|  |  |
| --- | --- |
| |  | | --- | | **Introduction à Code::Blocks**  Ce document décrit comment installer puis utiliser l'environnement Code::Blocks, vous permettant de tester vos programmes C/C++ chez vous, avant de les valider sur notre site.  Code::Blocks est un environnement de développement (IDE) permettant d'écrire et tester ses programmes en C ou C++. Il peut être utilisé en combinaison avec divers compilateurs C/C++. Pour installer simplement une version incluant un compilateur (gcc, via MinGW), téléchargez [cet installeur](http://prdownloads.sourceforge.net/codeblocks/codeblocks-1.0rc2_mingw.exe?download).  Après avoir téléchargé Code::Blocks, lancez l'exécutable, et suivez l'installation pas à pas. Vous pouvez conserver toutes les options par défaut.  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_install_1.PNG  Une fois l'installation terminée, lancez Code::Blocks. Voici à quoi ressemble l'environnement, par défaut. La zone de gauche permet de naviguer entre les différents fichiers d'un projet, la zone de droite est réservée à l'édition de votre code, et la zone du bas permet d'afficher entre autres le résultat de la compilation.  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_start.PNG  Pour pouvoir tester votre premier programme, il faut commencer par créer un projet. Pour cela, allez dans le menu File, puis cliquez sur "new Project". Choisissez la catégorie "Console", Sélectionnez "Console Application", et choisissez "C source" dans la zone "File(s) options".  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_new_project.PNG  Par défaut, votre projet est créé avec un fichier principal main.c, contenant un exemple de programme, qui affiche "Hello World !". Vous verrez dans le cours à quoi correspondent les différentes instructions de ce programme.  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_hello_world_project.PNG  Pour tester ce programme, allez dans le menu "Build", puis sélectionnez "Build & Run". Vous pouvez aussi vous contenter d'appuyer sur la touche F9. S'il n'y a pas d'erreur lors de la compilation, le programme se lance alors dans une fenêtre Dos, puis vous demande d'appuyer sur la touche Entrée pour fermer cette fenêtre.  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_hello_world_result.PNG  Dans la plupart des exercices proposés sur ce site, on vous demande d'écrire un programme qui lit des données en entrée, puis affiche un résultat. Le programme le plus simple de ce type se contente de lire un nombre en entrée, et d'afficher le nombre lu :  #include <stdio.h>  int main()  {  int nombreLu;  scanf("%d", &nombreLu);  printf("%d", nombreLu);  return 0;  }  Vous pouvez remplacer le code affichant "Hello World !" par le code ci-dessus. Ici aussi, nous verrons en détail dans le cours à quoi correspondent ces instructions. Sachez simplement que ce programme se contente de lire un nombre tapé au clavier, puis de l'afficher.  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_input_output_project.PNG  Pour tester ce programme, une première possibilité est de le lancer, puis de taper un nombre au clavier, et voir le résultat. Pour éviter d'avoir à taper toute l'entrée au clavier à chaque fois, et risquer de vous tromper, nous vous conseillons de saisir les données une fois pour toutes dans un fichier texte. Par exemple nous créons ici un fichier nommé "test.in", et contenant le nombre 42 :  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_input_output_in.PNG  Il est ensuite possible d'indiquer à Code::Blocks de fournir automatiquement le contenu de ce fichier en entrée à votre programme, lorsqu'il l'exécutera. Pour cela, allez dans le menu "Project", puis "Set program's arguments...".  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_menu_prog_args.PNG  Dans la zone "Program arguments", entrez alors "< test.in" (vous pouvez remplacer test.in par le nom du fichier d'entrée que vous avez choisi.  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_select_target.PNG  Vous pouvez alors compilez le programme et l'exécuter en appuyant sur F9. Vous voyez alors que la commande scanf("%d", &nombreLu); lit bien le nombre stocké dans le fichier. Son contenu a en effet été transmis au programme comme s'il avait été tapé au clavier :  http://www.france-ioi.org/train/prog/cours_c/codeblocks/cb_input_output_result.PNG  Voilà, vous avez tous les éléments pour commencer le cours et tester vos programmes chez vous avant de les soumettre sur notre site pour les valider.  Code::Blocks est un environnement puissant avec de très nombreuses options utiles. La plupart ne sont pas nécessaires dans le cadre de ce cours. Il peut cependant être intéressant de tester les fonctionnalités de "débuggueur", permettant par exemple d'exécuter pas à pas le programme pour bien comprendre son déroulement. Dans le menu "Debug", essayez en particlier la commande "Step Into", accessible par Shift+F7, qui exécute uniquement la prochaine instruction (lancez là plusieurs fois pour exécuter les instructions de votre programme une à une). | |

**Les outils nécessaires au programmeur**

Alors à votre avis, de quels outils un programmeur a-t-il besoin ?  
Si vous avez attentivement suivi le chapitre précédent, vous devez en connaître au moins un !  
  
Vous voyez de quoi je parle ?  
?  
?  
?  
Vraiment pas ? ^^  
  
Eh oui, il s'agit du **compilateur**, ce fameux programme qui permet de traduire votre langage C en langage binaire !  
  
Comme je vous l'avais un peu déjà dit dans le premier chapitre, il existe plusieurs compilateurs pour le langage C / C++. Nous allons voir que le choix du compilateur ne sera pas très compliqué dans notre cas ;)  
  
Bon, de quoi d'autre a-t-on besoin ?  
Je ne vais pas vous laisser deviner plus longtemps ^^ Voici le strict minimum pour un programmeur :

* **Un éditeur de texte** pour écrire le code source du programme (en C ou C++). En théorie un logiciel comme le Bloc-Notes sous Windows, ou "*vi*" sous Linux fait l'affaire. L'idéal, c'est d'avoir un éditeur de texte intelligent qui colore tout seul le code, ce qui vous permet de vous repérer dedans bien plus facilement
* **Un compilateur** pour transformer ("compiler") votre source en binaire.
* **Un débugger** pour vous aider à traquer les erreurs dans votre programme (on n'a malheureusement pas encore inventé le "correcteur", un truc qui corrigerait tout seul nos erreurs :soleil: )

A priori, si vous êtes un casse-cou de l'extrême, vous pouvez vous passer de débugger ? Mais bon, je sais pertinemment que dans moins de 5 minutes vous reviendrez en pleurnichant me demander où on peut trouver un débugger qui marche bien ^^  
  
A partir de maintenant on a 2 possibilités :

* Soit on récupère chacun de ces 3 programmes **séparément**. C'est la méthode la plus compliquée, mais elle fonctionne ^^ Sous Linux en particulier, bon nombre de programmeurs préfèrent utiliser ces 3 programmes séparément. Je ne détaillerai pas cette méthode ici, je vais plutôt vous parler de la méthode simple.
* Soit on utilise un programme "3-en-1" (comme les liquides vaisselle, oui oui) qui combine éditeur de texte, compilateur et débugger. Ces programmes "3-en-1" sont appelés **IDE**, ou encore "Environnements de développement"

Il existe plusieurs environnements de développement. Vous aurez peut-être un peu de mal à choisir celui qui vous plaît au début. Une chose est sûre en tout cas: vous pouvez faire n'importe quel type de programme, quel que soit l'IDE que vous choisissez.

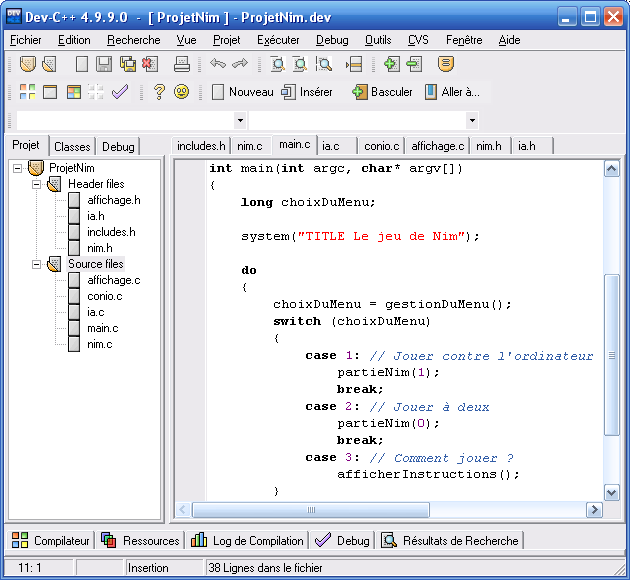
**Choisissez votre IDE**

Il m'a semblé intéressant de vous montrer 3 IDE parmi les plus connus. Tous sont disponibles gratuitement. Personnellement, je navigue un peu entre tous ceux-là et j'utilise l'IDE qui me plaît selon l'humeur du jour ;)

* Vous avez par exemple **Dev C++** qui est très bien. Que son nom ne vous trompe pas : vous pouvez aussi bien faire du C que du C++ avec lui ;)
* Plus récent que Dev C++, l'IDE **Code::Blocks** semble promis à un bel avenir. Il est aussi gratuit et plus tenu à jour que Dev. Il possède en outre quelques fonctionnalités intéressantes et fonctionne sous Windows et Linux. Je conseille d'utiliser celui-ci pour débuter.
* Un des IDE les plus connus, c'est celui de Microsoft : **Visual C++**. Il existe à la base en version payante (chère !), mais heureusement il existe une version gratuite intitulée **Visual C++ Express** qui est vraiment très bien (il y a peu de différences avec la version payante).
* Quel est le meilleur de tous ces IDE ?
* Personnellement, entre Dev C++, Code::Blocks et Visual C++ j'aurais tendance à préférer Visual. Je trouve son débugger plus puissant. Toutefois, comme vous débutez vous ne serez pas capables de profiter de toute sa puissance. Vous vous y mettrez sûrement, mais plus tard.  
    
  Cela nous laisse donc un choix entre Dev C++ et Code::Blocks pour commencer. Grosso modo, il faut retenir que Dev-C++ est un IDE qui a eu beaucoup de succès pendant un moment, mais il n'est plus mis à jour depuis bien trop longtemps à mon goût. Vous en entendrez quand même sûrement parler car beaucoup de personnes l'ont utilisé et l'utilisent encore.  
  Je vous recommande donc Code::Blocks pour commencer, mais ce n'est pas une obligation. Quel que soit l'IDE que vous choisissez vous serez capables de faire autant de choses. Vous n'êtes pas limités.  
    
  Dans la suite de ce chapitre, je vais présenter chacun de ces 3 IDE histoire que vous puissiez un peu les comparer, ne serait-ce que visuellement.  
    
  Au passage, notez que ces IDE sont disponibles sous Windows uniquement, sauf Code::Blocks qui fonctionne aussi sous Linux.
* Attention : si vous êtes sous Windows Vista, il faudra suivre des instructions particulières pour installer votre IDE correctement, sinon vous aurez des bugs à la compilation.  
  Je vous conseille d'installer Visual Studio Express (il marche sous Vista sans problème), ou encore d'installer Code::Blocks en suivant les instructions spéciales pour Vista que j'y donne.
* Et si je suis sous Mac ?
* 

## Si vous êtes sous Mac, sachez qu'il existe un IDE appelé "Xcode" et qu'il est présent sur le CD d'installation de Mac OS. Nous verrons son fonctionnement vers la fin de ce chapitre. Vous pouvez choisir... Dev-C++

Dev C++ est un environnement de développement (IDE) gratuit. C’est probablement le plus connu de tous. Toutefois, il n'est plus trop mis à jour. Aujourd'hui, on recommande de plus en plus d'utiliser [Code::Blocks](http://www.siteduzero.com/tutoriel-3-13920-ayez-les-bons-outils.html" \l "ss_part_4) (aussi gratuit, présenté plus bas).  
  
Dev C++ est disponible en français et vous pouvez le télécharger sur Internet rapidement. C'est avec cet IDE que j'ai commencé à rédiger ce cours, mais cela fait un moment que j'utilise Code::Blocks et Visual C++. Ne soyez donc pas étonnés si je fais souvent référence à Dev-C++ au début du cours.

  
*Dev-C++ est gratuit et a tout ce qu’il faut pour programmer !*

Comme tout programme, il y en a eu plusieurs versions. Les captures d’écran que je fais sont sur la version 4.9.9.0 comme vous pouvez le voir.  
Ce genre de programme évolue vite, mais si vous avez une version supérieure ne vous inquiétez pas. Le fonctionnement du programme ne change pas d’une version à l’autre. Peut-être avez-vous de nouvelles icônes, et encore… ^^  
  
Allez sur le site de Bloodshed (l’éditeur du programme) pour le récupérer. Prenez le premier lien de téléchargement que vous voyez (*Dev-C++ with Mingw/GCC* ) :

[Site web de Dev-C++](http://www.bloodshed.net/dev/devcpp.html)

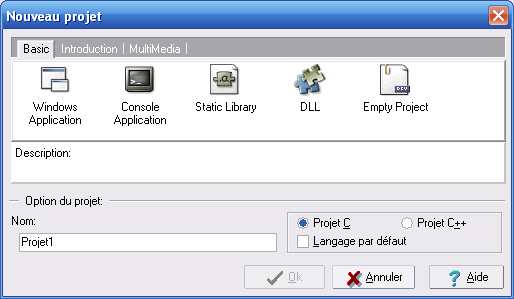
A l’installation, faites tout ce qu’on vous recommande de faire. Ca se passe normalement assez vite :)

### Le démarrage de Dev-C++

Lancez Dev C++.  
La première fois, il vous demandera de le configurer. Il vous demandera votre langue et l'aspect du logiciel que vous voulez avoir.  
Par ailleurs, il vous posera ensuite 2-3 questions sur la création de fichiers spéciaux pour vous aider. Je vous conseille de laisser les options par défaut, à savoir "*Oui, je le veux*" ^^  
Ca ne vous sera pas utile de suite, mais plus tard vous ne regretterez pas d'avoir répondu oui ;)  
  
Au départ, rien ne s’affiche. Il va falloir demander à Dev C++ de créer un nouveau projet.

Un **projet** c’est l’ensemble de tous les fichiers source du programme. En effet, quand on programme, on sépare souvent notre code dans plusieurs fichiers différents. Ces fichiers seront ensuite "combinés" par le compilateur qui en fera un exécutable (un ".exe").

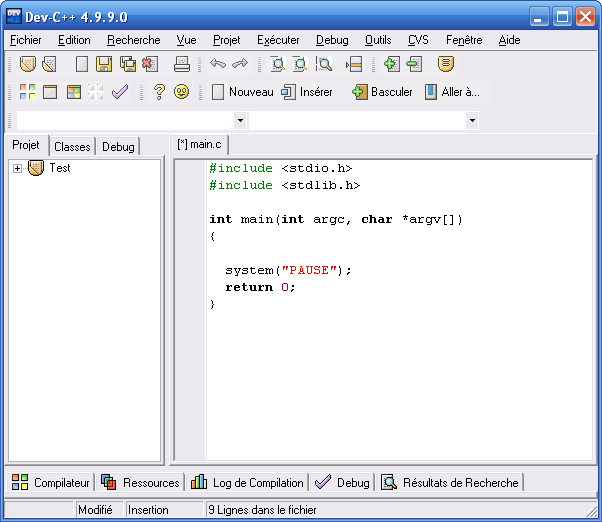
Pour créer un nouveau projet c’est simple : allez dans le menu "Fichier / Nouveau / Projet". Vous devriez voir quelque chose qui ressemble à ça :

  
*La fenêtre de création de projet de Dev C++*

Là, on vous demande quel genre de programme vous voulez créer. Retenez bien la marche à suivre, car vous devrez faire cela la plupart du temps (surtout au début) :

* Cliquez sur "**Console Application**". Eh oui, il n’est pas possible de commencer par créer des fenêtres avec "Windows Application", il est vraiment trop tôt ^^ On va pour commencer se contenter de créer des programmes qui s’affichent dans une console, qui ressemble un peu à DOS.
* Sélectionnez "**Projet C**" si ce n’est déjà fait.
* Cochez "**Langage par défaut**"
* Donnez **un nom** à votre projet (autre que "Projet 1")

Faites OK.  
On vous demande alors où placer le fichier " .dev ". Ce fichier, propre à Dev C++, est le fichier de votre projet. Il contient la liste des fichiers source de votre programme. Il vous faut enregistrer votre projet avant même d’avoir commencé à programmer ! Remarquez, avec Visual C++ c’est pareil, sauf que lui il n’utilise pas un fichier de projet mais plusieurs :p  
  
Je vous conseille de créer un dossier pour votre projet.  
Une fois que vous avez indiqué où enregistrer votre projet, Dev C++ crée alors un premier fichier source qui s’appelle " main.c ". Ce sera le fichier principal de notre programme, on aura l’occasion d’en reparler dans le prochain chapitre.  
  
Normalement, Dev C++ écrit déjà un peu de code dedans (le strict minimum). N’essayez pas de deviner ce qu’il signifie, attendez plutôt le chapitre suivant qu’on analyse ça en détail ;)

  
*Un nouveau projet tout neuf !*

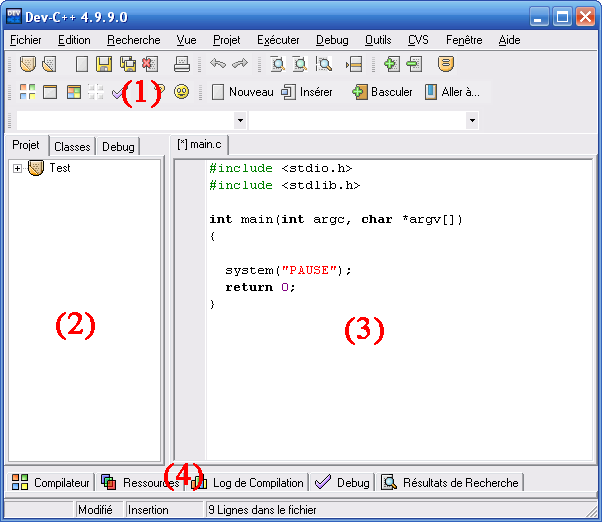
Pour ceux qui seraient déjà un peu perdus, j'ai réalisé une vidéo vous montrant comment je crée un nouveau projet sous Dev-C+. Je suis exactement les mêmes étapes que je vous ai énoncées plus haut :

[**Créer un nouveau projet avec Dev-C++ (254 Ko)**](http://www.siteduzero.com/uploads/fr/ftp/mateo21/dev_nv_projet.html)

Que dire de plus sur Dev C++ ?  
Il est constitué de plusieurs parties qu’on a tout intérêt à regarder de plus près histoire de voir comment tout ce bazar fonctionne ^^

### Les principales fonctionnalités de Dev-C++

Voyons voir plus en détail comment Dev-C++ est organisé :

  
*Les différentes parties de Dev C++*

J’ai séparé Dev en 4 grandes parties :

1. En haut, vous avez les menus et la barre d’outils. Je peux vous conseiller d’aller modifier un peu les options si ça vous chante. C’est dans les menus Outils / Options d’environnement et Outils / Options de l’éditeur.  
   Pour ce qui est des icônes de la barre d’outils, qu’on utilisera souvent, il y en a beaucoup que vous connaissez. Les premières servent notamment à créer un nouveau projet, un nouveau fichier, à enregistrer le fichier, à enregistrer tous les fichiers ouverts etc. Je souhaite attirer votre attention sur les boutons se situant au début de la deuxième ligne (du moins sur ma capture d’écran) :

Image utilisateur  
*Les icônes lançant la compilation*

Ces 5 icônes sont sans aucun doute les plus utilisées, et pour cause : ce sont elles qui permettent d’appeler le compilateur pour créer un exécutable de votre projet :)  
Dans l’ordre, de gauche à droite, ces icônes signifient :

* + **Compiler**: tous les fichiers source de votre projet sont envoyés au compilateur qui va se charger de créer un exécutable. S’il y a des erreurs (ce qui a de fortes chances d’arriver :D ), l’exécutable ne sera pas créé et on vous indiquera les erreurs en bas de DevC++ (dans la partie que j’ai numérotée 4)
  + **Exécuter** : cette icône lance juste le dernier exécutable que vous avez compilé. Cela vous permettra donc de tester votre programme et voir ainsi ce qu’il donne;) Dans l’ordre, si vous avez bien suivi, on doit d’abord compiler, puis exécuter pour tester ce que ça donne. On peut aussi utiliser le 3ème bouton…
  + **Compiler & Exécuter** : pas besoin d’être un génie pour comprendre que c’est la combinaison des 2 boutons précédents. C’est d’ailleurs ce bouton que vous utiliserez le plus souvent. Notez que s’il y a des erreurs pendant la compilation (pendant la génération de l’exécutable), le programme ne sera pas exécuté. A la place, vous aurez droit à une beeelle liste d’erreurs à corriger :p
  + **Tout reconstruire** : quand vous faites " Compiler ", DevC++ ne recompile en fait que les fichiers que vous avez modifiés et pas les autres. Parfois, je dis bien parfois, vous aurez besoin de demander à Dev de vous recompiler tous les fichiers. On verra plus tard quand on a besoin de ce bouton, et vous verrez plus en détail le fonctionnement de la compilation dans un chapitre futur. Pour l’instant, on se contente de savoir le minimum nécessaire pour pas tout mélanger ;)  
    Ce bouton ne nous sera donc pas utile de suite.
  + **Débugger** : ce bouton lance votre programme en mode débuggage. C’est un mode particulier qu’on apprendra à utiliser plus tard. Cela vous permet de traquer les erreurs de votre programme, de le mettre en " pause " lors de son exécution etc etc.

Je vous conseille d’utiliser les raccourcis plutôt que de cliquer sur les boutons, parce que c’est quelque chose qu’on fait vraiment très très souvent. Vous pouvez connaître le raccourci en pointant sur le bouton qui vous intéresse. Chez moi par exemple, je tape F9 pour faire " Compiler & Exécuter "

1. Dans la section de gauche de Dev-C++ s’affichent en général tous les fichiers de votre projet (qui s’appelle " Test " sur ma capture d’écran). Cliquez sur le petit " + " à gauche pour dérouler la liste des fichiers ouverts. On se sert souvent de cette liste pour naviguer d’un fichier du projet à un autre.  
   Les onglets en haut de cette section sont les suivants :
   * **Projet** : c’est là que vous avez la liste des fichiers du projet dont je viens de vous parler.
   * **Classes**: c’est un onglet que nous n’utiliserons pas en langage C. Cela ne sert que quand on fait du C++, et ne comptez pas sur moi pour vous expliquer maintenant ce que c’est :p
   * **Debug** : c’est l’onglet qui est utilisé pendant que vous débuggez votre programme. Il permet en particulier de voir ce qu’il y a dans votre mémoire vive. Vous ne savez pas ce que c’est une "mémoire vive" ? Ce n’est pas grave, on aura le temps de le découvrir plus tard ;)
2. Ah, la partie principale :) C’est là que s’affiche le fichier source en C ou C++ que vous êtes en train de modifier. C’est dans cette zone de l’écran que vous passerez le plus clair de votre temps ^^  
   Notez qu’en haut de cette zone, tous les fichiers ouverts apparaissent sous forme d’onglets. Sur ma capture d’écran il n’y a que "main.c" pour le moment. Cliquez sur l’un d’eux pour afficher le fichier correspondant.  
     
   Autre info utile : tous les fichiers modifiés et non enregistrés sont précédés d’une petite étoile [\*]. Enregistrez souvent. Enregistrez tout le temps. On ne compte plus le nombre de tentatives de suicide de personnes qui avaient oublié d’enregistrer et qui ont eu une coupure de courant (je rigole hein, vous jetez pas par la fenêtre si ça vous arrive, ça résoudra rien :p ). N’hésitez pas à utiliser le bouton "Sauvegarder tout" de la barre d’outils Image utilisateur , il enregistre tous les fichiers ouverts d’un seul coup.
3. Le bas de l’écran… C’est la zone que détestent tous les programmeurs. En effet, lors d’une compilation qui "plante", les erreurs s’affichent dans la partie basse de l’écran. En général, vous ne naviguerez pas trop entre les onglets de cette partie, sauf peut-être l’onglet *Debug*pour débugger votre programme et l’onglet "log de compilation" qui indique si la compilation s’est bien passée ou non.

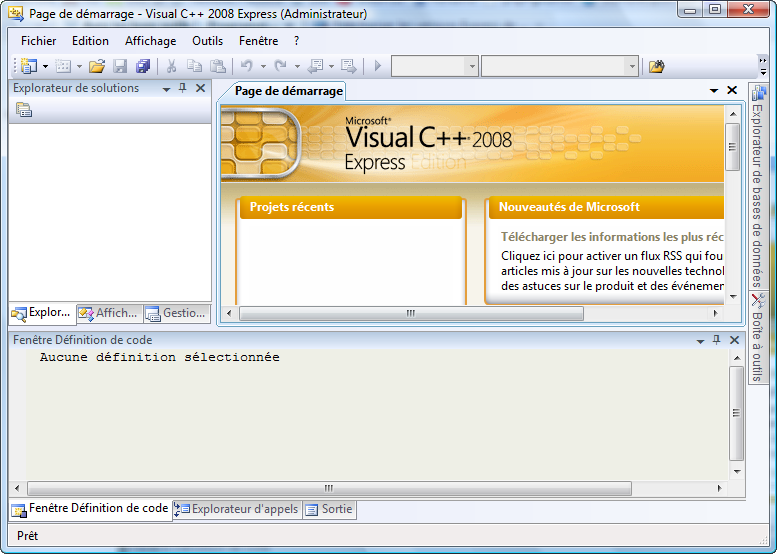
Pfiou ! On a fait à peu près le tour de Dev C++ :D  
On a vu les principales sections du programme qu’on utiliserait. Normalement ça devrait vous permettre de vous débrouiller la plupart du temps :)  
  
Passons maintenant à Visual C++ !

## Ou bien... Visual C++

Quelques petits rappels sur Visual C++ :

* C'est l'IDE de Microsoft
* Il est à la base payant, mais Microsoft a sorti une version gratuite intitulée Visual C++ Express.

Nous allons bien entendu voir ici la version gratuite, Visual C++ Express :)

  
*Aperçu de Visual C++ Express*

Quelles sont les différences avec le "vrai" Visual ?

Il n'y a pas d'éditeur de ressources (vous permettant de dessiner des images, des icônes, ou des fenêtres). Mais bon, ça entre nous on s'en fout parce qu'on n'aura pas besoin de s'en servir dans ce tutorial ;) Ce ne sont pas des fonctionnalités indispensables bien au contraire.  
  
Vous trouverez les instructions pour télécharger Visual C++ Express à cette adresse :

[Site de Visual C++ Express Edition](http://msdn.microsoft.com/fr-fr/express/aa975050.aspx)

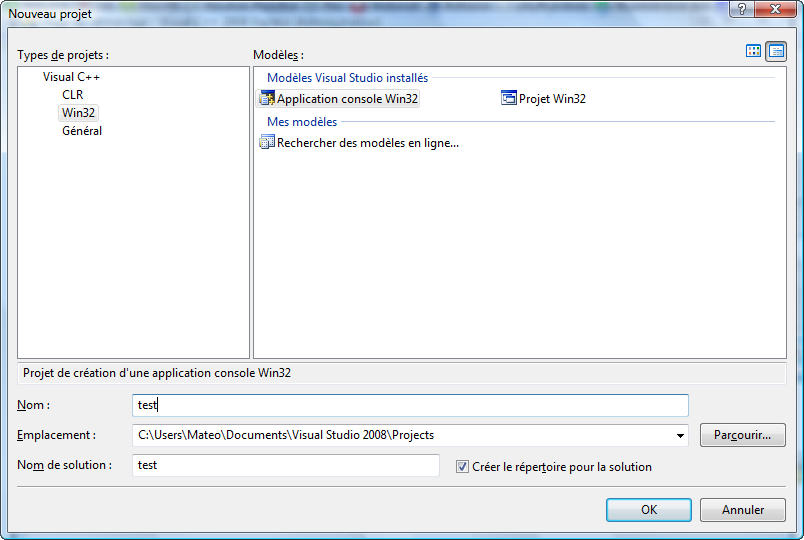
Sélectionnez Visual C++ Express Français un peu plus bas sur la page.  
  
Visual C++ Express est en français et est totalement gratuit. Ce n'est donc pas une version d'essai limitée dans le temps.  
  
C'est une chance d'avoir un IDE aussi puissant que celui de Microsoft disponible gratuitement, donc ne la laissez pas passer ;)

### Installation

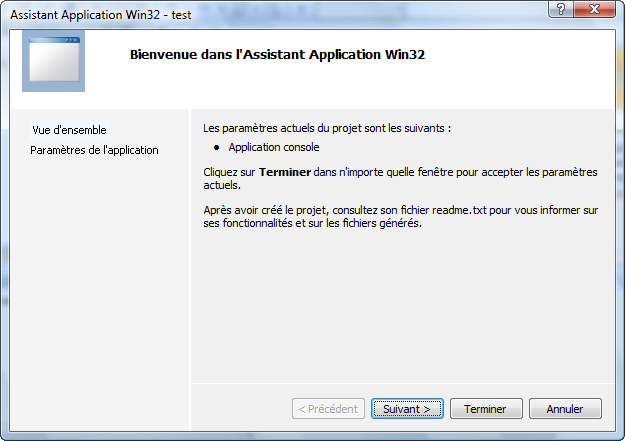
L'installation devrait normalement se passer sans encombre. Le programme d'installation va télécharger la dernière version de Visual sur Internet.  
Je vous conseille de laisser les options par défaut.  
  
A la fin, on vous dit qu'il faut vous enregistrer dans les 30 jours. Pas de panique, c'est gratuit et rapide mais il faut le faire.  
Cliquez sur le lien qui vous est donné : vous arrivez sur le site de Microsoft. Connectez-vous avec votre compte Windows Live ID (équivalent du compte hotmail ou msn) ou créez-en un si vous n'en avez pas, puis répondez au petit questionnaire.  
  
On vous donnera à la fin une clé d'enregistrement. Vous devrez recopier cette clé dans le menu "?" / "Inscrire le produit".

### Créer un nouveau projet

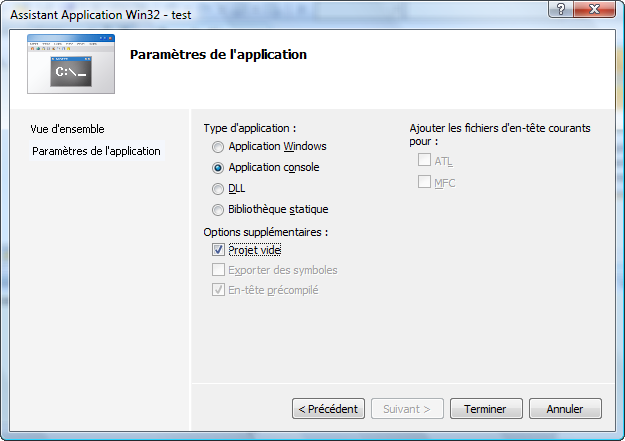
Pour créer un nouveau projet sous Visual, allez dans le menu Fichier / Nouveau / Projet.  
Sélectionnez "Win32" dans la colonne de gauche, puis "Application console Win32" à droite.  
  
Entrez un nom pour votre projet, par exemple "test" :



Validez. Une nouvelle fenêtre s'ouvre :



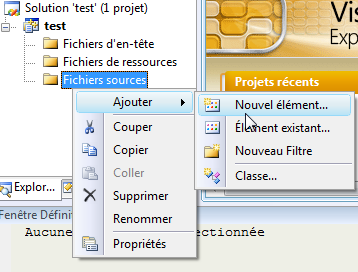
Cette fenêtre ne sert à rien ^^  
Par contre, cliquez sur "Paramètres de l'application" dans la colonne de gauche :



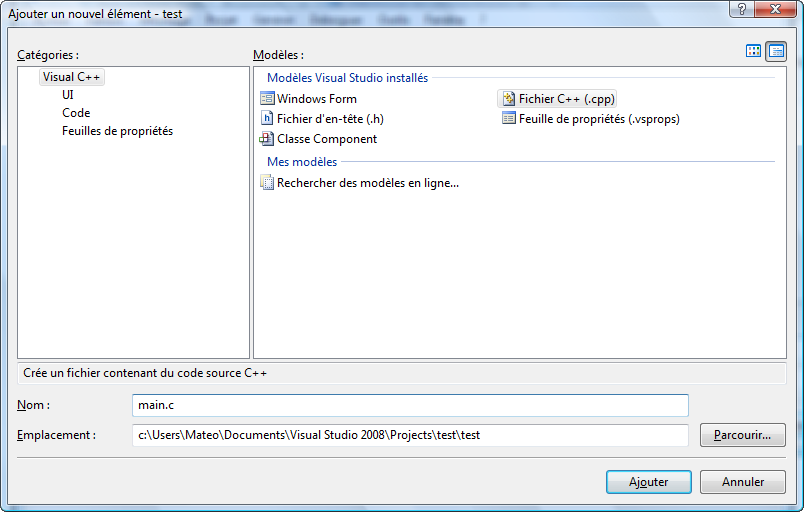
Veillez à ce que "Projet vide" soit coché comme sur ma capture d'écran.  
Puis, cliquez sur "Terminer".

### Ajouter un nouveau fichier source

Votre projet est pour l'instant bien vide. Faites un clic droit sur le dossier "Fichiers sources" situé sur votre gauche, puis allez dans Ajouter / Nouvel élément :



Une fenêtre s'ouvre.  
Sélectionnez "Fichier C++ (.cpp)" (je sais, on ne fait pas de C++ mais ça n'a pas d'importance ici). Entrez un nom pour votre fichier : "main.c", comme sur ma capture d'écran :



Cliquez sur "Ajouter". C'est bon, vous allez pouvoir commencer à écrire du code !

### La fenêtre principale de Visual

Voyons ensemble le contenu de la fenêtre principale de Visual C++ Express :



Comme vous pouvez le constater, il ressemble pas mal à Dev-C++.  
  
On va rapidement (re)voir quand même ce que signifient chacune des parties :

1. La barre d'outils, tout ce qu'il y a de plus standard. Ouvrir, enregistrer, enregistrer tout, couper, copier, coller etc. Par défaut, il semble qu'il n'y ait pas de bouton de barre d'outils pour compiler. Vous pouvez les rajouter en faisant un clic droit sur la barre d'outils, puis en choisissant "Déboguer" et "Générer" dans la liste.  
   Toutes ces icônes de compilation ont leur équivalent dans les menus "Générer" et "Déboguer". Si vous faites "Générer", cela créera l'exécutable (ça signifie "Compiler" pour Visual). Si vous faites "Déboguer / Exécuter", on devrait vous proposer de compiler avant d'exécuter le programme. F7 permet de générer le projet, et F5 de l'exécuter.
2. Dans cette zone très importante vous voyez normalement la liste des fichiers de votre projet. Cliquez sur l'onglet "Explorateur de solutions" en bas si ce n'est déjà fait. Vous devriez voir que Visual crée déjà des dossiers pour séparer les différents types de fichiers de votre projet (sources, en-tête et ressources). Nous verrons un peu plus tard quels sont les différents types de fichiers qui constituent un projet :)
3. La partie principale. C'est là qu'on modifie les fichiers source.
4. C'est là encore la "zone de la mort", celle où on voit apparaître toutes les erreurs de compilation. C'est dans le bas de l'écran aussi que Visual affiche les informations de débuggage quand vous essayez de corriger un programme buggé. Je vous ai d'ailleurs dit tout à l'heure que j'aimais beaucoup le débugger de Visual, et je pense que je ne suis pas le seul ^^ On essaiera d'apprendre à l'utiliser un peu plus tard si on trouve le temps.

Voilà, on a fait le tour de Visual C++.  
Vous pouvez aller jeter un œil dans les options (Outils / Options) si ça vous chante, mais n'y passez pas 3 heures. Il faut dire qu'il y a tellement de cases à cocher de partout qu'on ne sait plus trop où donner de la tête :lol:

## Ou encore... Code::Blocks

Code::Blocks est un IDE libre et gratuit, plus récent que Dev-C++.  
Si Dev reste un des IDE gratuits les plus connus, il le doit surtout à son ancienneté.  
  
Code::Blocks étant relativement nouveau, il n'est pas encore aussi connu. Pourtant, n'allez pas penser qu'il est peu avancé pour autant ! Bien au contraire, je dois avouer que j'ai été surpris par cet éditeur et je vous recommande de l'essayer :)  
  
En outre, Code::Blocks est disponible **pour Windows, Mac et Linux**.  
Code::Blocks n'est disponible pour le moment qu'en anglais. Ca ne devrait PAS vous repousser à l'utiliser. Quand vous programmerez vous serez de toute façon confronté bien souvent à des documentations en anglais, donc raison de plus pour s'entraîner à utiliser cette langue. Ca ne complique pas l'utilisation du logiciel de toute manière.

### Télécharger Code::Blocks

Rendez-vous [sur la page de téléchargements de Code::Blocks](http://www.codeblocks.org/downloads.shtml).  
  
Si vous êtes sous Windows, repérez la section "Windows" un peu plus bas sur cette page. Téléchargez le logiciel en prenant le programme qui contient mingw dans le nom (ex. : codeblocks-8.02mingw-setup.exe). L'autre version étant sans compilateur, vous auriez eu du mal à compiler vos programmes :p  
Si vous êtes sous Linux, il y a un lien en haut pour choisir le package RPM à télécharger.  
  
L'installation est très simple et rapide. Laissez toutes les options par défaut et lancez le programme.

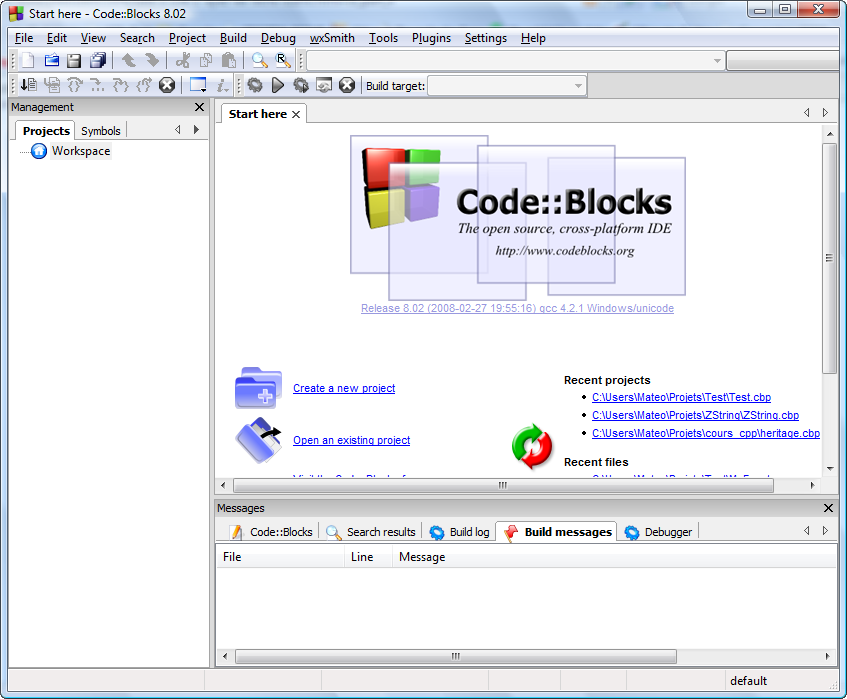
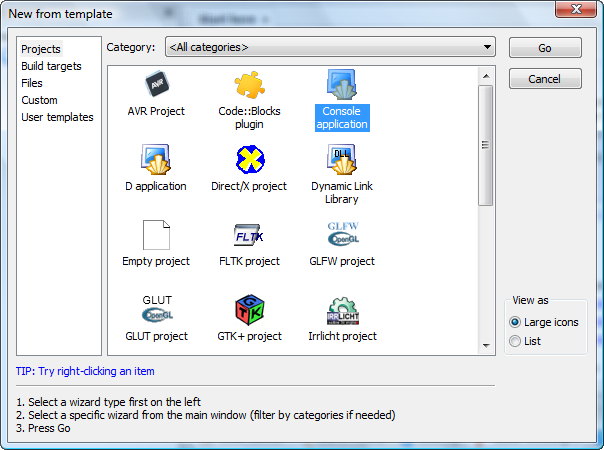


Image utilisateur

Le fonctionnement du programme est quasiment le même que pour Dev et Visual, vous ne serez pas perdus.  
Vous trouverez là encore dans la barre d'outils les boutons (dans l'ordre) "Compiler", "Exécuter", "Compiler & Exécuter" et "Tout recompiler" (comme Dev ^^ )

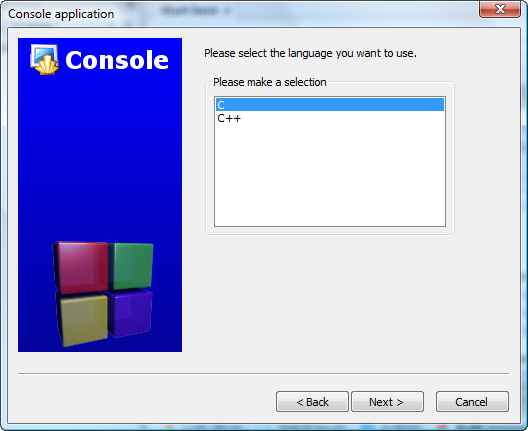
### Créer un nouveau projet

Pour créer un nouveau projet c'est très simple : allez dans le menu File / New / Project.  
Dans la fenêtre qui s'ouvre, choisissez "Console application" :

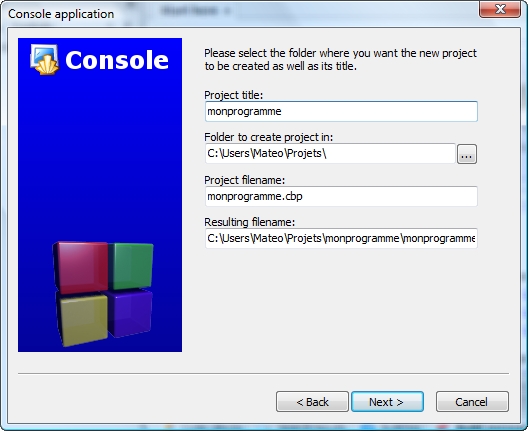


Comme vous pouvez le voir, Code::Blocks propose de réaliser pas mal de types de programmes différents qui utilisent des librairies connues comme la SDL (2D), OpenGL (3D), QT et wxWidgets (Fenêtres) etc etc... Pour l'instant, ces icônes servent plutôt à faire joli car les librairies ne sont pas installés sur votre ordinateur, vous ne pourrez donc pas les faire marcher.  
Nous nous intéresserons à ces autres types de programmes bien plus tard. En attendant il faudra vous contenter de "Console", car vous n'avez pas encore le niveau nécessaire pour créer les autres types de programmes.

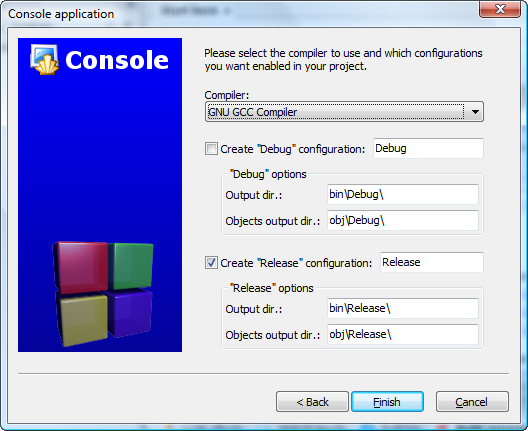
Cliquez sur "Go" pour créer le projet. Un assistant s'ouvre.  
  
Faites "Next", la première page ne servant à rien.  
On vous demande ensuite si vous allez faire du C ou du C++ : répondez C.



On vous demande le nom de votre projet, et dans quel dossier les fichiers source seront enregistrés :

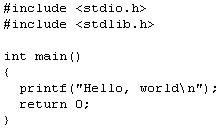
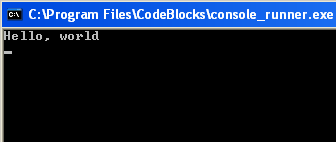
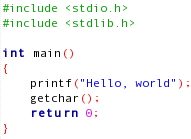


Enfin, la dernière page vous permet de choisir de quelle façon le programme doit être compilé. Vous pouvez laisser les options par défaut, ça n'aura pas d'incidence pour ce que nous allons faire dans l'immédiat (veillez à ce que "Debug" ou "Release" au moins soit coché).



Cliquez sur "Finish", c'est bon !  
Code::Blocks vous créera un premier projet avec déjà un tout petit peu de code source dedans ;)  
  
Dans le cadre de gauche "Projects", développez l'arborescence en cliquant sur le petit "+" pour afficher la liste des fichiers du projet. Vous devriez avoir au moins un main.c que vous pourrez ouvrir en double-cliquant dessus.

**Procédure pour installer la traduction française sur XP/Vista**  
1- Télécharger le zip fourni en pièce jointe  
  
2- Copier le dossier "locale" dans "C:\Program Files\CodeBlocks\share CodeBlocks"  
  
3- Lancer Code::Blocks  
  
4- Aller dans le menu "Settings -> Environment ..."  
  
5- A gauche de la fenêtre "Environment settings" cliquer sur "View"  
  
6- Cocher la case Internationalization (needs restart)  
  
7- Puis juste à droite choisissez "French"  
  
8- Cliquer en bas de la fenêtre sur OK  
  
9- Fermer Code::Blocks  
  
10-Relancer Code::Blocks, il est en français du moins une bonne partie.

**Votre premier programme en C**  
  
  
Bonjour à tous, je me présente, je suis Elfen, je serais votre guide durant ce cours de C qui sera composé de plusieurs "tutos". J'essayerais de faire un cours le plus simple possible.  
  
  
**1/ Le programme**  
  
Ça y est, vous y êtes, vous allez bientôt apprendre un des langages les plus puissant et un des plus utilisés du monde, le langage dans lequel est programmé plus de la moitié (et je suis gentil) de votre système d'exploitation : windows, probablement.  
  
Venons-en a ce pour quoi vous êtes ici : coder votre premier programme en C, le voici :  
  
  
  
Mais, qu'est-ce que ce programme peut bien faire ?  
Et bien j'ai le plaisir de vous annoncer que vous avez tapé 101 caractères pour finalement écrire la phrase : "Hello, world" dans une immonde mocheté (question de goût, perso je ne la trouve pas si moche que ça http://www.top-logiciel.net/uploads/smil3dbd4e398ff7b.gif ) : la console. Si vous êtes bien sous windows vous devriez obtenir ça :  
  
  
  
**2/ les IDE**  
  
Seulement voilà, si vous avez bien suivit ce que je vous ai dit et uniquement ce que je vous ai dit vous n'obtiendrez pas de resultat, et ce pour une raison simple, c'est que vous ne savez pas :  
1/ où taper ce code (code = le texte incompréhensible que je vous ai donné)  
2/ comment faire pour que ce texte qui ressemble vaguement à de l'anglais puisse être compris par un ordinateur, qui ne comprend que le binaire (10101011 11011110 00110110 par exemple).  
  
Comme je suppose que 99% d'entre vous sont sous windows (je ne saurait que trop vous recommander de **ne pas** continuer ainsi ...) je vais vous proposer uniquement les programmes windows qui permettent cela (pour le pourcent qui reste je vous conseille l'excellent[www.google.com](http://www.google.com) http://www.top-logiciel.net/uploads/smil3dbd4ddd6835f.gif ), ces programmes sont appelés IDE ou EDI (signifiant respectivement Integrated Developpment Environment et Environnement de Développement Intégré) :  
  
Premier choix : [Microsoft visual c++ 2005 express edition](http://www.top-logiciel.net/wfdownloads-singlefile.title-Visual_C_Express-cid-11-lid-251.htm) (l'édition express à pour avantage d'être gratuite), c'est un choix honorable si vous n'êtes ni amoureux du logiciel libre ni de la portabilité http://www.top-logiciel.net/uploads/smil3dbd4e398ff7b.gif (portabilité = fait pour un programme de pouvoir être utilisé sous différentes plateformes par simple recompilation du code, vous verrez bientôt ce qu'est la compilation), cependant pour un produit du microsoft il est de bonne qualité http://www.top-logiciel.net/uploads/smil3dbd4ddd6835f.gif .  
  
Second choix : [Dev-C++](http://www.top-logiciel.net/wfdownloads-singlefile.cid-11-lid-450.htm" \t "_blank), ce n'est pas le meilleur mais c'est simplement LA référence en la matière, il n'est pas assez mis a jour, c'est là son moindre défaut. Il a aussi pour avantage d'être libre (logiciel libre = logiciel souvent gratuit dont on peut obtenir le code source, contrairement aux logiciels proprietaires, le code source designe tout le code necessaire pour faire un programme).  
  
Dernier choix : [Code::Blocks](http://www.top-logiciel.net/wfdownloads-singlefile.cid-11-lid-449.htm" \t "_blank), c'est le meilleur, le plus "intelligent" et il est libre comme Dev-C++, il a de nombreux avantages que vous verrez au fur et a mesure de son utilisation, il peut entre autre utiliser le système de "plugins" de Dev-C++, il a donc les avantages de Dev sans ses inconvénients. La coloration syntaxique est d'après moi la meilleure chez Code::Blocks car elle est à la fois dense et esthétique. Code::Blocks est également disponible sur la plupart des plateformes.  
  
  
Donc choisissez un IDE (Code::Blocks de préférence, si vous en choisissez un autre, sachez que je n'expliquerait les manoeuvres que sur Code::Blocks a partir de maintenant) et créez un nouveau projet EN MODE CONSOLE ET EN C (et pas en C++).  
Sous Code::Blocks il faut aller dans "Files/New project", choisir "Console Application" et ne pas oublier de mettre "C source" dans "File(s) option" pour les autres DEBROUILLEZ-VOUS http://www.top-logiciel.net/uploads/smil3dbd4ddd6835f.gif , si vous êtes sous Dev cela devrait ressembler a la manoeuvre sous Code::Blocks http://www.top-logiciel.net/uploads/smil3dbd4e398ff7b.gif (profitez-en pour essayer de mieux connaître votre IDE).  
Ensuite vous récrirez le code dans l'éditeur qui se présente à vous (un bout de code est déjà écrit, sous Code::Blocks, et probablement sous les autres, il s'agit du Hello world, cependant Code::Blocks donne une version légèrement différente de la mienne ("Hello world!" au lieu de mon "Hello, world").  
  
**3/ Et c'est partit (enfin ... )**  
  
Maintenant nous allons transformer notre programme en un exécutable (fichier binaire compréhensible par l'ordinateur), cette étape s'appelle la compilation, puis nous allons le lancer, pour cela, au lieu de dire à notre IDE "compile moi ça !!!" puis "lance le fichier que tu vient de compiler !!!", on va utiliser la méthode dite "du flemmard" : dire à l'IDE "compile moi ça et lance-le ... ". Pour cela vous allez utiliser la fonction "build and run" qui sous Code::Blocks se situe dans "build", vous compilez et lancez donc votre programme et si vous n'avez pas pris Code::Blocks ... ça ne marche toujours pas http://www.top-logiciel.net/uploads/smil3dbd4ddd6835f.gif , mais pourquoi ? Et bien parce que quand windows exécute votre programme, il fait tout ce qu'il a à faire (afficher "Hello, world" dans notre cas), puis ferme la console, donc en réalité le programme marche mais on a pas le temps de le voir ... http://www.top-logiciel.net/uploads/smil3dbd4e398ff7b.gif  
  
Pour remédier a cela il y a plusieurs possibilités : le system("pause") et le getchar(), le premier vous propose d'appuyer sur une touche pour continuer et le second attend que vous appuyiez sur [Entrée] sauf qu'il ne vous le précise, malgré tout je vous conseille d'utiliser getchar() (de toute façon dans le prochain cours vous apprendrez à écrire "appuyez sur une touche" juste avant le getchar(), si vous n'avez pas déjà compris) car l'autre fonctoin n'est pas portabl (ne fonctionne que sous Windows).  
Voilà a quoi ressemble maintenant notre programme sous notre IDE (si vous êtes sous Code::Blocks ne mettez pas le getchar()) :  
  
  
  
Vous pouvez remarquer deux choses : d'abord les jolies couleurs, c'est l'IDE qui fait ça, c'est plus facile de se retrouver dans un code comme ça, ensuite le ";" derrière chaque instruction, ceci est obligatoire en langage C pour la simple raison qu'on aurait pu écrire :   
  
http://www.top-logiciel.net/uploads/img448ad7889d05f.png  
  
ATTENTION !!! Ce n'est pas parce que c'est possible que vous pouvez le faire (vous en avez la possibilité mais pas le droit), je vous interdit d'ailleurs formellement cette écriture qui serait plus qu'illisible sur un programme de plus de dix lignes, il existe d'autres styles de programmation que celui que je vais utiliser ici (regardez [ici](http://en.wikipedia.org/wiki/Indent_style), je vous conseille les 4 premiers styles, les plus utilisés sont le K&R et le BSD/Allman, personnellement je trouve que le style GNU est le plus beau mais pour l'utiliser il est préférable d'avoir l'éditeur[Emacs](http://www.gnu.org/software/emacs/)) mais celui la je vous l'interdit franchement.  
  
**4/ Problème**  
  
Vous avez peut-être compris quelle partie du programme permettait d'afficher du texte dans la console et vous vous êtes peut-être dit " http://www.top-logiciel.net/uploads/smil3dbd4e7853679.gif ben, et si j'essayai d'écrire autre chose dans la console", vous avez eu raison, on ne progresse vraiment qu'en écrivant du code, en ayant des problèmes, en les résolvant (meme si pour cela on doit parfois demander de l'aide), et bien si vous avez voulu écrire du texte en français, vous êtes peut-être tombé sur votre premier problème : vous vouliez écrire "J'ai programmé ce truc tout seul." et, comble de l'horreur, vous vous êtes retrouvé avec "?" au lieu d'un "é" http://www.top-logiciel.net/uploads/smil3dbd4d8676346.gif (ou autre chose, c'est ce que j'ai eu personnellement) la raison est simple : la console de windows n'affiche pas les caractères spéciaux (j'ai essayé le plus de caractères spéciaux possible et j'ai remarqué qu'un des seuls qu'elle affichait bien était le "$", c'est microsoft après tout http://www.top-logiciel.net/uploads/smil3dbd4d4e4c4f2.gif ), les solutions sont simples aussi, il y en a deux principales :  
- Passer sous un autre système que windows (Linux).  
- Ne pas mettre les accents, cette solution me paraît meilleure pour le moment ^^.  
  
**5/ Dernière petite chose**  
  
Enfin dernière chose que nous allons apprendre : ce sont les commentaires. Premièrement : les commentaires à quoi ça sert ? Eh bien on pourrait considérer que ça ne sert à rien puisque votre programme sera, après compilation, exactement le même que vous aillez mis des commentaires ou non, en fait le compilateur ne "lit" pas vos commentaires, il passe tout bonnement a cote, cependant, vous ou un autre programmeur peut les lire, en fait les commentaires servent à dire au programmeur "là, on écrit le texte dans le fichier" ou "là, on affiche 'Hello, world'", vous me suivez ? En fait on se retrouve mieux dans un code grâce aux commentaires. Venons en au fait : comment on met ces commentaires qui ont l'air si formidables ? Et bien il y a deux méthodes : la méthode C et la méthode C++, ces deux méthodes fonctionnent sur les compilateurs C modernes. La méthode C consiste à encadrer son commentaire avec /\* et \*/ : /\* Je suis un commentaire \*/, la méthode C++ consiste à commencer son commentaire avec// et à le terminer avec un retour a la ligne (en gros votre commentaire commence avec // et se continue jusqu'à la fin de la ligne), perso je préfère la méthode dite "C++" car elle est plus simple et que de toute façon vous aurez rarement besoin de commentaires de plusieurs lignes.  
  
  
  
Que constatons nous ici les enfants ? On constate que commentaire C qui avait l'air si fort a oublié de se fermer, ce qui a causé un terrible bug lors de la compilation (Oh ! non ! http://www.top-logiciel.net/uploads/smil3dbd4ddd6835f.gif )  
  
**Conclusion**  
  
Voilà, c'est la fin de ce premier tuto de C, j'espère qu'il vous a plu, sachez que j'essayerais d'en faire une (longue) série dans laquelle vous apprendrez, en plus des bases, certaines librairies essentielles comme les librairies réseaux : les chaussettes (vous comprendrez pourquoi ce nom plus tard) et peut-être même une librairie graphique (GTK+ ?).