

DÉVELOPPEMENT COLLABORATIF, GITHUB ET QT CREATOR

TABLE DES MATIERES

Développement Collaboratif, Github et Qt Creator	1
1. Créer un compte github.....	2
2. Créer un “repository”	2
3. Création d’un token d’accès.....	2
4. Ajout de vos collaborateurs.....	3
5. Notion de “commit”	3
6. Installez git sur votre machine.....	3
7. Initialisation de git.....	3
8. Notions de push, pull et clone.....	4
9. Configurez Qt Creator	4
10. Importer votre repo dans Qt Creator.....	4
11. Ajout de l’url d’un repo git dans Qt Creator	5
12. Commit avec Qt Creator.....	5
13. Push avec Qt Creator	5
14. Pull avec Qt Creator	5
Annexes	6
Doc 1, Bouton new repository	6
Doc 2, Création d’un repository	6
Doc 3, Accès au menu des tokens	7
Doc 4, Génération d’un token.....	7
Doc 5, Ajout de collaborateurs.....	8
Doc 6, Initialisation de git, ici sous windows	8
Doc 7, Configuration de Qt Creator, ici sous windows.....	8
Doc 8, Import du projet.....	9
Doc 9, Import de repository.....	9
Doc 10, Commit du projet.....	10

1. CRÉER UN COMPTE GITHUB

Rendez-vous sur github.com et créez-vous un compte si ce n'est pas déjà fait !

2. CRÉER UN "REPOSITORY"

Pour développer un projet avec GitHub, il vous faut créer un "repository", ce repository sera l'emplacement où sera stocké et partagé votre projet.

- Dans votre page d'accueil GitHub, vous trouverez en haut à droite de l'interface, un bouton "*New repository*" ([Cf. Doc 1](#)), il vous suffit de cliquer dessus.
- Donnez un nom à votre "repository" et choisissez sa visibilité ([Cf. Doc 2](#)). En public tout le monde pourra voir et copier les fichiers contenus dans votre "repository". Néanmoins, personne ne pourra modifier les fichiers sur le "repository".
En privé, seuls vous et les personnes que vous autorisez peuvent voir, copier et modifier les fichiers.

NB : A partir de maintenant le terme "repository" sera abrégé en "repo"

3. CRÉATION D'UN TOKEN D'ACCÈS

Étape importante du processus, la création d'un token d'accès permettra à votre utilitaire git d'accéder à votre repo.

Accédez à vos *paramètres*, puis aux *paramètres développeurs* et enfin au menu "*Personal Access Token*" ([Cf. Doc 3](#)).

Une fois dans ce menu, cliquez sur "*Generate new token*".

Lors de la création de ce token plusieurs paramètres importants et sensibles sont nécessaires :

- La date d'expiration (une fois cette date passée, votre token ne sera plus utilisable)
- Les permissions (ou scopes) :
Seul le scope repo et ses dépendances sont nécessaires pour écrire ou accéder au fichier depuis un utilitaire git ([Cf. Doc 4](#)). Cochez seulement les permissions qui vous sont strictement nécessaires.

Attention, une fois que vous aurez cliqué sur générer votre token **ne sera plus consultable**, vous pourrez néanmoins toujours modifier les permissions qui lui sont associées.

NB : Attention, les tokens sont des outils sensibles, ils doivent rester strictement personnels.

4. AJOUT DE VOS COLLABORATEURS

Pour que vos collaborateurs puissent accéder à votre repo et donc au projet, il faut leur autoriser l'accès.

Rendez-vous dans les *options* de votre repo, puis dans “*Manage Access*”, vous pourrez ensuite ajouter vos collaborateurs avec leurs noms d'utilisateurs GitHub, grâce au bouton “*Add people*”. (Cf. [Doc 5](#))

Vous pourrez ensuite modifier les permissions accordées aux utilisateurs depuis le même menu.

5. NOTION DE “COMMIT”

GitHub, et les autres outils git, fonctionnent avec des “commits”. Ces commits sont des sortes de captures de l'avancée du projet. A chaque fois que vous aurez à modifier votre projet, vous devrez “commit”. C'est-à-dire, capturer l'état actuel et expliquer les modifications que vous avez apportées. Cela vous permettra par exemple de retrouver un état antérieur du projet.

6. INSTALLEZ GIT SUR VOTRE MACHINE

Ici deux chemins s'offrent à vous,

- Si vous êtes sous Windows, il vous faut tout d'abord installer git sur votre machine, vous pouvez le faire en suivant ce lien : <https://gitforwindows.org/>
- Si vous êtes sous Linux, git est normalement déjà préinstallé. Le cas échéant, il suffit de taper “*apt install git*” dans votre terminal.

7. INITIALISATION DE GIT

- Sous Windows vous devez tout d'abord lancer “*git-cmd.exe*”, vous pouvez y accéder via la barre de recherche sous Win10 ou à l'emplacement suivant : C:\Program Files\Git
- Sous Linux, ouvrez simplement un interpréteur de commande

Une fois git-cmd ou votre terminal ouvert, tapez les deux commandes suivantes (Cf. [Doc 6](#)) :

- *git config --global user.name “Votre Nom”*
- *git config --global user.email votreemail@email.com*

Git est maintenant initialisé sur votre machine.

8. NOTIONS DE PUSH, PULL ET CLONE

Trois interactions avec votre repo existent : push, pull et clone.

- “Clone” est l’action que vous allez effectuer sur une nouvelle machine pour importer le projet. Cela va copier tous les fichiers du repo sur votre machine.
- “Pull” ressemble à clone mais est utilisé pour mettre à jour les modifications faites au projet, par exemple si un collaborateur a modifié un bout de code, vous n’aurez qu’à “pull” pour obtenir les modifications.
- “Push” est l’action inverse à “pull”, il va prendre les fichiers que vous avez modifiés et les envoyer sur le repo. Attention pour “push” des fichiers il vous faudra au préalable “commit” les modifications.

9. CONFIGUREZ QT CREATOR

Avant de pouvoir utiliser git avec Qt Creator, il faut indiquer l’emplacement de git à Qt Creator.

Pour cela cliquez sur *Outils* (en haut de votre écran), puis sur *Options*. Naviguez ensuite dans le menu déroulant à droite jusqu’à trouver “*Gestion de versions*”. Trouvez ensuite le volet “*git*” et entrez le chemin de git. ([Cf. Doc 7](#))

Sous Windows le chemin est souvent : C:\Program Files\Git\bin.

Sous Linux le chemin est souvent : /usr/bin/git

10. IMPORTER VOTRE REPO DANS QT CREATOR

NB : L’étape 10 est à sauter si vous n’avez aucun fichier dans votre repo. Il vous suffit alors de créer un projet Qt classique.

Maintenant que votre repo est créé, que git est configuré et que vous avez compris les notions de clone, push et pull ; nous allons pouvoir importer votre repo dans Qt Creator.

Ouvrez Qt Creator puis cliquez sur *nouveau projet*. Choisissez ensuite “*Import Project*” et “*git clone*” ([Cf. Doc 8](#)).

Qt Creator vous demandera ensuite d’entrer l’URL de votre repo puis l’emplacement et le dossier où le projet va être stocké. ([Cf. Doc 9](#)).

Enfin, votre nom d’utilisateur GitHub ainsi que le token (généré à [l’étape 3](#)) vous seront demandés.

11. AJOUT DE L'URL D'UN REPO GIT DANS QT CREATOR

NB : Si vous avez suivi l'étape 10, l'url de votre repo est déjà dans Qt Creator, vous pouvez donc sauter cette étape

Pour ajouter ou consulter la liste des repositories du projet dans Qt Creator, il vous faut aller dans : *Outils → Git → Remote Repository → Manage Remotes*

Vous aurez accès à la liste des repositories du projet et vous pourrez choisir sur quel repo vous voulez push ou pull.

12. COMMIT AVEC QT CREATOR

Pour commit sur Qt Creator il vous faut vous rendre dans : *Outils → Git → Local Repository → Commit.*

Une fenêtre s'ouvrira alors pour vous demander de sélectionner les fichiers que vous souhaitez commit et d'entrer une description. Vous pourrez ensuite cliquer sur soumettre. ([Cf. Doc 10](#))

13. PUSH AVEC QT CREATOR

Rappel : Il vous faut au préalable avoir commit pour pouvoir push.

Une fois que vous avez commit, vous pouvez maintenant push vos modifications sur le repo.

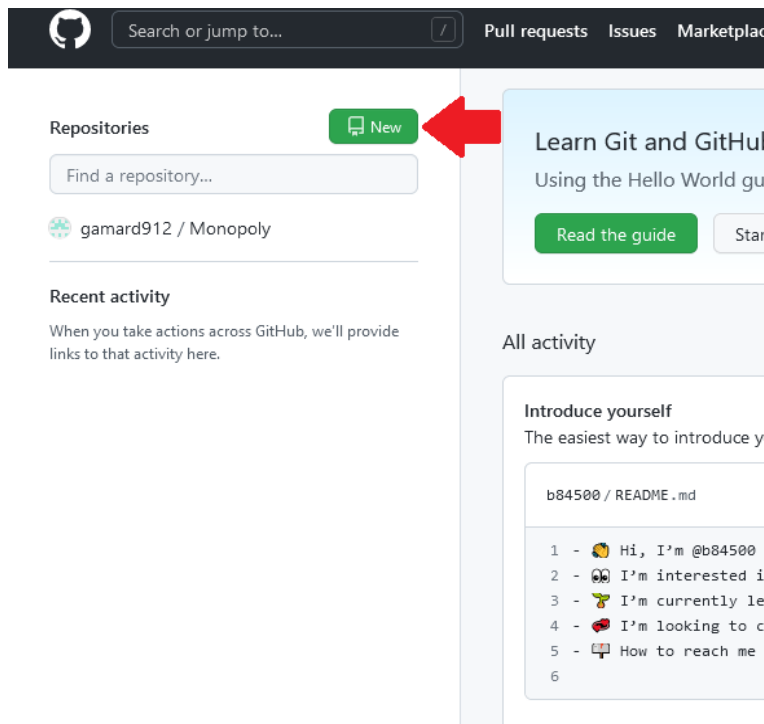
Allez dans : *Outils → Git → Remote Repository → Push*

14. PULL AVEC QT CREATOR

Pour pull sur Qt Creator allez dans : *Outils → Git → Remote Repository → Pull*

ANNEXES

DOC 1, BOUTON NEW REPOSITORY




DOC 2, CRÉATION D'UN REPOSITORY

Create a new repository



A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * Repository name *

 b84500 /

Great repository names are short and memorable. Need inspiration? How about [psychic-couscous](#)?

Description (optional)

- ☐  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☒  **Private**
You choose who can see and commit to this repository.

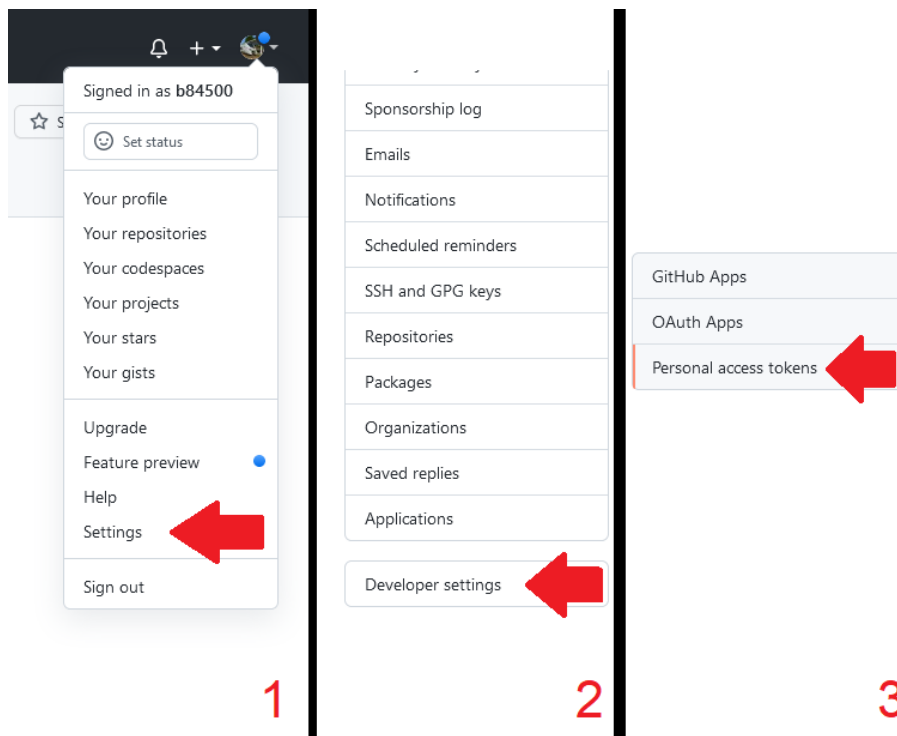
Initialize this repository with:

Skip this step if you're importing an existing repository.

- ☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)
- ☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)
- ☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

DOC 3, ACCÈS AU MENU DES TOKENS



DOC 4, GÉNÉRATION D'UN TOKEN

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

Tutoriel Github et QT

What's this token for?

Expiration *

30 days

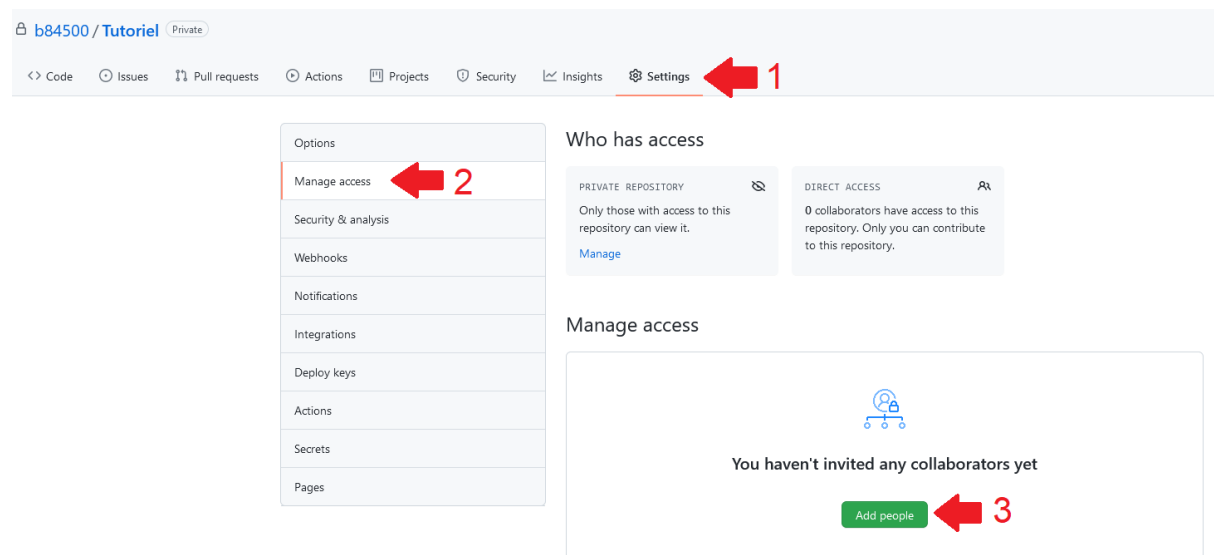
The token will expire on Wed, Dec 1 2021

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows

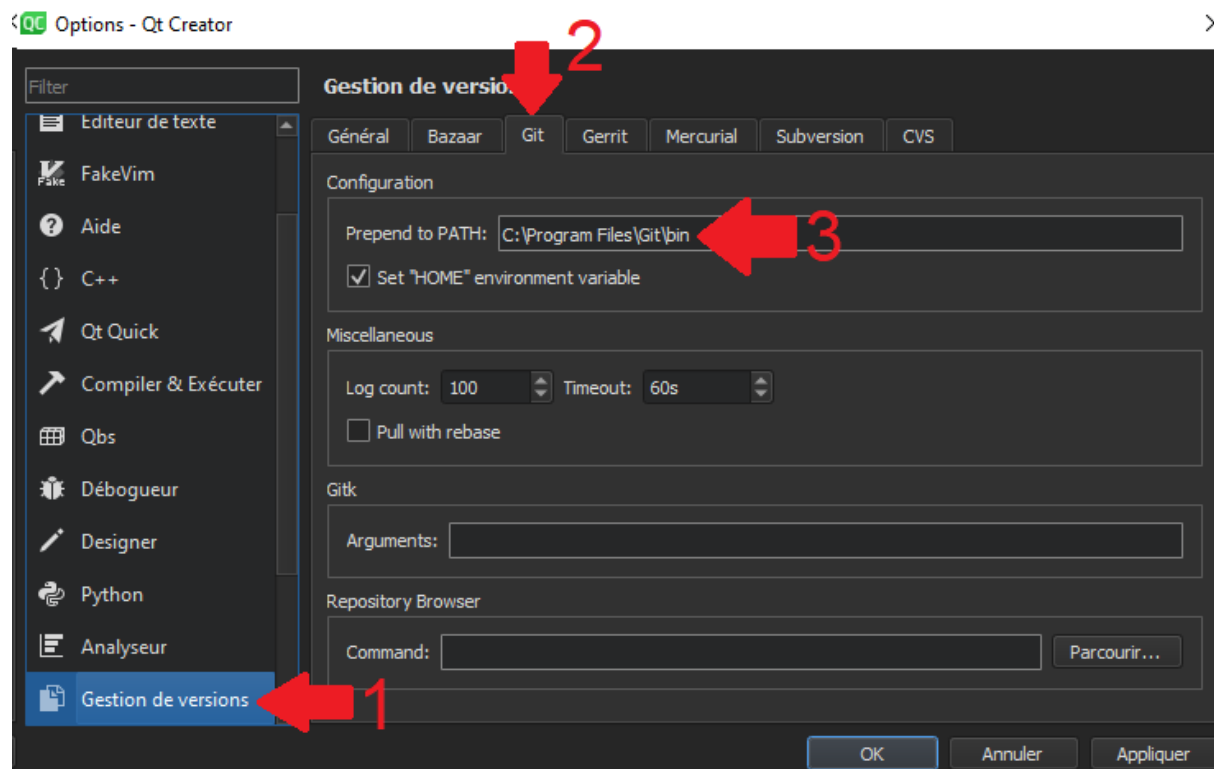
DOC 5, AJOUT DE COLLABORATEURS



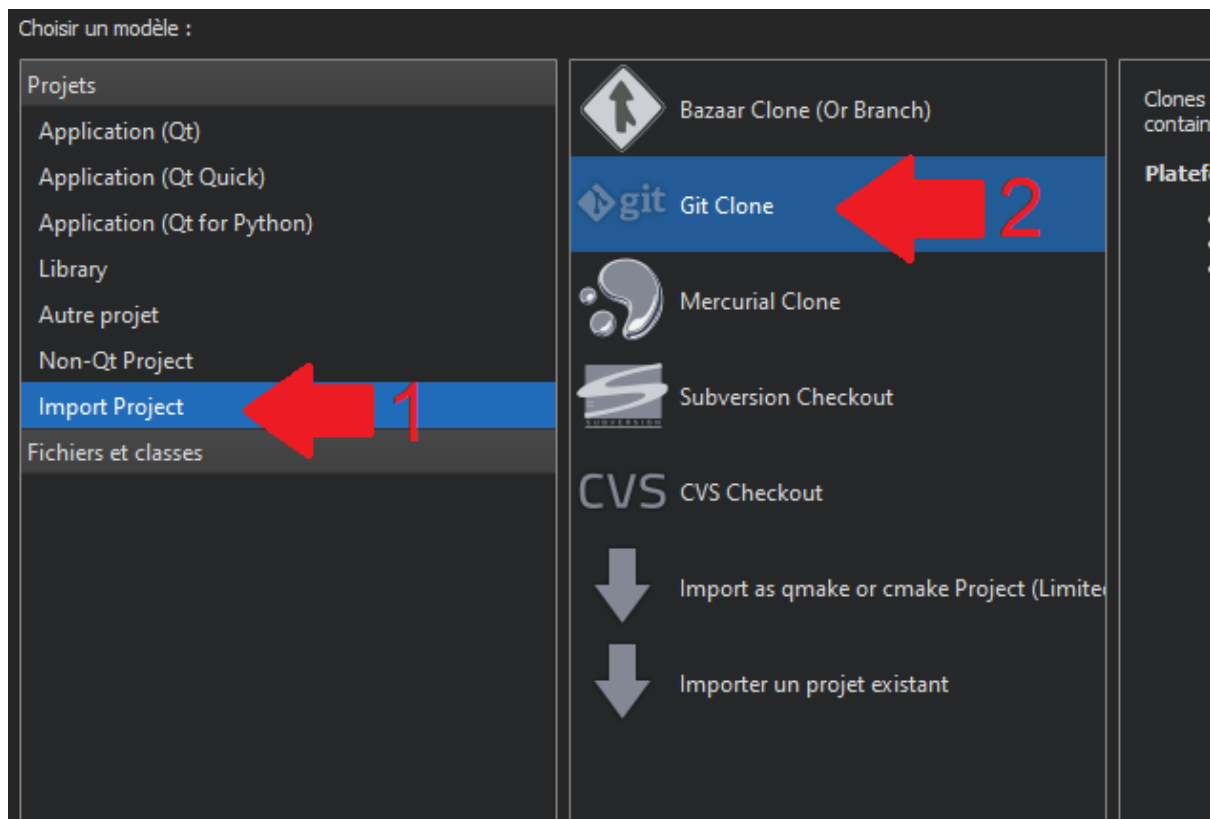
DOC 6, INITIALISATION DE GIT, ICI SOUS WINDOWS

```
git config --global user.email eziocangialosi@gmail.com
git config --global user.name "Ezio Cangialosi"
```

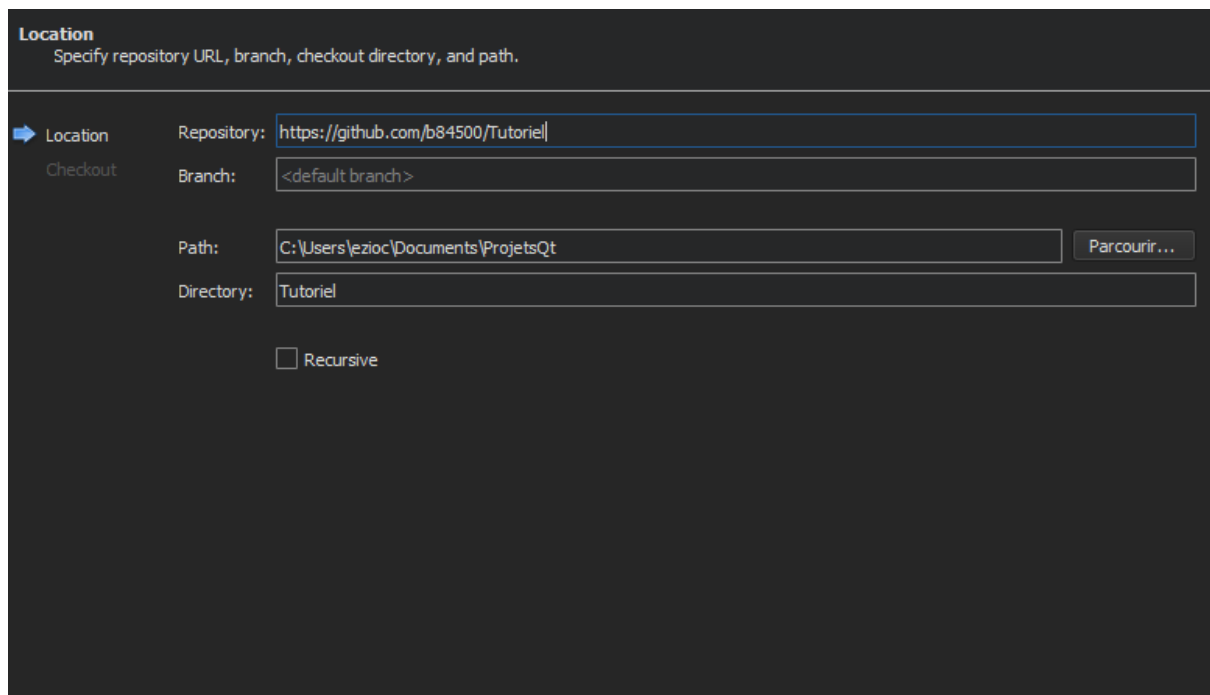
DOC 7, CONFIGURATION DE QT CREATOR, ICI SOUS WINDOWS



DOC 8, IMPORT DU PROJET



DOC 9, IMPORT DE REPOSITORY



DOC 10, COMMIT DU PROJET

Informations générales

Dépôt : C:\Users\ezio\Documents\ProjetsQt\Tutoriel
Branche : master...origin/master

Informations de commit

Auteur :
Email :
☐ Éviter les crochets ☐ Sign off

Description

Warning: The commit message is very short.
Tutoriel de commit

Fichiers

☒ Select all

État	Fichier
<input checked="" type="checkbox"/> modifié	main.cpp

Soumettre 1/1 fichier

Diff Select