

Allegro 4.4.3

Procédure d'installation pour Windows & Code::Blocks

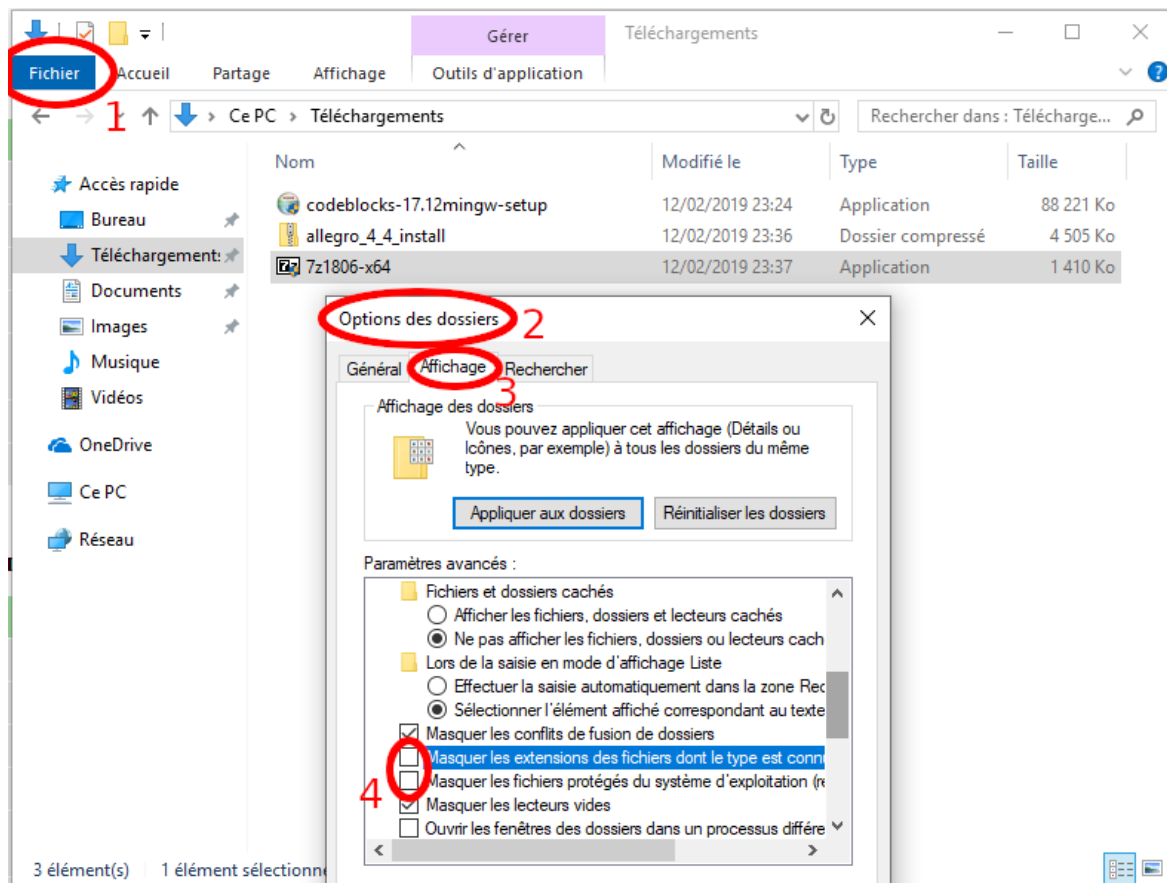
Ce tutoriel vous guide pas à pas dans l'installation du pack pré-compilé **allegro_4_4_3_install.zip** permettant de créer vos exécutables graphiques en C ou C++ dans l'IDE Code::Blocks.

Prérequis

- **Un système Windows 10 opérationnel**
→ *Sur machine Apple vous pouvez installer une machine virtuelle Windows dans votre macOS, par exemple avec [VirtualBox](#). Dans ce cas prenez le temps de paramétrer votre machine virtuelle pour qu'elle fonctionne bien. Vous avez droit à un système Windows gratuit en tant qu'étudiant, sinon il est possible (et légal) de télécharger une image Windows.iso depuis un Windows (à transférer ensuite par exemple avec une clé USB) à partir de cette adresse <https://www.microsoft.com/fr-fr/software-download/windows10> « Vous souhaitez installer Windows 10 sur votre PC ? » « [Télécharger maintenant l'outil](#) » et en indiquant qu'on souhaite une installation sur une autre machine pour obtenir un fichier iso (environ 4Go). Pendant l'installation de Windows en tant que machine virtuelle cliquer sur suivant quand il est question de licence : le Windows ne sera pas enregistré mais il sera fonctionnel (Microsoft vous laisse faire, ce n'est pas un accident).*
- **Code::Blocks** avec le compilateur MinGW, c'est le cas si vous pouvez déjà compiler vos propres programmes en mode console ! Pour rappel ceci correspond au 4^{ème} lien : **codeblocks-20.03mingw-setup.exe** sur <http://www.codeblocks.org/downloads/26>
- **7-Zip** utilitaire très utile pour gérer les fichiers archives (zip, rar etc...) à installer depuis <https://www.7-zip.org/download.html> (2^{ème} lien Download)
Toujours télécharger les applications Windows depuis des liens directement sur les sites des éditeurs pour éviter les mauvaises surprises (adwares etc...)

Recommandation initiale : voir les extensions des fichiers

Pour rappel il est conseillé en tant qu'utilisateur "expert" de Windows de configurer l'explorateur de fichiers pour bien voir les extensions (.zip .jpg .exe etc.) **Voir capture écran page suivante** : dans une fenêtre d'explorateur de fichiers faire menu Fichier -> Options -> onglet Affichage dans la liste Paramètres avancés : décocher les 2 cases de "Masquer les extensions des fichiers dont le type est connu" et "Masquer les fichiers protégés du système d'exploitation (recommandé)" Cette dernière manip rend Windows paranoïaque : confirmer que vous voulez bien voir tout ça.

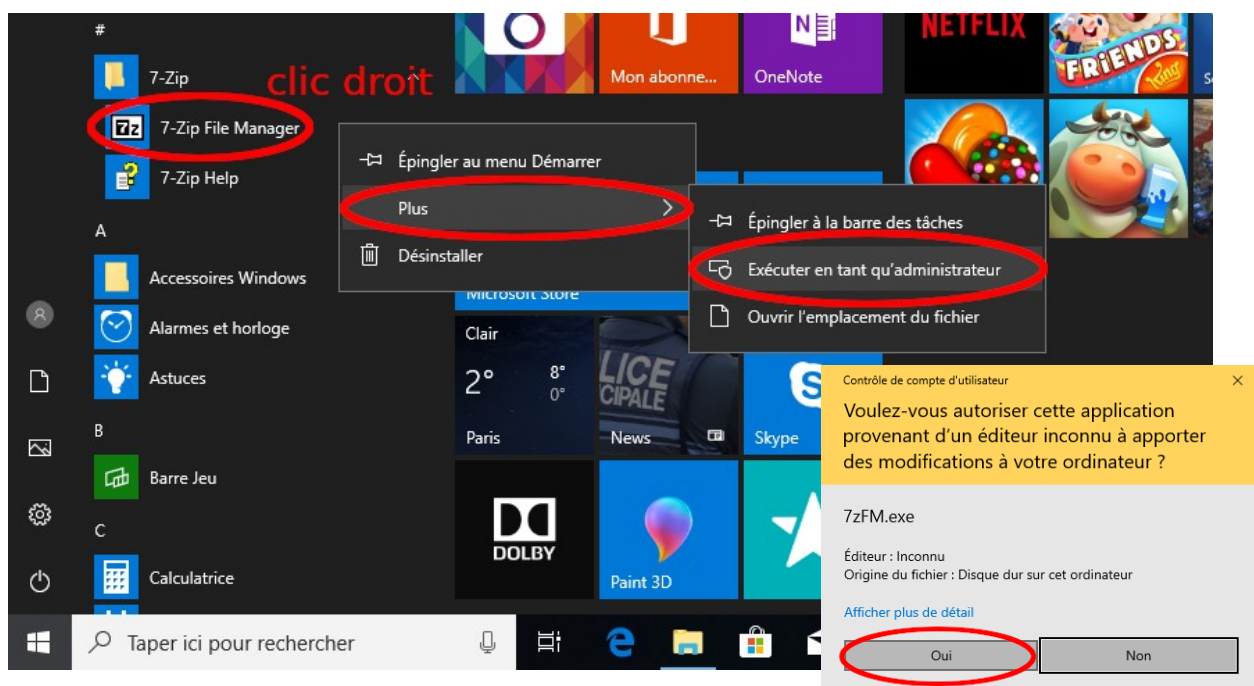


Télécharger le pack de développement : allegro_4_4_3_install.zip

C'est ici que ça se passe :

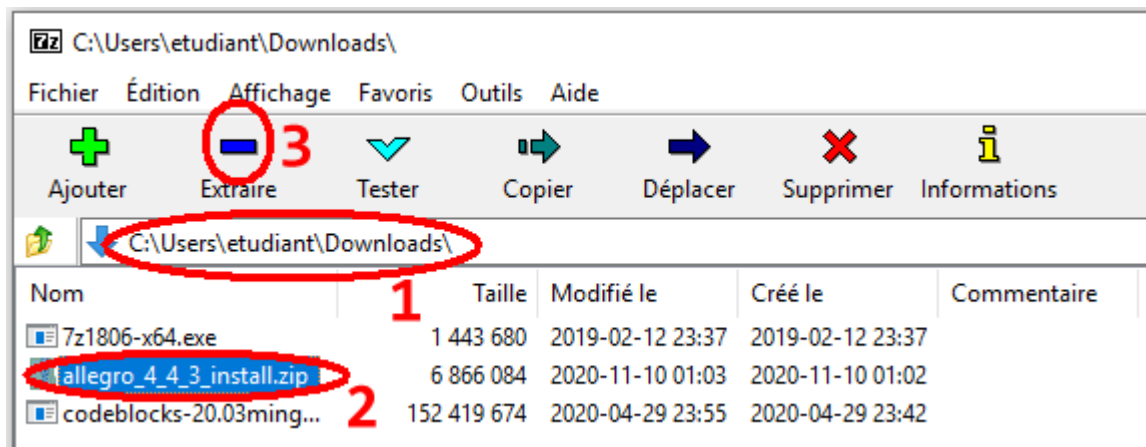
<https://fercoq.bitbucket.io/allegro/index.html#Installation>

Lancer 7-Zip en mode administrateur

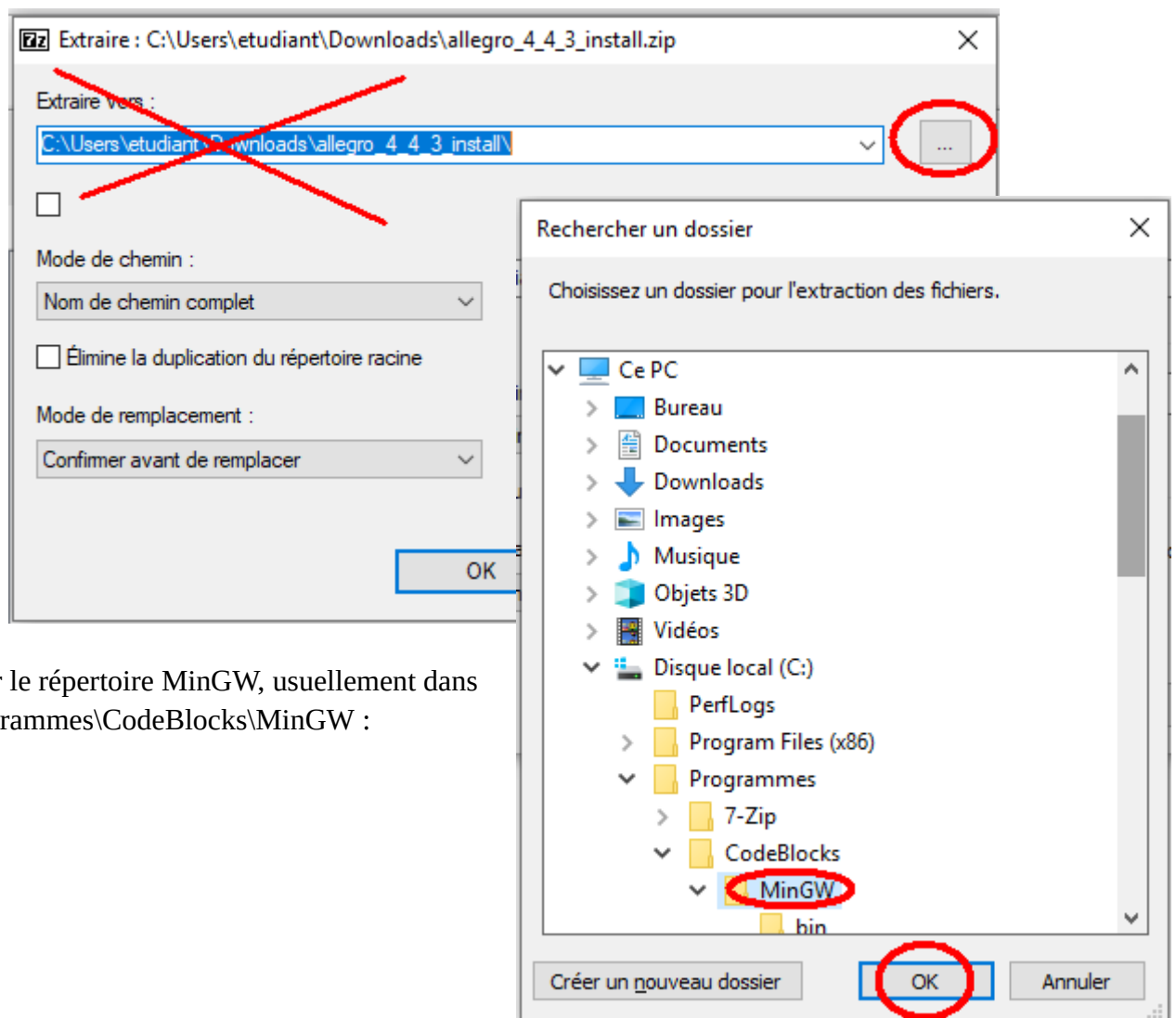


Dans 7-Zip en mode administrateur décompresser le zip vers MinGW

Commencer par trouver le zip, en général dans C:\Users\VotreNomUtilisateur\Downloads, le sélectionner puis cliquer sur Extraire...

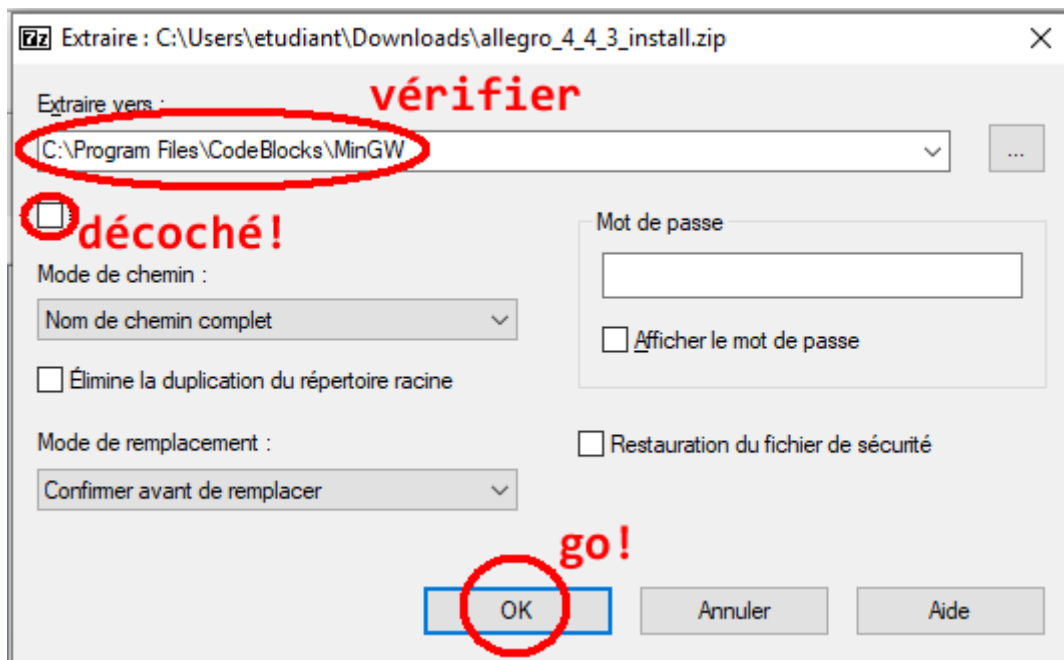


Le répertoire de destination de l'extraction proposé ne convient pas, cliquer sur le bouton avec 3 petits points [...] à droite pour naviguer dans le système de fichier :



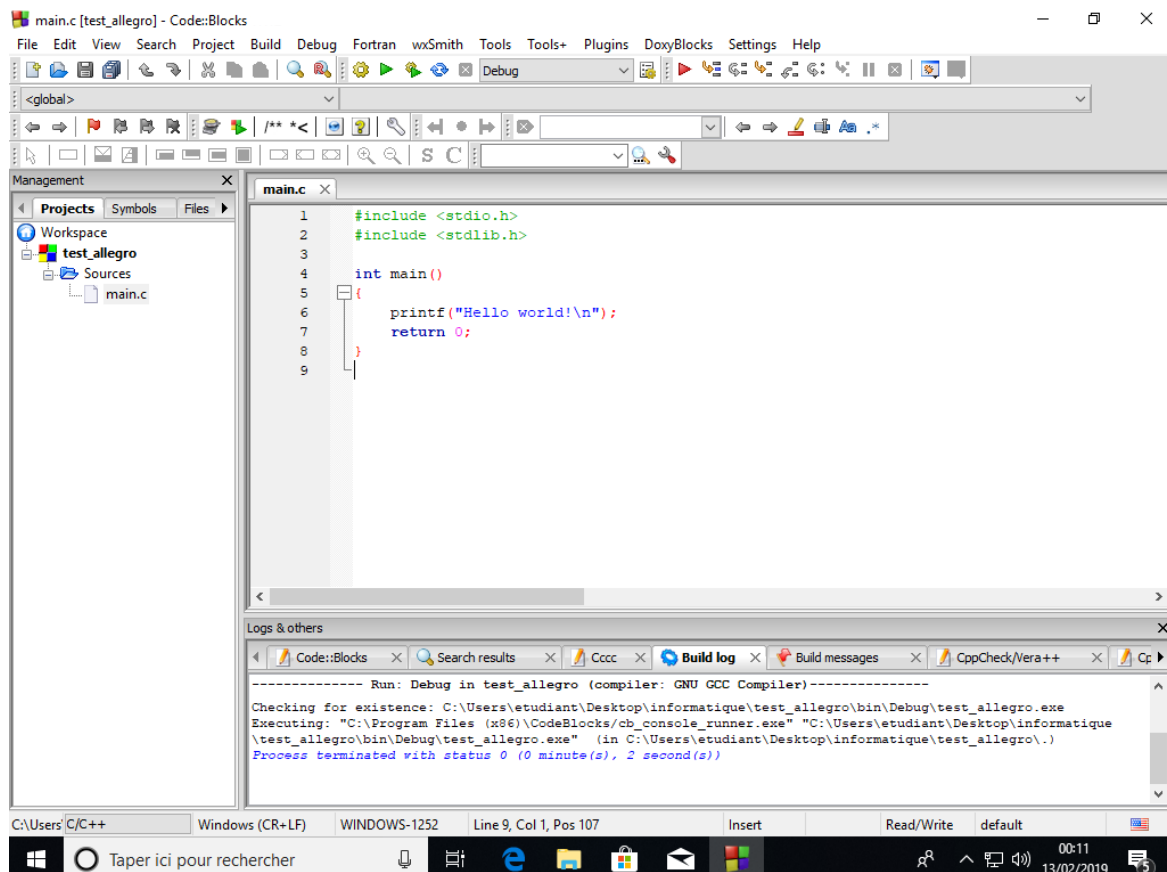
Trouver le répertoire MinGW, usuellement dans C:\Programmes\CodeBlocks\MinGW :

On a bien indiqué à 7-Zip le répertoire MinGW (le compilateur) comme répertoire de destination, vérifier que **le nom du zip comme préfixe est bien décoché**, puis cliquer OK, l'opération d'installation devrait se dérouler sans incident (confirmer écraser fichier si demande) :



Tester une compilation d'un programme Allegro en utilisant un code source du cours

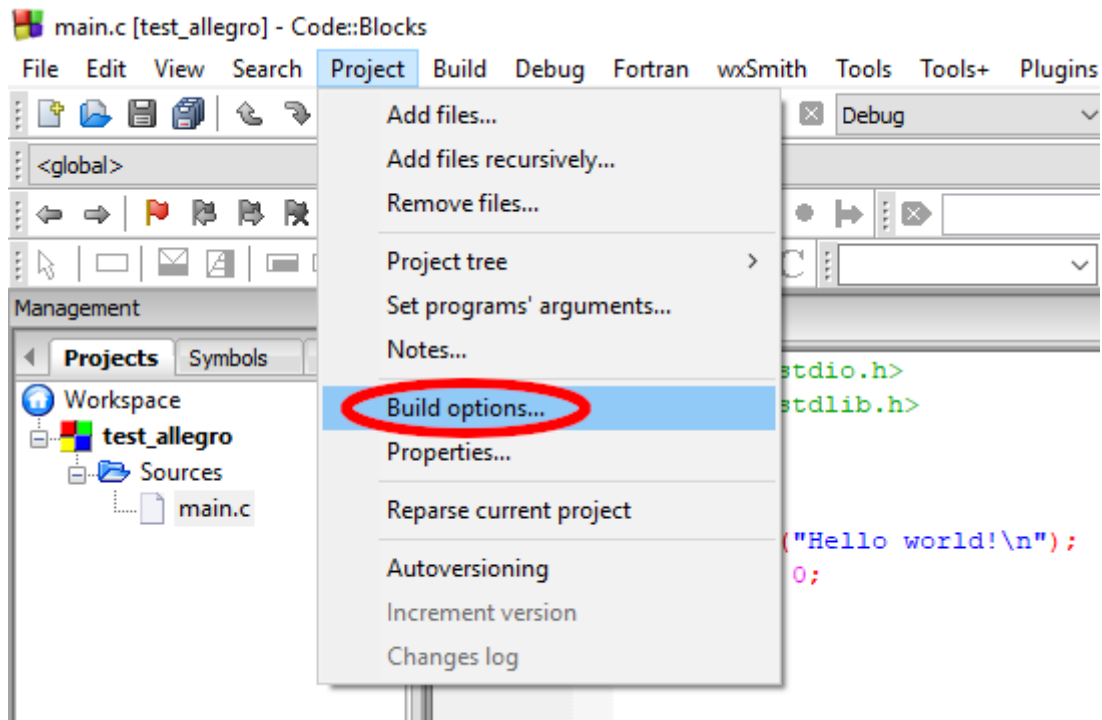
Lancer Code::Blocks et créer un nouveau projet en « Console application » (comme d'habitude !)



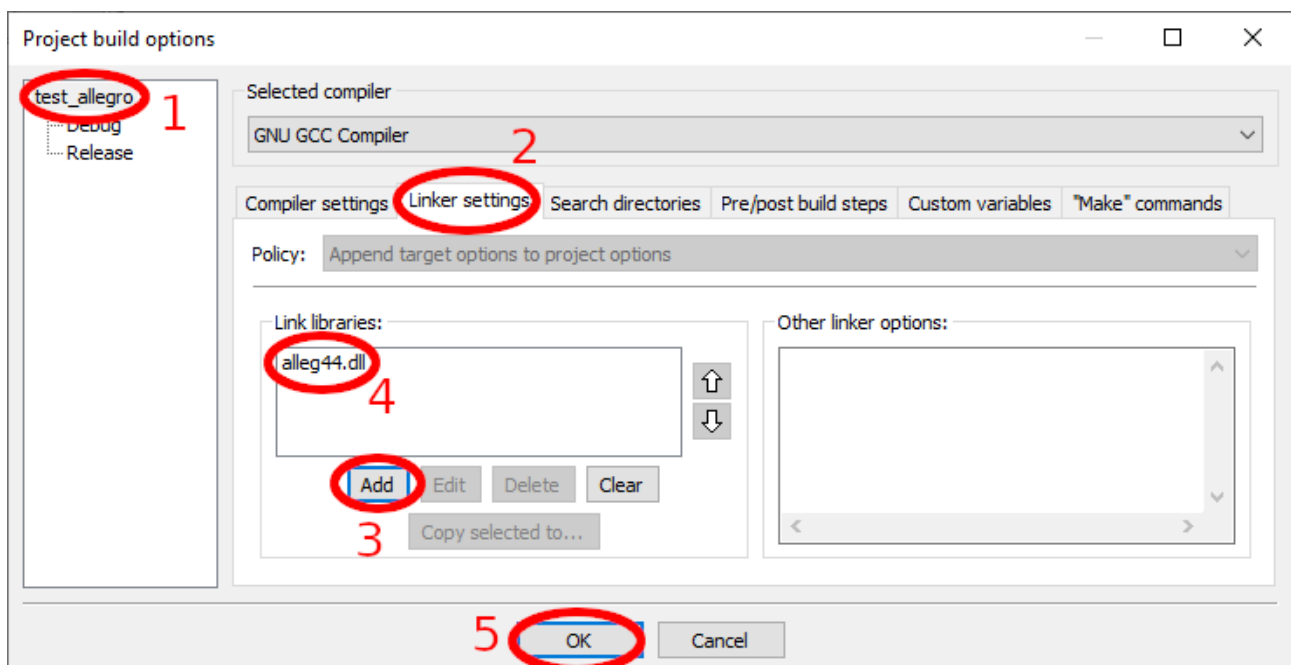
Vérifier que ce projet console compile et fonctionne (on obtient bien « Hello world! » en console).

Pour compiler un programme graphique Allegro il faudra préciser au compilateur que notre exécutable utilise une bibliothèque (library) en l'occurrence la bibliothèque à liaison dynamique (dynamic link library = dll) **alleg44.dll** qui a été installée dans le système de compilation.

Aller dans les options de compilation, menu Projet => Build options...

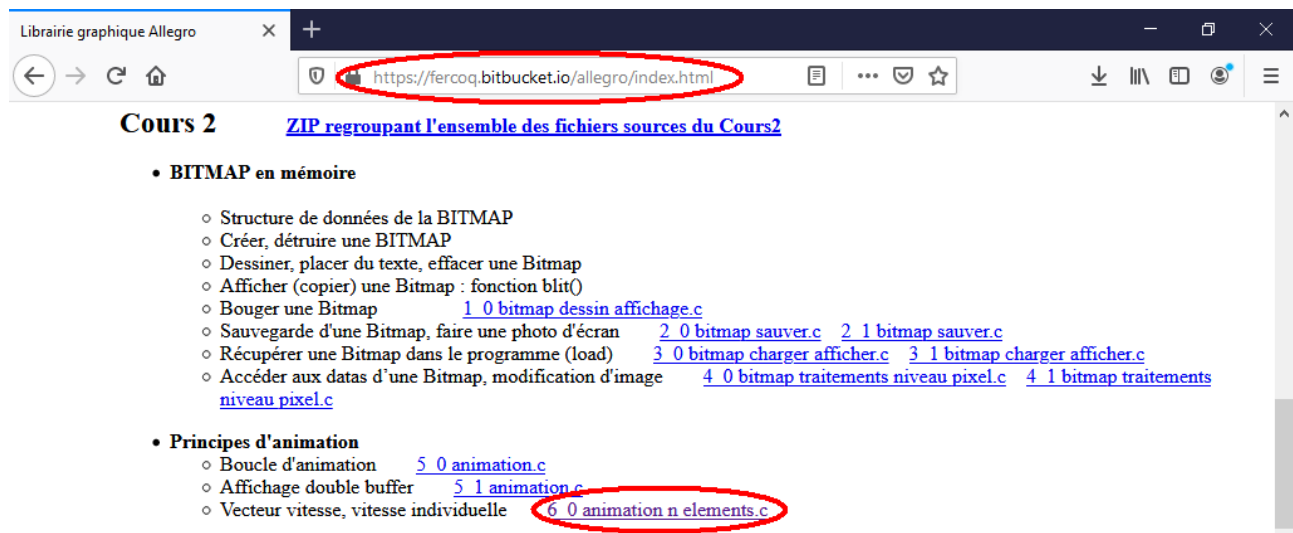


Suivre la séquence ...



Cette manipulation devra être faite pour chaque nouveau projet Allegro. L'alternative serait d'indiquer alleg44.dll dans les options globales de Code::Blocks mais ceci me semble une fausse bonne idée : en effet tous les projets que nous allons compiler ne sont pas graphiques et/ou ne dépendent pas d'Allegro.

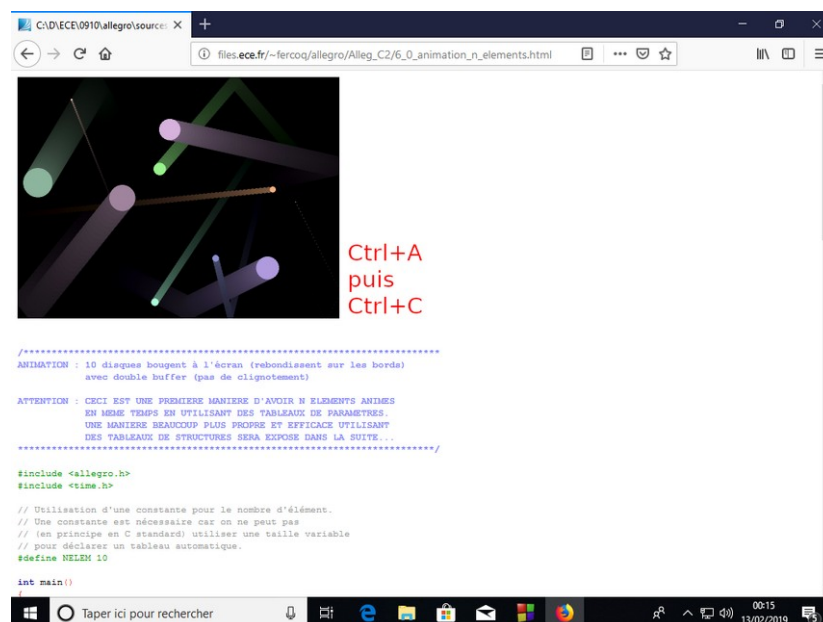
Pour tester un programme Allegro sur la base d'un code source fiable, aller sur le site avec les exemples du cours et choisir un exemple de code sans fichier graphique (*c'est plus simple, sinon il faut avoir le fichier graphique à disposition => ZIP regroupant l'ensemble des fichiers sources...*) Par exemple « [6 0 animation n elements](#) » du cours 2 :



Cours 2 [ZIP regroupant l'ensemble des fichiers sources du Cours2](#)

- **BITMAP en mémoire**
 - Structure de données de la BITMAP
 - Créer, détruire une BITMAP
 - Dessiner, placer du texte, effacer une Bitmap
 - Afficher (copier) une Bitmap : fonction blit()
 - Bouger une Bitmap [1 0 bitmap dessin affichage.c](#)
 - Sauvegarde d'une Bitmap, faire une photo d'écran [2 0 bitmap sauver.c](#) [2 1 bitmap sauver.c](#)
 - Récupérer une Bitmap dans le programme (load) [3 0 bitmap charger afficher.c](#) [3 1 bitmap charger afficher.c](#)
 - Accéder aux datas d'une Bitmap, modification d'image [4 0 bitmap traitements niveau pixel.c](#) [4 1 bitmap traitements niveau pixel.c](#)
- **Principes d'animation**
 - Boucle d'animation [5 0 animation.c](#)
 - Affichage double buffer [5 1 animation.c](#)
 - Vecteur vitesse, vitesse individuelle [6 0 animation n elements.c](#)

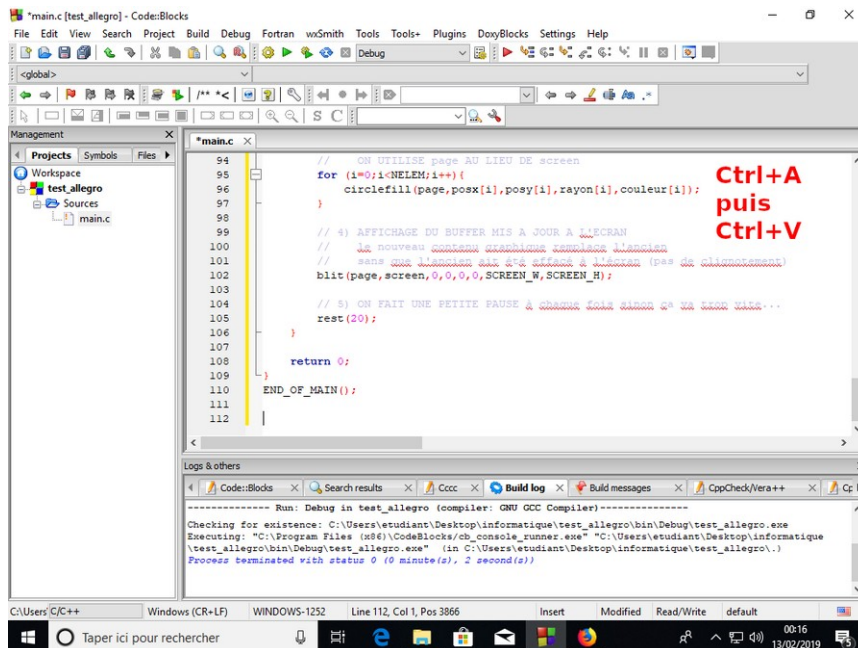
Une fois sur un exemple de code source, **prendre le code source en faisant Ctrl+A (select All, tout sélectionner) puis Ctrl+C (copier).**



Ctrl+A
puis
Ctrl+C

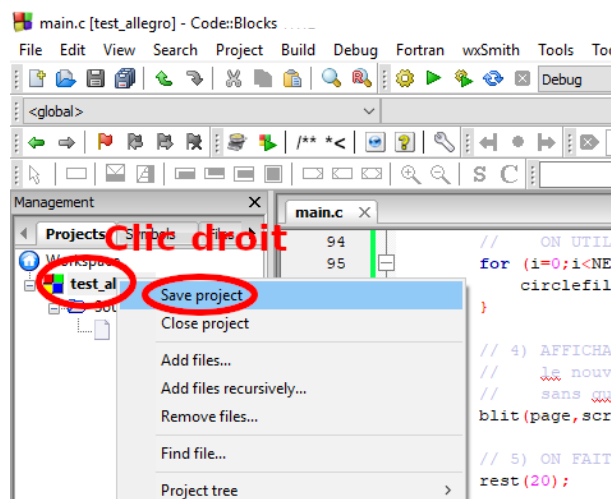
```
.....  
ANIMATION : 10 disques bougent à l'écran (rebondissent sur les bords)  
avec double buffer (pas de clignotement)  
  
ATTENTION : CECI EST UNE PREMIERE MANIERE D'AVOIR N ELEMENTS ANIMES  
EN MEME TEMPS EN UTILISANT DES TABLEAUX DE PARAMETRES.  
UNE MANIERE BEAUCOUP PLUS PROPRE ET EFFICACE UTILISANT  
DES TABLEAUX DE STRUCTURES SERA EXPOSE DANS LA SUITE...  
.....  
  
#include <allegro.h>  
#include <time.h>  
  
// Utilisation d'une constante pour le nombre d'éléments.  
// Une constante est nécessaire car on ne peut pas  
// (en principe en C standard) utiliser une taille variable  
// pour déclarer un tableau automatique.  
#define NLELEN 10  
  
int main()  
{  
    // Taper ici pour rechercher
```

De retour dans Code::Blocks, cliquer sur le code source du main.c et **remplacer tout par le code de l'exemple du cours en faisant Ctrl+A puis Ctrl+V**



Enfin compiler et exécuter le projet (Build and run), si tout s'est bien passé vous devez avoir une animation conforme à la vignette en haut de la page d'exemple du cours (ici des disques de couleur qui bougent dans un cadre sur fond noir). *Les « traînées de vitesse » ne font pas partie du programme, c'est un effet qui a été ajouté lors de la capture pour illustrer le mouvement des objets.*

Pour conserver les options de compilation du projet (alleg44.dll etc...) n'oubliez pas de le sauvegarder :



Si tout ne se passe pas comme prévu, pas de panique, demandez autour de vous, et sollicitez votre chargé de TP si nécessaire.

Dans [allegro checking.zip](#) vous avez des projets montrant l'utilisation d'extensions de la bibliothèque, en particulier le son compressé (format ogg vorbis) et les **images .png et .jpg**. Regardez attentivement les options du linker (Linker settings) à ajouter à alleg44.dll pour utiliser ces extensions dans votre projet (attention, l'ordre doit être respecté).

Bonne programmation graphique !