



CPPLI : TD 0 : Qt Creator

Romain Absil Jonas Beleho Pierre Bettens
David Hauweele Pierre Hauweele
Nicolas Vansteenkiste * (ESI – HE2B)

Année académique 2021 – 2022

Ce TD¹ décrit quelques manipulations récurrentes lors de l'utilisation de l'environnement de développement intégré Qt Creator.

Table des matières

1	Portabilité	2
2	Création d'un projet C	2
2.1	C99	3
2.2	C11	3
2.3	C17	3
3	Création d'un projet C++	4
3.1	C++17	4
3.2	C++20	5
4	Ajout d'un fichier à un projet	5
4.1	Nouveau fichier	5
4.1.1	Fichier d'en-têtes	5

*Et aussi, lors des années passées : Monica Bastregghi, Stéphan Monbaliu, Anne Rousseau et Moussa Wahid.

1. https://poesi.esi-bru.be/pluginfile.php/7311/mod_folder/content/0/td00_c_cpp/td00_c_cpp.pdf (consulté le 13 septembre 2021).

4.1.2	Fichier source	5
4.2	Fichier existant	6
4.2.1	Fichier d'en-têtes ou fichier source	6
5	Production, test et débogage d'un exécutable	6
5.1	Production	6
5.2	Exécution	6
5.3	Débogage	7
6	Alternance de travail à l'école et ailleurs	7
7	Plugins à activer (éventuellement)	7
7.1	Beautififier	8
7.2	Todo	8

1 Portabilité

Les binaires exécutables ou objet produits à partir de codes C ou C++ ne sont pas portables d'un système d'exploitation à l'autre, voire d'un compilateur à l'autre sur un même système d'exploitation. Par contre, les sources peuvent être portables. Pour ce faire, il faut, au moins :

- respecter la casse² dans les noms de fichiers³ ;
- utiliser le séparateur « / »⁴ entre les répertoires dans les chemins d'accès lors des inclusions de fichiers d'en-têtes.

2 Création d'un projet C

Démarrez Qt Creator⁵.

Pour créer un projet C :

- Choisissez dans la barre de menu l'entrée *File* et dans celle-ci *New File or Project...*
- Dans la boîte de dialogue ouverte par ce choix, cliquez *Non-Qt Project* dans le panneau de gauche. Par défaut, *Plain C Project* est sélectionné, cela nous convient. Appuyez sur le bouton *Choose...* en bas à droite.
- Donnez le nom de votre choix au projet, par exemple *xXx*, et créez-le soit en local sur votre bureau, soit quelque part dans votre espace disque personnel *Z:*. Cliquez *Next >*.

2. [https://fr.wikipedia.org/wiki/Casse_\(typographie\)](https://fr.wikipedia.org/wiki/Casse_(typographie)) (consulté le 13 septembre 2021).

3. <https://stackoverflow.com/q/1951969> (consulté le 13 septembre 2021).

4. <https://stackoverflow.com/q/5790202> (consulté le 13 septembre 2021).

5. <https://doc.qt.io/qtcreator> (consulté le 13 septembre 2021).

(d) À l'écran *Kit Selection*, cliquez *Next* > sans rien modifier si le nom du *kit* en question se termine par « MinGW 64 bits », sinon sélectionnez le *kit* « Desktop Qt 6.x.y MinGW 64 bits », où x et y sont des composantes du numéro de version.

(e) À l'écran *Project Management*, cliquez *Finish* sans rien modifier.

Le projet est alors créé et ouvert en mode d'édition.

2.1 C99

Pour [configurer le projet](#)⁶ de sorte que le compilateur travaille avec le standard [C99](#)⁷, voici ce qu'il faut faire :

(a) Ouvrez le fichier `xXx.pro`.

(b) Ajoutez-lui les lignes :

```
QMAKE_CFLAGS += -std=c99 \  
               -pedantic-errors \  
               -D__USE_MINGW_ANSI_STDIO
```

où le dernier *flag* ne doit être utilisé que si vous travaillez sous *MS-Windows*.

2.2 C11

Pour configurer le projet de sorte que le [compilateur](#)⁸ travaille avec le standard [C11](#)⁹, voici ce qu'il faut faire :

(a) Ouvrez le fichier `xXx.pro`.

(b) Ajoutez-lui les lignes :

```
QMAKE_CFLAGS += -std=c11 \  
               -pedantic-errors \  
               -D__USE_MINGW_ANSI_STDIO
```

où le dernier *flag* ne doit être utilisé que si vous travaillez sous *MS-Windows*.

2.3 C17

Pour configurer le projet de sorte que le compilateur travaille avec le standard [C17](#)¹⁰, voici ce qu'il faut faire :

(a) Ouvrez le fichier `xXx.pro`.

(b) Ajoutez-lui les lignes :

6. <https://stackoverflow.com/q/18520817> (consulté le 13 septembre 2021).
7. <https://en.wikipedia.org/wiki/C99> (consulté le 13 septembre 2021).
8. <https://gcc.gnu.org/onlinedocs/gcc-11.2.0/gcc/C-Dialect-Options.html#C-Dialect-Options> (consulté le 13 septembre 2021).
9. [https://en.wikipedia.org/wiki/C11_\(C_standard_revision\)](https://en.wikipedia.org/wiki/C11_(C_standard_revision)) (consulté le 13 septembre 2021).
10. [https://en.wikipedia.org/wiki/C17_\(C_standard_revision\)](https://en.wikipedia.org/wiki/C17_(C_standard_revision)) (consulté le 13 septembre 2021).

```
QMAKE_CFLAGS += -std=c17 \  
                -pedantic-errors \  
                -D__USE_MINGW_ANSI_STDIO
```

où le dernier *flag* ne doit être utilisé que si vous travaillez sous *MS-Windows*.

3 Création d'un projet C++

Démarrez [Qt Creator](#)¹¹.

Pour créer un projet C++ :

- (a) Choisissez dans la barre de menu l'entrée *File* et dans celle-ci *New File or Project...*
- (b) Dans la boîte de dialogue ouverte par ce choix, cliquez *Non-Qt Project* dans le panneau de gauche. Par défaut, *Plain C Project* est sélectionné, cela ne nous convient pas. Changez la sélection en *Plain C++ Project* puis appuyez sur le bouton *Choose...* en bas à droite.
- (c) Donnez le nom de votre choix au projet, par exemple *yYy*, et créez-le soit en local sur votre bureau, soit quelque part dans votre espace disque personnel *Z:*. Cliquez *Next >*.
- (d) À l'écran *Kit Selection*, cliquez *Next >* sans rien modifier si le nom du *kit* en question se termine par « MinGW 64 bits », sinon sélectionnez le *kit* « Desktop Qt 6.x.y MinGW 64 bits », où *x* et *y* sont des composantes du numéro de version.
- (e) À l'écran *Project Management*, cliquez *Finish* sans rien modifier.

Le projet est alors créé et ouvert en mode d'édition.

3.1 C++17

Pour configurer le projet de sorte que le [compilateur](#)¹² travaille avec le standard [C++17](#)¹³, voici ce qu'il faut faire :

- (a) Ouvrez le fichier *yYy.pro*.
- (b) Remplacez-y la ligne :

```
CONFIG += console c++11
```

par :

```
CONFIG += console # c++11
```

- (c) Ajoutez-lui la ligne :

```
QMAKE_CXXFLAGS += -std=c++17 -pedantic-errors
```

11. https://wiki.qt.io/Qt_Creator (consulté le 13 septembre 2021).

12. <https://gcc.gnu.org/projects/cxx-status.html#cxx17> (consulté le 13 septembre 2021).

13. <https://en.wikipedia.org/wiki/C%2B%2B17> (consulté le 13 septembre 2021).

Remarque Malgré la configuration décrite ci-dessus, on observe que l'option de compilation `-std=gnu++1z` est automatiquement ajoutée. Le standard C++17 n'est donc pas strictement respecté mais des [extensions GNU](#)¹⁴ lui sont ajoutées.

3.2 C++20

Au moment de la rédaction de ce TD, la [version standardisée ISO du C++](#)¹⁵ est le C++20¹⁶. Ce standard est très bien [pris en charge](#)¹⁷ par le compilateur `gcc`¹⁸. Malheureusement, la version de `gcc` installée à l'école est celle de numéro 8.1.0. Pour celle-ci la prise en charge du C++20 reste très limitée. Voilà pourquoi nous travaillons, dans le cadre des laboratoires C++, en C++17.

4 Ajout d'un fichier à un projet

4.1 Nouveau fichier

4.1.1 Fichier d'en-têtes

Prenons le projet C ***xXx*** créé à la section 2. Pour lui ajouter un fichier d'en-têtes inexistant au préalable, procédez de la sorte :

- Choisissez dans la barre de menu l'item *File* et dans celui-ci *New File or Project...*
- Dans le panneau de gauche de la fenêtre qui s'ouvre alors, sous *Files and Classes*, sélectionnez *C++*, puis *C++ Header File* dans le panneau central et cliquez *Choose...*
- Fournissez le nom de votre choix au fichier d'en-têtes, ***test***¹⁹, par exemple.

Le fichier d'en-têtes ***test.h*** est alors créé, ajouté au projet courant — pour vous en convaincre, allez voir le fichier ***xXx.pro*** — et ouvert dans l'éditeur.

Projet C++ Pour ajouter un nouveau fichier d'en-têtes à un projet C++, tel ***yYy*** de la section 3, il faut procéder exactement de la même manière, à savoir passer par la création d'un nouveau *C++ Header File*.

4.1.2 Fichier source

Continuons avec le projet C ***xXx***. Pour lui ajouter un fichier source inexistant au préalable, procédez de la sorte :

14. https://gcc.gnu.org/onlinedocs/gcc/C_002b_002b-Extensions.html (consulté le 13 septembre 2021).

15. <https://isocpp.org/std/status> (consulté le 13 septembre 2021).

16. <https://en.wikipedia.org/wiki/C%2B%2B20> (consulté le 13 septembre 2021).

17. https://en.cppreference.com/w/cpp/compiler_support#cpp20 (consulté le 13 septembre 2021).

18. <https://gcc.gnu.org/projects/cxx-status.html#cxx20> (consulté le 13 septembre 2021).

19. Ou ***test.h***.

- (a) Choisissez dans la barre de menu l'item *File* et dans celui-ci *New File or Project...*
- (b) Dans le panneau de gauche de la fenêtre qui s'ouvre alors, sous *Files and Classes*, sélectionnez *C++*, puis *C++ Source File* dans le panneau central et ensuite cliquez *Choose...*
- (c) Fournissez le nom de votre choix au fichier source, `test.c`²⁰, par exemple.

Le fichier d'en-têtes `test.c` est alors créé, ajouté au projet courant — pour vous en convaincre, allez voir le fichier `xXx.pro` — et ouvert dans l'éditeur.

Projet C++ Pour ajouter un nouveau fichier source à un projet C++, il faut procéder semblablement, à savoir passer par la création d'un nouveau *C++ Source File*. Lors du nommage du nouveau fichier, vous pouvez omettre l'extension `.cpp`, car, par défaut, un fichier source d'extension `.cpp` est créé par Qt Creator.

4.2 Fichier existant

4.2.1 Fichier d'en-têtes ou fichier source

Pour ajouter un fichier d'en-têtes ou un fichier source existant à un projet C ou C++, voici la marche à suivre :

- (a) Affichez la vue *Activate Projects* en sélectionnant l'entrée *Projects* dans la *boîte combinée*²¹ (*combobox*) en haut du panneau à droite des boutons *Welcome*, *Edit*, *Design*, etc., ou, plus simplement, en frappant les touches **Alt-x**.
- (b) Cliquez avec le bouton droit sur le nom du projet auquel vous voulez ajouter le fichier et sélectionnez *Add Existing Files...* dans le menu contextuel surgissant.
- (c) Naviguez jusqu'à l'emplacement du fichier et cliquez *Open* en bas à droite de la boîte de dialogue.

5 Production, test et débogage d'un exécutable

5.1 Production

Pour compiler les fichiers sources du projet `xXx`, puis éditer les liens des fichiers objets produits en vue de générer un exécutable, cliquez *Build Project "xXx"* dans le menu *Build*, après avoir sauvegardé tous les fichiers sources, d'en-têtes, etc.

5.2 Exécution

Pour tester l'exécutable produit, cliquez *Run* dans le même menu *Build*.

20. N'oubliez surtout pas l'extension `.c` !

21. https://fr.wikipedia.org/wiki/Bo%C3%A0ete_combin%C3%A9e (consulté le 13 septembre 2021).

5.3 Débogage

Pour déboguer l'exécutable produit :

- (a) Placez — éventuellement ! — d'abord un point d'arrêt sur une ligne du code source. Pour cela, éditez le fichier source correspondant et cliquez dans la rigole à gauche de la ligne désirée : un gros point rouge avec un sablier devrait apparaître.
- (b) Cliquez *Start Debugging* dans le sous-menu *Start Debugging* du menu *Debug*.
- (c) Plusieurs scénarios sont possibles :
 - (i) L'application termine proprement sans atteindre le point d'arrêt : le débogage finit lui aussi proprement.
 - (ii) L'application plante avant d'atteindre le point d'arrêt : dans ce cas, la ligne problématique est pointée dans la perspective *Debug* et la vue *Stack* renseigne la *stack trace*²².
 - (iii) Le point d'arrêt est atteint : la perspective *Debug* indique une multitude d'informations, utilisez *Continue*, *Step Over*, *Step Into* et *Step Out* à bon escient.

6 Alternance de travail à l'école et ailleurs

Pour travailler, par exemple, chez vous sur un projet commencé à l'école, ou l'inverse, il faut d'abord avoir bien sauvegardé le(s) répertoire(s) contenant tous les fichiers du projet. Pour retrouver votre projet, utilisez l'entrée *Open File or Project...* du menu *File* et choisissez dans la fenêtre de sélection de fichier qui s'ouvre alors le fichier d'extension *.pro* correspondant au projet.

Il se peut qu'à l'ouverture du projet, une boîte de dialogue de titre *Settings File for 'xXx' from a different Environment ?*, où xXx est le nom du projet, apparaisse. Le plus simple dans ce cas est de cliquer *Yes* et d'essayer *Build* \Rightarrow *Build Project "xXx"*. Si cela échoue, tentez *Build* \Rightarrow *Run qmake* puis *Build* \Rightarrow *Build Project "xXx"*. Si cela ne fonctionne toujours pas, vous devez configurer explicitement le projet pour la nouvelle machine en commençant par cliquer sur le bouton *Projects*, dans la colonne de gauche de l'interface de Qt Creator, entre les boutons *Debug* et *Analyze*.

7 Plugins à activer (éventuellement)

Pour activer un *plugin* inactif de Qt Creator, il faut commencer par afficher l'état des *plugins* en sélectionnant l'entrée *About Plugins...* du menu *Help*. Une fois la demande d'activation du *plugin* réalisée en cochant sa case *Load*, il faut redémarrer Qt Creator pour la rendre effective.

22. https://en.wikipedia.org/wiki/Stack_trace (consulté le 13 septembre 2021).

7.1 Beautifier

Le *plugin* Beautifier utilise un logiciel tiers pour mettre en forme les codes sources. Il est rangé parmi les *plugins* C++.

À l'école, c'est le logiciel libre [Artistic Style](#)²³ qui est utilisé par Beautifier. Voici comment configurer ce *plugin* :

- (a) dans le menu *Tools*, choisissez *Options...*
- (b) sélectionnez *Beautifier* dans le panneau de gauche ;
- (c) sous l'onglet *Artistic Style*, fournissez le chemin :

C:\Program Files (x86)\AStyle\bin\AStyle.exe²⁴

à droite de l'étiquette *Artistic Style command* ;

- (d) selon vos préférences, choisissez un fichier de style global ou propre à chaque projet ;
- (e) cliquez *OK*.

À titre d'exemple, voici le contenu du fichier de style utilisé par l'auteur de ce TD :

```
--style=allman
--indent=spaces=4
--indent-modifiers
--indent-switches
--indent-cases
--indent-col1-comments
--pad-oper
--pad-header
--align-pointer=middle
--align-reference=middle
--break-closing-brackets
--keep-one-line-statements
--convert-tabs
--max-code-length=70
--break-after-logical
--close-templates
--lineend=linux
```

Davantage d'informations sur les options de formatage de code sont disponibles sur le [site web](#)²⁵ d'Artistic Style.

7.2 Todo

Le *plugin* Todo permet d'atteindre rapidement des étiquettes disséminées dans les sources d'un projet. Il est rangé parmi les *plugins* *Utilities*.

23. <http://astyle.sourceforge.net/> (consulté le 13 septembre 2021).

24. Ou C:\Program Files\AStyle\bin\AStyle.exe

25. <http://astyle.sourceforge.net/astyle.html> (consulté le 13 septembre 2021).

Pour le configurer et ajouter de nouveaux *tags*, faites comme pour le *plugin* Beautifier mais choisissez *To-Do* dans le panneau de gauche de la fenêtre *Options*. Le résultat de l'analyse du fichier ou du projet courant, selon la configuration, apparaît en bas de la zone d'édition du code, dans la vue *To-Do Entries*.