage** ».

1. Voir <https://visualstudio.microsoft.com/fr/downloads/> [↑](#footnote-ref-1)