

# INFO2009 - Introduction à l'informatique

## Le laboratoire à distance

### Introduction

Si l'informatique peut s'avérer déroutante au début, elle a un avantage majeur par rapport aux autres matières enseignées en sciences appliquées : on peut facilement s'assurer, au moins pour sa majeure partie, qu'un programme est cohérent et fonctionne. Il y a d'une part des outils théoriques de conception de programme (notamment les spécifications et les invariants de boucle), enseignés par le cours d'INFO2009, et d'autre part, l'étudiant a la possibilité de pouvoir tester ses programmes en les *compilant* sur un ordinateur. C.-à-d., les programmes sont convertis en langage machine afin de pouvoir les exécuter et de s'assurer ainsi de leur bon fonctionnement<sup>1</sup>. Pouvoir compiler chez soi a aussi l'avantage d'aider l'étudiant à se familiariser avec la syntaxe du langage C, chose qui n'est pas toujours aisée à faire sur papier. Nous proposons donc plusieurs tutoriels détaillant différentes solutions pour compiler, chez soi, un programme écrit dans le langage C.

Dans ce tutoriel, nous expliquons comment l'étudiant peut se connecter à distance à un ordinateur de l'ULg afin de bénéficier du même environnement que celui des séances de laboratoire, de sorte qu'il puisse les poursuivre à la maison ou compiler des exercices supplémentaires. Cette solution nécessite l'installation et l'usage d'au moins deux programmes: un pour pouvoir établir un terminal à distance et un second pour échanger des fichiers avec la machine distante afin d'y envoyer de nouvelles solutions ou de récupérer sur son ordinateur celles qui s'y trouvent déjà. Nous présentons d'abord les machines que l'étudiant peut utiliser à distance et expliquons ensuite comment installer et utiliser les programmes requis.

En annexe, nous présentons quelques éditeurs de texte que l'étudiant peut télécharger et utiliser pour pouvoir programmer de manière confortable (c.-à-d., coloration du code, outils d'indentation, auto-complétion, etc.) avant d'envoyer son code sur la machine distante. Nous abordons aussi brièvement l'usage de `vim`, un éditeur de texte qui peut simplifier l'interaction avec le laboratoire puisqu'il permet d'écrire son code directement dans le terminal, ainsi que l'utilisation de `X11` pour utiliser *gedit* à distance. Une dernière annexe explique aussi les principes de la connexion à distance, pour les étudiants curieux.

---

<sup>1</sup>Gardez cependant à l'esprit qu'un bon fonctionnement avec un certain scénario d'exécution ne garantit pas que la solution ne contient pas d'erreurs. Comme l'a dit Edsger Wybe Dijkstra (célèbre informaticien néerlandais) : “*tester démontre la présence de bugs, pas leur absence*”.

## A propos des machines utilisables

Vous avez à votre disposition deux groupes de machines utilisables.

- Les machines **candiXX**.montefiore.ulg.ac.be (où  $XX = 01, 02 \dots, 54$ ): ce sont les mêmes ordinateurs que ceux des séances de laboratoire. Vous pouvez donc vous y connecter avec les identifiants *pml*i que vous avez reçu à votre première séance de laboratoire.
- Les machines **ms8xx**.montefiore.ulg.ac.be (où  $xx = 00, 01, \dots, 20$ , aussi appelé “Réseau 8”): ce sont les ordinateurs qui se trouvent en libre accès au premier étage de l’institut Montefiore. Ils nécessitent eux aussi des identifiants de connexion, à obtenir en demandant la création d’un compte sur <http://www.student.montefiore.ulg.ac.be/>. Ce sont les ordinateurs de référence pour les travaux pratiques de programmation de nombreux cours.

Ces deux groupes de machines proposent un environnement de programmation quasi identique. En revanche, **dans le cadre du cours INFO2009, nous vous conseillons de toujours passer par les machines candiXX** pour deux raisons. La première est que vous pourrez retrouver les fichiers créés lors de vos séances de laboratoire, et donc vous pourrez les consulter à nouveau, les modifier ou les télécharger. La seconde raison est que plus d’ordinateurs sont à votre disposition et qu’il est dès lors plus facile de trouver une machine disponible lorsqu’une ou plusieurs machines sont injoignables.

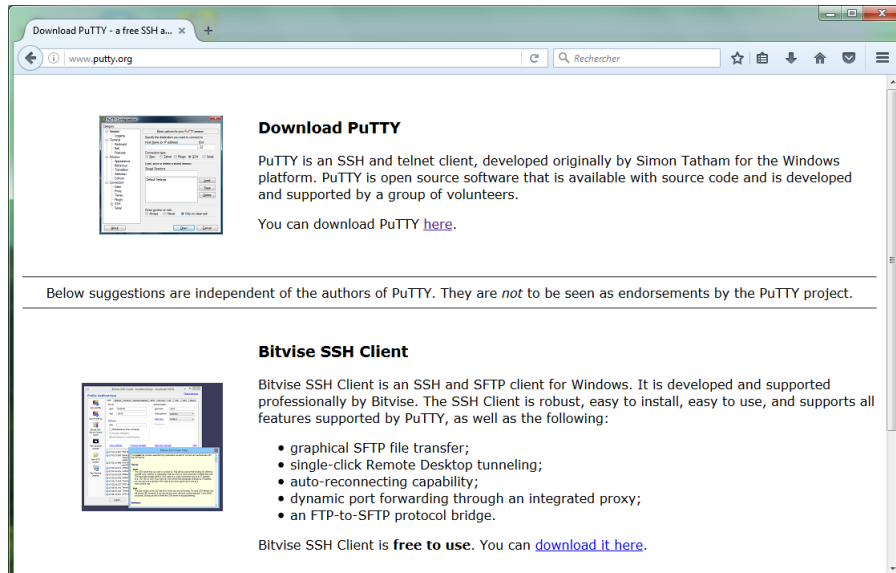
Quel que soit le groupe choisi, la méthode pour se connecter ou déplacer des fichiers entre votre ordinateur personnel et la machine distante est exactement la même.

## Installations des programmes requis

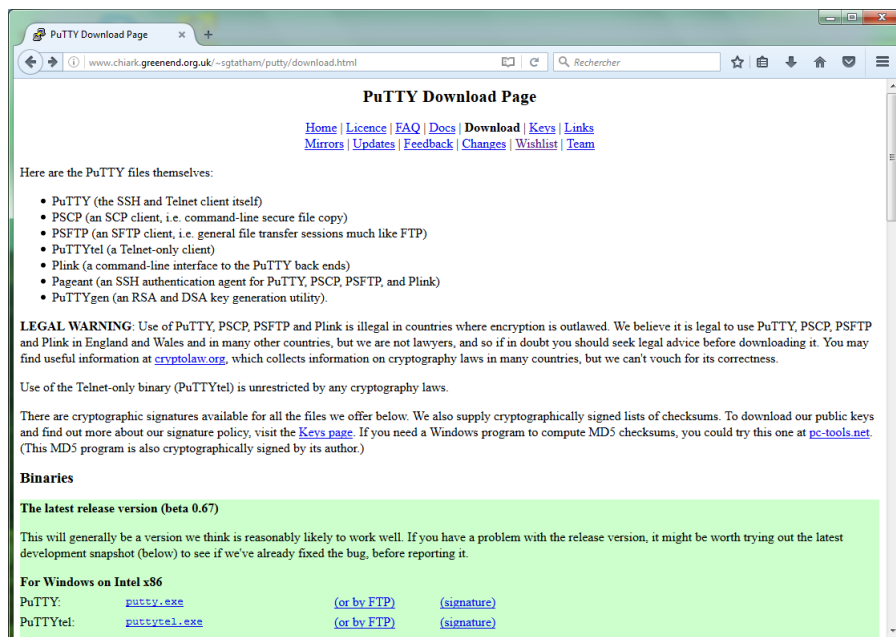
### Programmes pour établir la connexion à distance

#### Sous Windows

Sous Windows (toute version confondue), vous aurez besoin de *PuTTY*, un petit programme gratuit. Pour le télécharger, rendez-vous sur <http://www.putty.org> (montré à la capture d’écran (a)) et cliquez sur le lien “*here*” (premier lien) pour accéder à une page de téléchargement (capture (b)). Le premier lien vous permettra de télécharger la dernière version stable du programme.

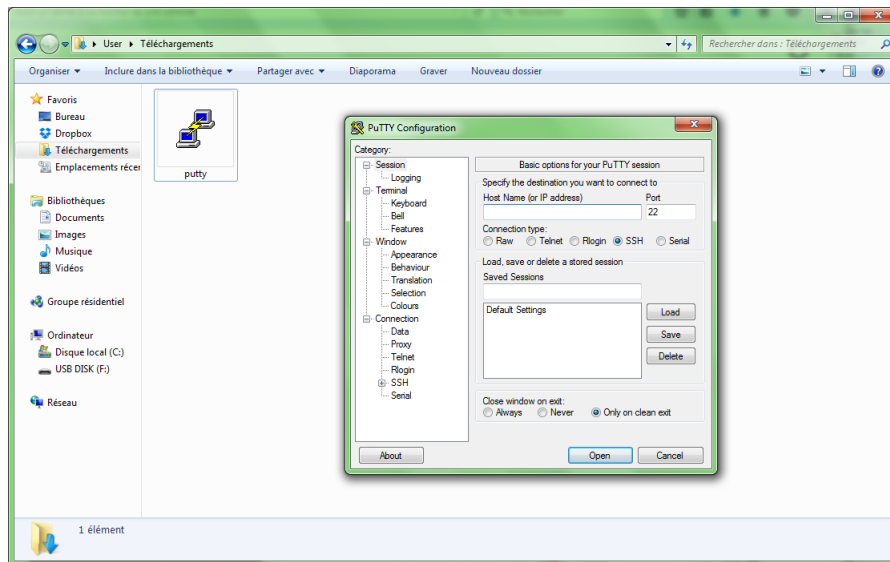


(a) Site officiel de *PuTTY*.

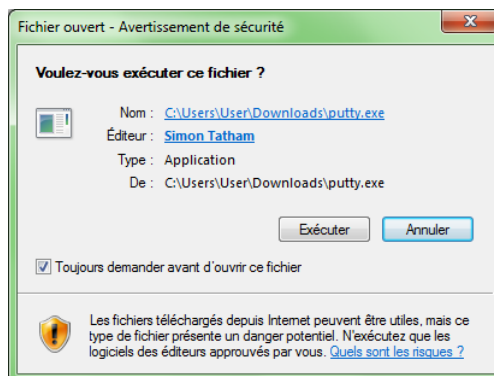


(b) Téléchargement de *PuTTY*.

Contrairement à d'autres programmes, *PuTTY* est directement fourni sous forme de programme exécutable et n'a donc, par conséquent, pas besoin d'être installé. Il suffit de double-cliquer dessus pour pouvoir l'utiliser (comme le montre la capture (c)). Notez, cependant, que Windows risque de vous afficher un avertissement de sécurité (comme à la capture d'écran (d)) lors de la première exécution. *PuTTY* ne posant aucun risque, cliquez sur le bouton "*Exécuter*", éventuellement après avoir décoché la case "*Toujours demander avant d'ouvrir ce fichier*" pour ne plus recevoir d'avertissement. Enfin, lorsque la fenêtre "*PuTTY configuration*" est ouverte, il suffit de cliquer sur le bouton "*Cancel*" pour quitter le programme.



(c) Lancement de *PuTTY*.



(d) Un possible avertissement de Windows.

## Sous Linux

Par défaut, la plupart des distributions courantes de Linux (notamment Ubuntu) proposent dès leur installation *OpenSSH*, un programme qui permet d'établir la connexion à distance. Pour vérifier que ce dernier est bien installé, ouvrez un terminal et entrez la commande suivante:

```
$ ssh -V
```

Si votre terminal vous annonce qu'il ne reconnaît pas cette commande, vous devrez installer *OpenSSH* vous-même. Il suffit d'exécuter la commande

```
$ sudo apt-get install openssh-client
```

La commande précédente sera utilisable juste après.

## Sous Mac OS X

*OpenSSH* est pré-installé sous Mac OS X et s'utilise, comme sous Linux, à travers le terminal.

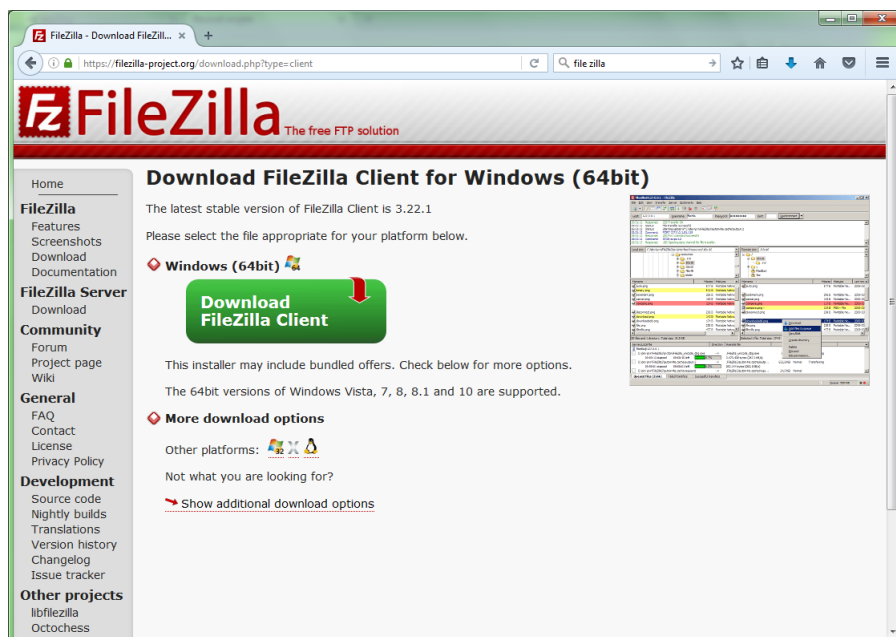
## Un programme pour envoyer ses fichiers à distance: *FileZilla*

### Sous Windows et Mac OS X

La solution la plus simple et directe consiste à utiliser *FileZilla*, un programme avec interface graphique qui permet de se connecter à une machine distante pour y envoyer ses fichiers ou même télécharger les fichiers qui s'y trouvent déjà (bien pratique pour récupérer vos exercices de laboratoire). Pour installer *FileZilla*, rendez-vous sur <http://filezilla-project.org/> et cliquez sur le bouton "*Download FileZilla Client (All platforms)*". Cette étape est illustrée à la capture d'écran (e). Par défaut, la nouvelle page devrait afficher en premier lien de téléchargement (sous forme de bouton vert) la version qui sied le mieux à votre système d'exploitation. Dans notre exemple (capture d'écran (f)), nous réalisons le téléchargement à partir de Windows 7.



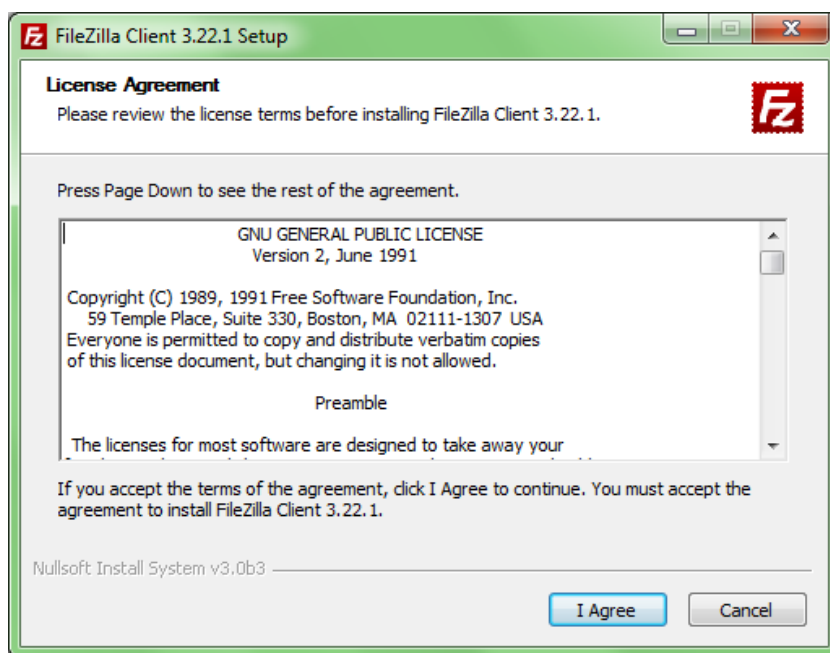
(e) Site web officiel du projet *FileZilla*.



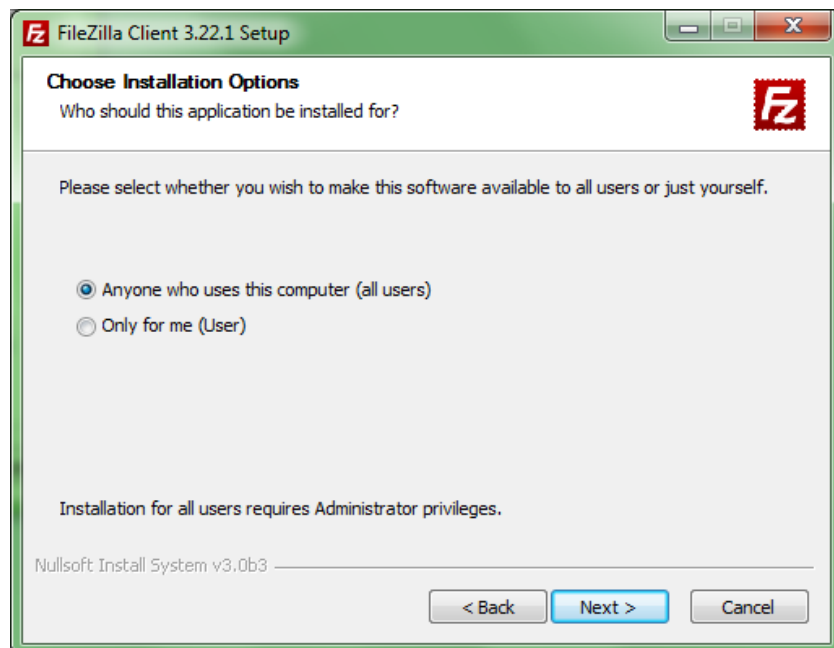
(f) Téléchargement de *FileZilla* depuis Windows 7 (cliquez sur le bouton vert).

Notez qu'il existe deux versions pour Windows: une pour Windows 32-bit, et une autre pour Windows 64-bit. Si vous n'êtes pas sûr de quelle version vous devez choisir, rendez-vous dans votre *Panneau de configuration* (à partir du menu de démarrage, sinon avec la recherche via la loupe sous Windows 10) et cliquez sur le volet "*Système*". Dans la nouvelle fenêtre, vous devriez voir une ligne "*Type du système*" qui vous indiquera si vous utilisez la version 64-bit ou 32-bit de Windows. Choisissez alors la version correspondante de FileZilla.

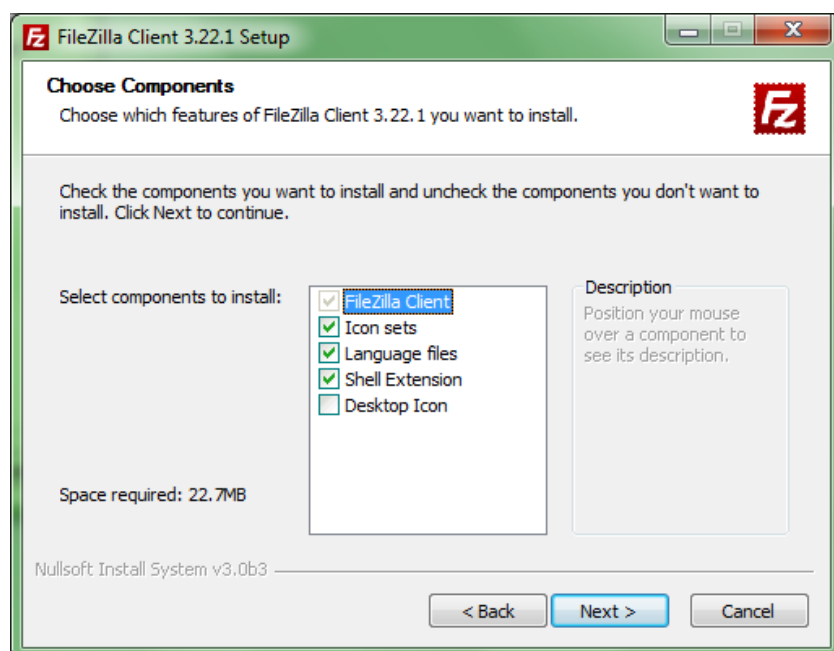
Une fois le téléchargement du fichier d'installation terminé, il vous reste à l'ouvrir pour lancer l'installation proprement dite. Nous avons illustré la procédure d'installation sous Windows aux captures d'écran (g), (h), (i), (j), (k) et (l). Celle-ci est très classique: vous pouvez simplement cliquer sur "*I Agree*" et de passer les différentes étapes en laissant les paramètres par défaut. Si souhaité, vous pourrez néanmoins changer ceux-ci pour par exemple créer une icône sur votre bureau (cochez l'option *Desktop Icon* à l'écran (i)). Une fois l'installation terminée, vous aurez le choix entre quitter et ouvrir directement *FileZilla*. Si vous quittez et n'avez pas créé une icône sur votre bureau, vous pourrez relancer *FileZilla* à partir de votre menu de démarrage ou en réalisant une recherche (la loupe sous Windows 10).



(g) Conditions d'utilisation.

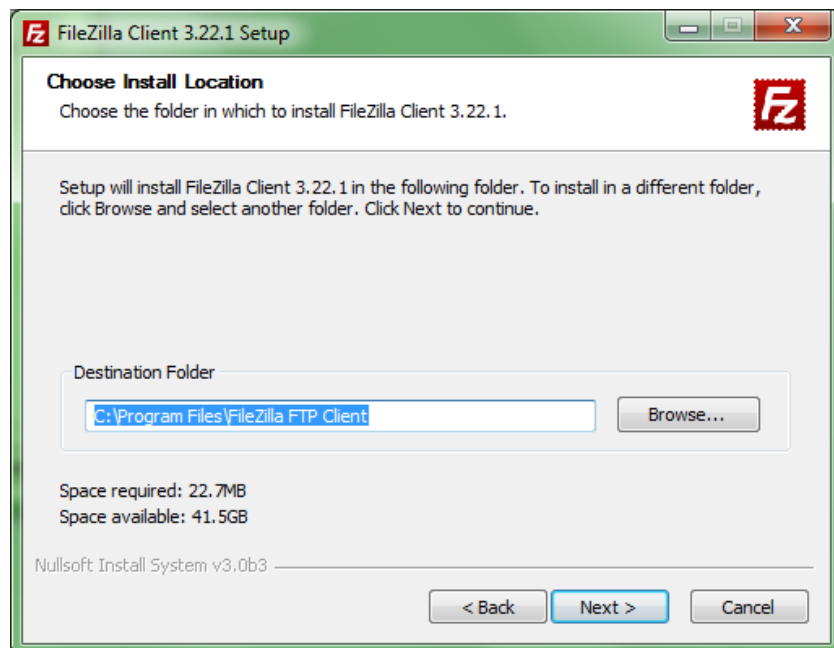


(h) Choix des utilisateurs concernés.

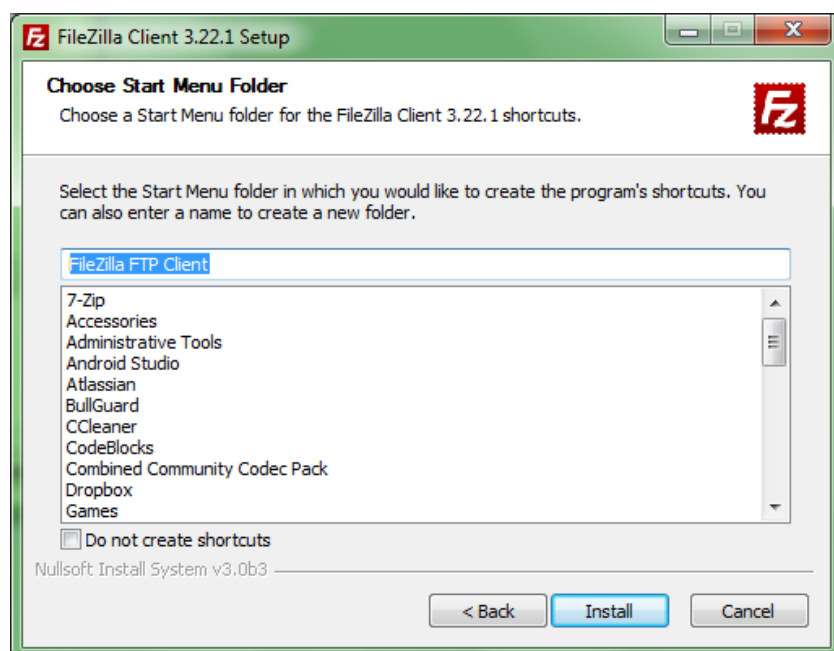


(i) Options d'installation (les choix par défaut suffisent).

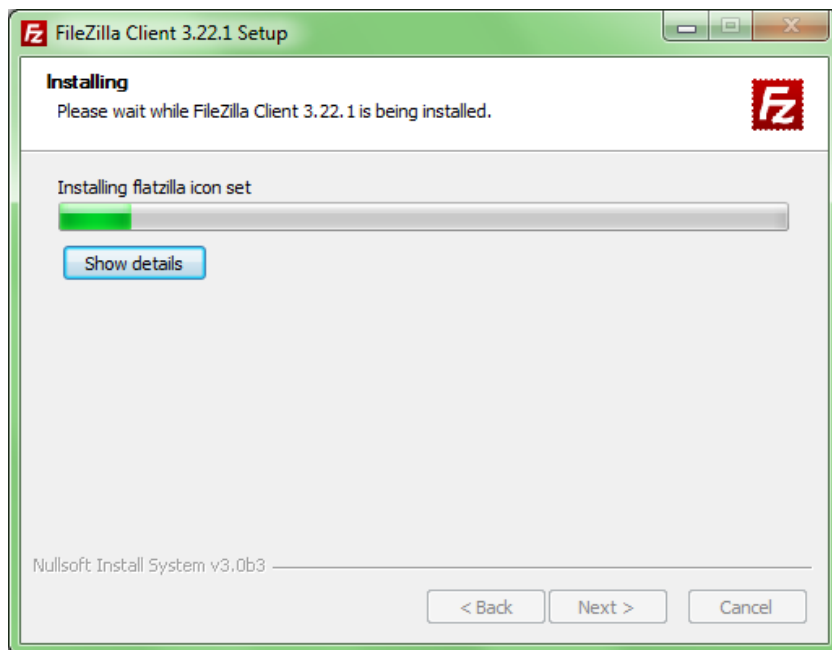




(j) Choix du dossier d'installation (le choix par défaut suffit).



(k) Choix du nom du volet dans le menu de démarrage.



(1) L'installation peut enfin commencer réellement.

## Sous Linux

Avec une distribution Linux, nous allons également recourir à *FileZilla*. S'il est possible de l'installer en téléchargeant une archive .tar.gz sur le site officiel, il sera encore plus rapide d'ouvrir un terminal et de taper la commande:

```
$ sudo apt-get install filezilla
```

*FileZilla* sera utilisable juste après.

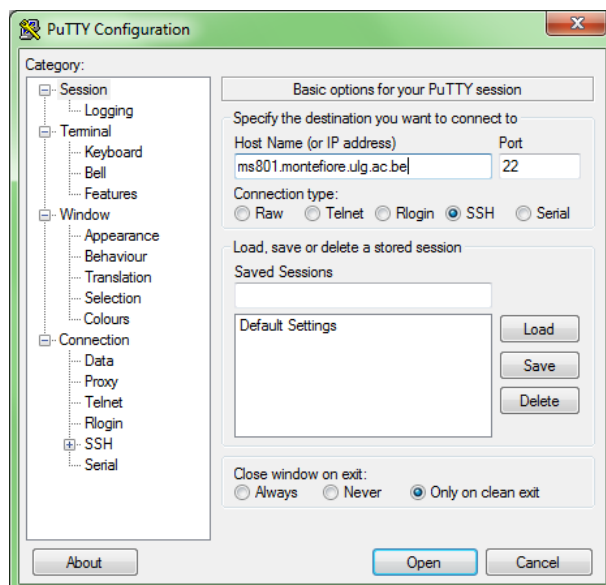
## Connexion et utilisation à distance

### Etablir le terminal à distance

#### Sous Windows (avec PuTTY)

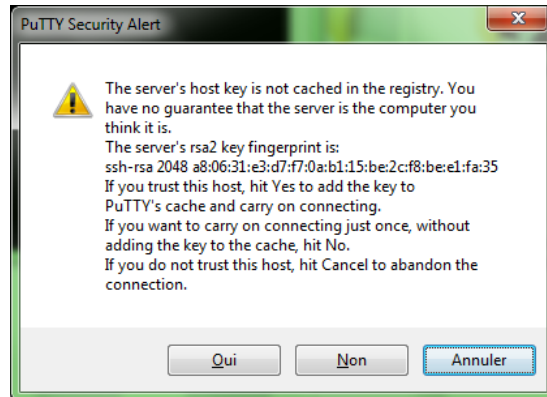
Démarrez *PuTTY*. Pour rappel, il suffit de cliquer dessus dès la fin du téléchargement; vous pouvez aussi déplacer le fichier à votre guise et par exemple le mettre sur votre bureau. Comme expliqué précédemment, vous aurez peut-être un avertissement de sécurité de Windows au lancement du programme (cf. la capture d'écran (d)). Dans ce cas, cliquez sur “*Exécuter*”, en ayant éventuellement décoché la case “*Toujours demander avant d’ouvrir ce fichier*”.

Dans la fenêtre de configuration de *PuTTY*, repérez les champs *Host Name* et *Port*. Dans le champ *Port*, entrez simplement “22”, et entrez dans *Host Name* une adresse `[machine].montefiore.ulg.ac.be` où vous aurez remplacé `[machine]` par la machine distante de votre choix (candiXX où XX = 01, 02..., 54 pour le laboratoire du B37 et ms8xx où xx = 00, 01..., 20 pour le Réseau 8). Vérifiez, enfin, que l’option “*SSH*” est bien cochée dans *Connection type*. La capture d'écran (m) illustre une tentative de connexion à la machine ms801.montefiore.ulg.ac.be. Enfin, cliquez sur le bouton “*Open*”.



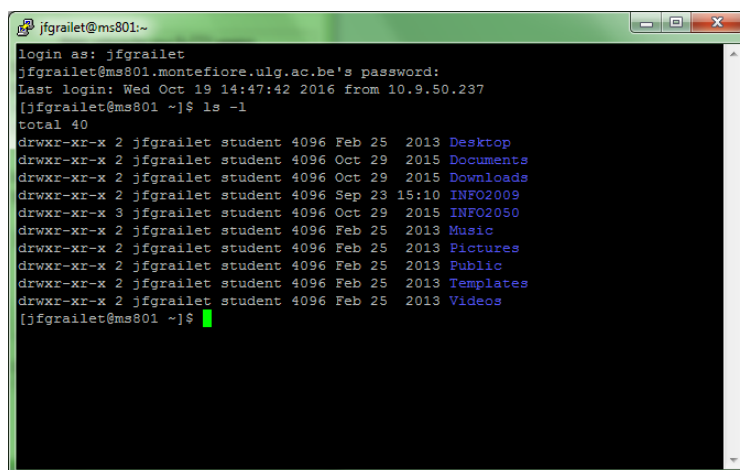
(m) Connexion à ms801.montefiore.ulg.ac.be.

Au moment d'ouvrir la connexion, *PuTTY* vous enverra certainement un avertissement stipulant qu'il n'arrive pas à confirmer l'authenticité de la machine distante, comme montré à la capture d'écran (n). Cliquez sur le bouton "*Oui*".



(n) Avertissement de *PuTTY*. Cliquez sur "*Oui*".

Un terminal va alors s'ouvrir et vous demander successivement votre identifiant de connexion (votre identifiant *pml*i pour le laboratoire du B37, ou votre identifiant de Montefiore pour le Réseau 8) et votre mot de passe. Une fois l'identification réussie, vous pourrez commencer à utiliser des commandes sur la machine distante. La capture d'écran (o) illustre le terminal après connexion (ici, à *ms801.montefiore.ulg.ac.be*). Pour clore la session, il suffit de cliquer sur la croix en haut à droite.



(o) Connexion réussie sur *ms801.montefiore.ulg.ac.be*.

Si la connexion avec une machine (par exemple, *candi01*) échoue, n'hésitez pas à essayer d'établir une connexion vers une autre machine.

## Sous Linux et Mac OS X

Ouvrez un terminal et entrez la commande suivante:

```
$ ssh [user]@[machine].montefiore.ulg.ac.be
```

Où vous aurez remplacé `[user]` par votre identifiant de connexion (soit votre identifiant *pml*i pour le laboratoire du B37, soit votre identifiant de Montefiore pour le Réseau 8) et `[machine]` par `candiXX` (`XX = 01, 02...`, `54`) pour utiliser une machine du laboratoire du B37 ou `ms8xx` (`xx = 00, 01...`, `20`) pour utiliser une machine du Réseau 8. Le terminal vous indiquera alors, très probablement, qu'il ne peut confirmer l'authenticité de la machine distante et vous demandera si vous souhaitez continuer. Entrez *"yes"* pour poursuivre. Notez que cette question ne survient que lors de la toute première connexion avec une machine donnée. Le terminal vous demandera alors votre mot de passe. Une fois celui-ci correctement entré, vous pourrez commencer à entrer vos commandes sur la machine distante. La capture d'écran (p) montre la connexion avec *OpenSSH* sous Ubuntu sur la machine `ms804.montefiore.ulg.ac.be` (Réseau 8).

```
jfg@ms804:~$ ssh jfg@ms804.montefiore.ulg.ac.be
jfg@ms804:~$ ssh jfg@ms804.montefiore.ulg.ac.be
The authenticity of host 'ms804.montefiore.ulg.ac.be (139.165.8.227)' can't be established.
RSA key fingerprint is SHA256:tk4L1m1/PRTgpLmYXgZ4KvKKZUg0fe3pT/T7FgpQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ms804.montefiore.ulg.ac.be,139.165.8.227' (RSA) to the list of known hosts.
jfg@ms804.montefiore.ulg.ac.be's password:
[jfg@ms804 ~]$ ls -l
total 40
drwxr-xr-x 2 jfg student 4096 25 fév 2013 Desktop
drwxr-xr-x 2 jfg student 4096 29 oct 2015 Documents
drwxr-xr-x 2 jfg student 4096 29 oct 2015 Downloads
drwxr-xr-x 2 jfg student 4096 19 oct 14:57 INFO2009
drwxr-xr-x 3 jfg student 4096 29 oct 2015 INFO2050
drwxr-xr-x 2 jfg student 4096 25 fév 2013 Music
drwxr-xr-x 2 jfg student 4096 25 fév 2013 Pictures
drwxr-xr-x 2 jfg student 4096 25 fév 2013 Public
drwxr-xr-x 2 jfg student 4096 25 fév 2013 Templates
drwxr-xr-x 2 jfg student 4096 25 fév 2013 Videos
[jfg@ms804 ~]$
```

(p) Connexion avec *OpenSSH* sur ms804.montefiore.ulg.ac.be.

Si la connexion avec une machine (par exemple, candi01) échoue, n'hésitez pas à essayer d'établir une connexion vers une autre machine.

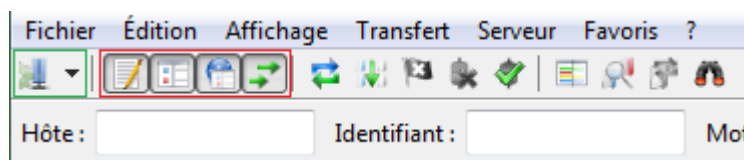
### Remarque importante

Le terminal à distance vous permettra d'utiliser toutes les commandes utilisées lors des séances de laboratoire (tel `gcc`, `ls`, `mkdir`) mais **ne permet pas d'utiliser *gedit*** car il s'agit d'un programme à interface graphique. Si vous désirez toutefois écrire votre code à distance, nous vous recommandons de lire les annexes de ce document portant sur l'éditeur de texte en console `vim` et sur `X11` (pour employer *gedit* à distance).

## Envoyer ses fichiers sur la machine distante avec *FileZilla*

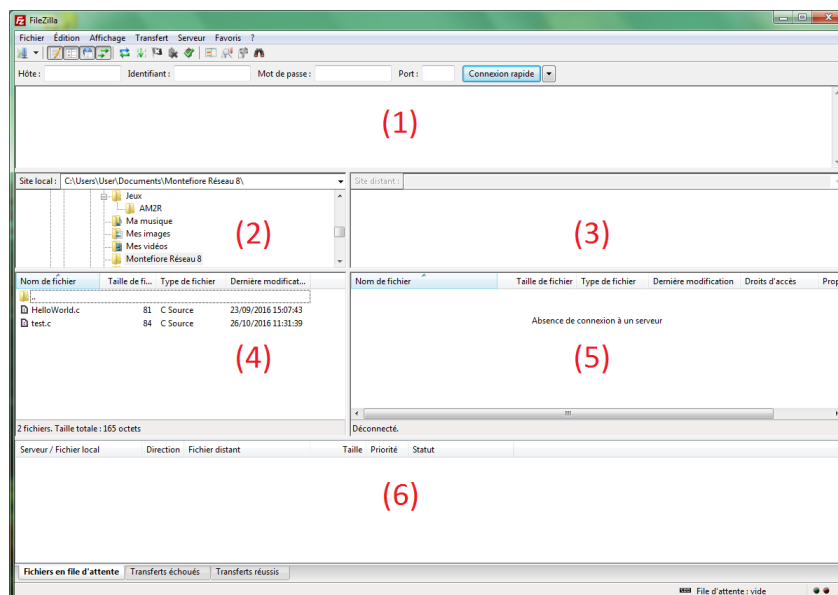
### L'interface de *FileZilla*

Démarrez *FileZilla*. Commencez par vérifier les boutons qui se trouvent en haut à gauche sous les menus déroulants, tels que montrés à la capture d'écran (q). Si ces boutons n'apparaissent pas, allez dans le menu déroulant *Affichage* et sélectionner l'item *Barre d'outils*. Pour garder une interface complète, assurez-vous que les quatre boutons (encadrés en rouge sur la capture (q)) après le premier en partant de la gauche (encadré en vert) sont bien activés. Sinon, cliquez une fois dessus. Ces boutons vous permettent en fait d'afficher/masquer une partie de la fenêtre. Pour votre première utilisation de *FileZilla*, nous vous conseillons de tout afficher.



(q) Les principaux boutons de *FileZilla*

Vous devriez alors avoir une interface identique à celle montrée à la capture d'écran (r), avec quelques variations mineures selon le système d'exploitation (sur nos captures, nous sommes sous Windows).



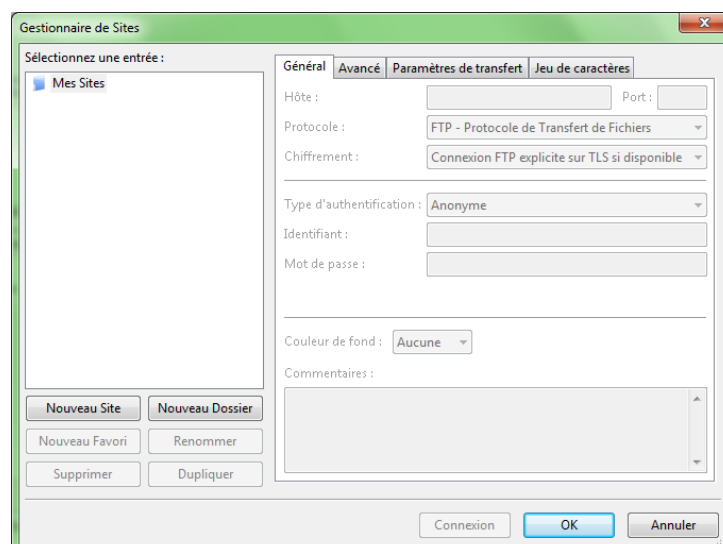
(r) L'interface de *FileZilla*

Sur cette capture, vous pouvez observer 6 zones:

- **Zone 1 - Le journal:** cette zone affiche le détail des différentes opérations effectuées par *FileZilla*. Elle peut par exemple indiquer pourquoi une tentative de connexion échoue (par exemple, à cause d'un mot de passe mal écrit).
- **Zones 2 & 3 - Arborescence locale/distante:** ces zones affichent l'arborescence des dossiers de votre ordinateur (à gauche) et des dossiers de la machine distante (à droite). Elles peuvent s'avérer utile pour savoir rapidement où vous vous trouvez.
- **Zones 4 & 5 - Dossier local/distant:** juste en dessous des zones 2 et 3, ces zones affichent le contenu du dossier en cours de consultation. C'est dans ces dossiers que les envois et téléchargements se feront.
- **Zone 6 - La file d'attente:** cette dernière zone affiche un historique des opérations de transfert entre votre ordinateur et la machine distante depuis que vous avez ouvert le programme. Elle sert aussi à voir la progression de chaque transfert.

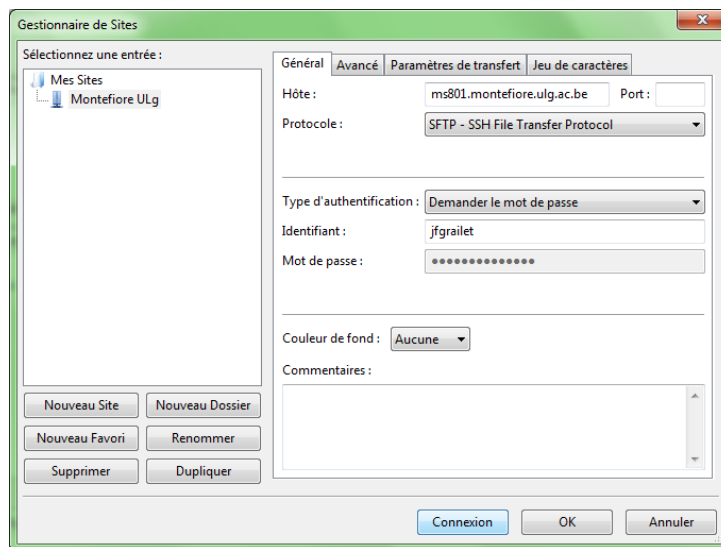
## Etablir la connexion

Cliquez sur le premier bouton sous les menus en haut à gauche (encadré en vert à la capture (q)). Vous devriez voir une fenêtre comme à la figure (s).



(s) Gestion des sites sous *FileZilla*

Cliquez sur “*Nouveau site*” et donner un nom à ce dernier. Ensuite, entrez dans le champ “*Hôte*” la machine de votre choix, et sélectionnez le protocole SFTP dans le menu déroulant juste en dessous. Vous avez ensuite le choix pour la méthode pour demander l’identifiant de connexion et le mot de passe; vous pouvez les sauvegarder tous les deux ou juste entrer l’identifiant (le mot de passe sera demandé à chaque connexion). Dans notre exemple, nous avons choisi de demander le mot de passe à chaque connexion. Cliquez enfin sur “*Connexion*” pour poursuivre. Avant de cliquer sur celui-ci, assurez-vous que la fenêtre est remplie de la même façon qu’à la capture d’écran (t), où nous nous connectons à la machine ms801 du Réseau 8.

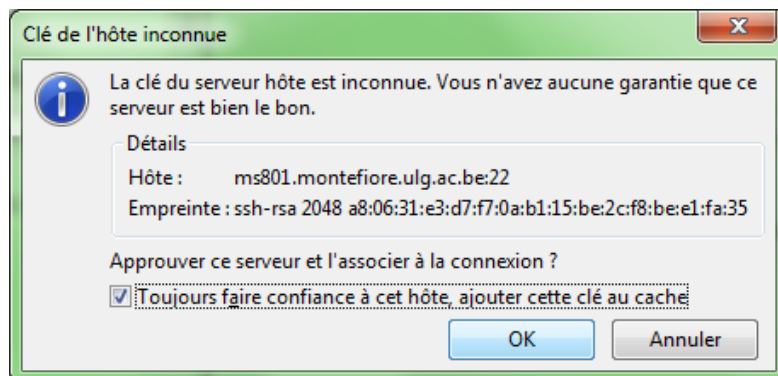


(t) Création d’un nouveau site sous *FileZilla*

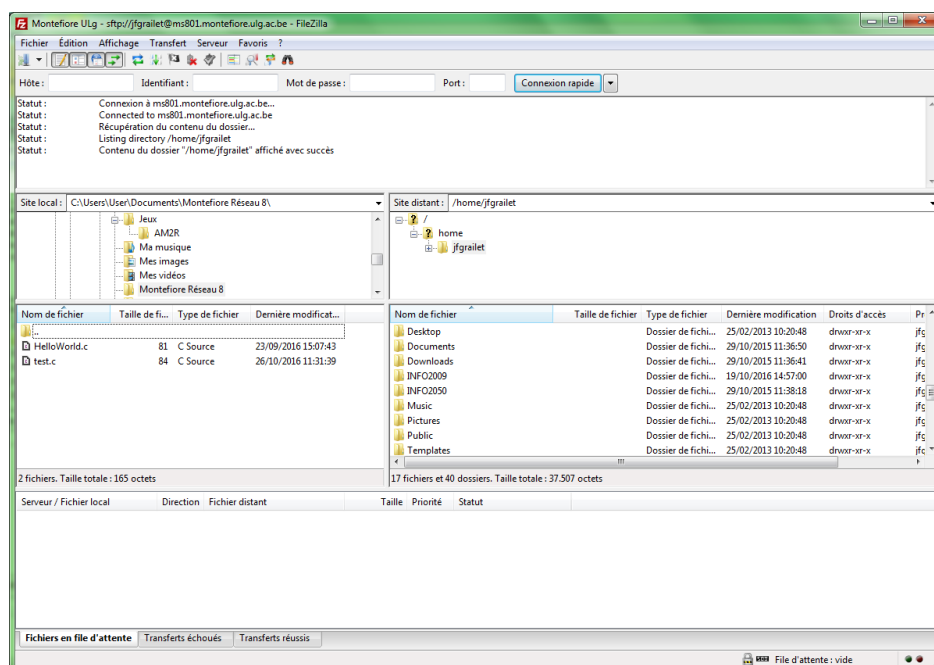
Suivant que vous ayez configuré *FileZilla* pour vous demander le mot de passe ou non, vous devrez peut-être entrer votre mot de passe juste après avoir cliqué sur “*Connexion*”. Ensuite, s’il s’agit de votre première connexion à une machine précise avec *FileZilla*, vous devriez recevoir un avertissement comme illustré à la capture d’écran (u). Cet avertissement est fort proche de celui de *PuTTY* (cf. capture d’écran (n)) au moment de se connecter, si ce n’est que cet avertissement devrait survenir à chaque connexion, à moins de cocher la case “*Toujours faire confiance à cet hôte...*” avant de cliquer sur “*OK*”.

Une fois la connexion établie (visible à partir du journal), vous êtes prêt à envoyer vos fichiers ou à télécharger ceux qui se trouvent sur la machine distante. La capture d’écran (v) montre l’interface de *FileZilla* une fois la connexion établie.





(u) Avertissement de *FileZilla* lors d'une nouvelle connexion

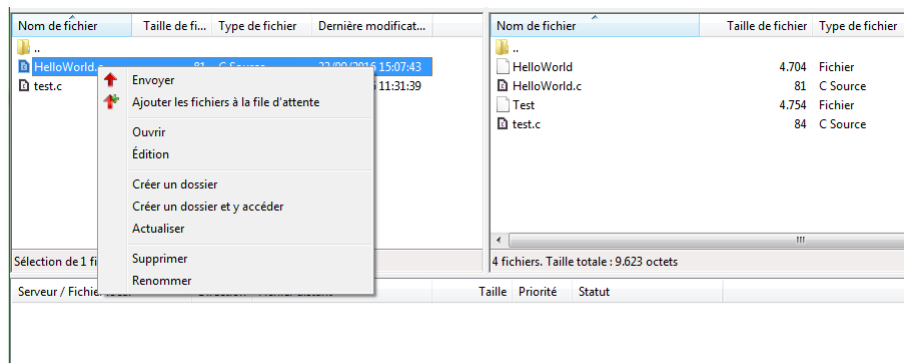


(v) Interface *FileZilla* après une connexion réussie

## Envoyer/télécharger des fichiers

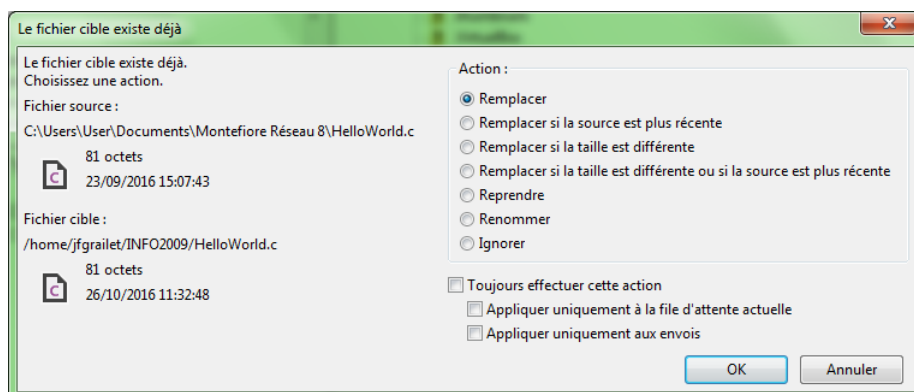
Lorsque la connexion est établie, les zones 3 et 5 (telles que montrées à la capture d'écran (r)) devraient à présent afficher une arborescence de dossiers ainsi que la liste des fichiers d'un dossier courant. En général, sur les machines de l'ULg, le premier dossier sélectionné est toujours votre dossier personnel (c.-à-d. `/home/pml1YYYY/` dans le cas des ordinateurs du B37).

A partir d'ici, l'envoi ou le téléchargement de fichiers entre les deux dossiers affichés est très simple. A gauche (zone 4), cliquez deux fois sur un fichier pour l'envoyer dans le dossier courant de la machine distante. A droite (zone 5), cliquez deux fois sur un fichier pour le télécharger dans votre dossier local courant. En faisant un clic droit sur un fichier ou plus généralement dans une des deux zones correspondant à un dossier, vous aurez un menu déroulant proposant des actions comme la suppression d'un fichier ou la création d'un nouveau sous-dossier dans le dossier courant. La capture d'écran (w) illustre ces zones et un exemple de menu déroulant.



(w) Gestion des envois/téléchargements entre les deux machines

Notez que si vous envoyez un fichier qui existe déjà sur l'autre machine (ou plutôt, qui porte le même nom) afin de le mettre à jour, vous aurez une fenêtre supplémentaire pour demander l'action à réaliser, montrée à la capture d'écran (x). Vous pouvez simplement conserver le choix par défaut: "Remplacer". En cliquant sur *OK*, vous écraserez le fichier distant pour le remplacer par celui que vous envoyez.



(x) Remplacement d'un fichier distant par une nouvelle version

## Annexe 1: quel éditeur de texte choisir ?

Si la plupart des systèmes d'exploitation proposent généralement des éditeurs de texte compatibles avec la programmation, ceux-ci sont généralement rudimentaires et ne proposent pas certaines fonctions qui s'avèrent vite indispensables (comme la coloration du code). Nous proposons ci-contre quelques solutions pour rédiger un programme en C.

### Sous Windows

Une première possibilité sous Windows est d'installer *Notepad++*<sup>2</sup>, un éditeur de texte gratuit qui supporte une pléthore de modes de coloration (un par langage) et embarque des fonctionnalités d'auto-complétion, de recherche et de remplacement, d'affichage/masquage de blocs de code, etc. Il s'agit donc d'un outil puissant qui se révélera très utile même au-delà du cours d'INFO2009.

Une autre possibilité est d'installer *Code::Blocks* (cf. notre tutoriel sur le sujet). En plus de fournir tous les outils nécessaires pour rédiger en langage C, *Code::Blocks* est également utile pour compiler du code sans passer par le laboratoire.

### Sous Mac OS X

Une première approche consiste à installer *Xcode*, un IDE (*Integrated Development Environment*) facilement trouvable sur le Mac App Store et utilisable pour la programmation en C. Comme *Code::Blocks* sous Windows, il permet aussi de compiler du code.

Si toutefois vous n'avez besoin que d'un éditeur de texte, les intéressés pourront se tourner vers *TextWrangler*, également disponible gratuitement sur le Mac App Store. Cet éditeur est souvent considéré comme un des meilleurs équivalents de *Notepad++* (Windows) sous Mac OS X.

### Sous Linux

La plupart des distributions actuelles de Linux vous proposeront d'emblée *gedit*, un éditeur de texte que vous avez probablement déjà utilisé lors des séances de laboratoire. Sans être aussi riche que d'autres éditeurs cités plus haut (comme *Notepad++*), il suffit amplement pour le cours d'INFO2009.

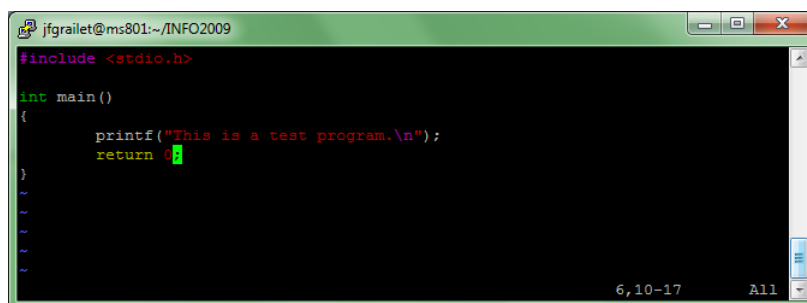
---

<sup>2</sup>Cf. <http://notepad-plus-plus.org/>

## Annexe 2: l'éditeur de texte vim

Si ce tutoriel conseille de passer par un éditeur de texte local et un programme de transfert de fichiers comme *FileZilla* pour écrire son code avant de l'envoyer sur la machine distante, ce n'est en réalité pas indispensable. En effet, il est tout à fait possible d'écrire du code à distance. Les ordinateurs de l'ULg proposent notamment vim, un éditeur de texte qui permet de modifier un fichier texte ou écrit dans un langage de programmation directement à partir du terminal. Comme il est prévu pour le terminal, cet éditeur de texte ne peut être utilisé qu'avec le clavier. Impossible donc, par exemple, d'utiliser la souris pour sélectionner une portion de code et la répliquer. L'étudiant désireux de recourir à vim devra donc apprendre à manipuler les quelques raccourcis claviers qui permettront de naviguer dans le fichier en cours d'écriture et de le sauvegarder. De nombreux tutoriels sur Internet expliquent comment utiliser vim, et nous laisserons donc le soin aux étudiants curieux de se renseigner sur le sujet.

La capture d'écran (y) montre à quoi ressemble l'écriture d'un programme C basique sous vim, à partir d'un terminal créé avec *PuTTY*. La capture d'écran (z) montre quelques commandes utilisées après l'écriture du même programme pour le compiler et l'exécuter.

A screenshot of a terminal window titled 'jfgrailet@ms801:~/INFO2009'. The vim editor is open, showing a C program with the following code: 

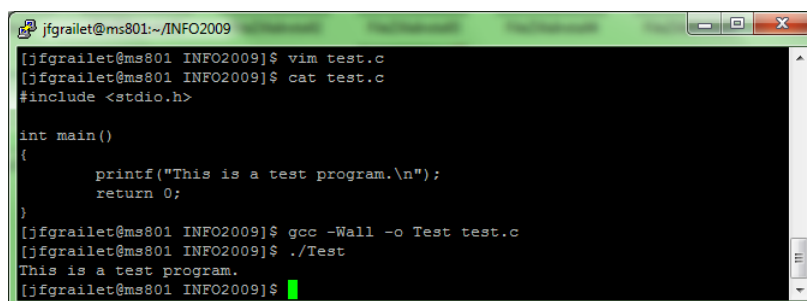
```
#include <stdio.h>

int main()
{
    printf("This is a test program.\n");
    return 0;
}

~
~
~
~
```

 The cursor is at the end of the 'return 0;' line. The status bar at the bottom right shows '6,10-17 All'.

(y) Edition d'un programme basique en C avec vim.

A screenshot of a terminal window titled 'jfgrailet@ms801:~/INFO2009'. It shows the following commands and output: 

```
[jfgrailet@ms801 INFO2009]$ vim test.c
[jfgrailet@ms801 INFO2009]$ cat test.c
#include <stdio.h>

int main()
{
    printf("This is a test program.\n");
    return 0;
}

[jfgrailet@ms801 INFO2009]$ gcc -Wall -o Test test.c
[jfgrailet@ms801 INFO2009]$ ./Test
This is a test program.
[jfgrailet@ms801 INFO2009]$
```

(z) Compilation et exécution du même programme.

## Annexe 3: *X11* et *gedit* à distance

En marge de *vim*, il est également possible d'utiliser *gedit* à distance, de la même manière qu'aux séances de laboratoire, en recourant à *X11*. *X11* est le diminutif de *X Window System Version 11*, ou plus simplement *X Window System*, un protocole réseau conçu pour l'interaction à distance avec des programmes à interface graphique, tel *gedit*. Utiliser *X11* vous permettra donc d'écrire du code directement sur une machine du laboratoire via *gedit*, sans passer par *vim*.

Sous Windows, vous devrez installer un programme qui va agir en tant que *serveur X11*. Nous vous conseillons *Xming* (le programme d'installation est à télécharger sur <https://sourceforge.net/projects/xming/>), qui est à la fois gratuit et très simple à installer (les choix par défaut du programme d'installation étant amplement suffisants). Sur Mac OS X, vous devrez recourir à *XQuartz* (disponible sur <https://www.xquartz.org/>).

Sous une distribution Linux, vous aurez besoin d'installer *xorg* et *openbox* avec la commande suivante:

```
$ sudo apt-get install xorg openbox
```

Une fois le programme adéquat installé, il y a deux choses à faire pour pouvoir utiliser *gedit* à distance:

- Démarrer le serveur *X11* en exécutant le programme associé, par exemple en double cliquant sur *Xming* dans votre menu Démarrer sous Windows (une icône dans la barre de tâches, tout à droite, signalera que le serveur est en activité). Sous Linux, vous devrez simplement exécuter la commande *openbox*.
- Autoriser le *forwarding X11* avec SSH. Avec la commande *ssh* (Mac OS X et Linux), il suffit d'ajouter l'option *-X* à votre ligne de commande au moment d'établir le terminal à distance. Sous Windows, avec *PuTTY*, avant d'établir la connexion, vous devrez dérouler le menu *Connection*, et dans la nouvelle liste, dérouler à son tour le menu *SSH*. Cliquez sur l'item *X11* et cochez la case "*Enable X11 Forwarding*". Il ne vous reste plus qu'à démarrer la connexion.

Une fois ces deux étapes accomplies, vous pourrez utiliser *gedit* pour écrire du code sur la machine distante, de la même manière qu'aux séances de laboratoire, bien que le fait d'agir à distance rendra l'éditeur sensiblement moins réactif à l'usage.

## Annexe 4: principes de la connexion à distance

La connexion à distance au laboratoire repose sur un protocole réseau : le protocole SSH (pour *Secure Shell*<sup>3</sup>). Il s'agit d'un protocole de communication grâce auquel on peut ouvrir un terminal sur une machine distante pour utiliser des commandes sur celle-ci. Il est dit *Secure* car il utilise des techniques de chiffrement pour empêcher un intermédiaire, qui collecte d'une façon ou d'une autre les paquets de données échangés entre les deux machines, de connaître les commandes utilisées.

Pour utiliser SSH, il suffit de disposer dans son ordinateur personnel d'un *client* SSH, c.-à-d. un programme qui implémente le protocole dans le sens de l'utilisateur vers la machine distante, c.-à-d. le sens de la demande. Un programme qui va plutôt implémenter le protocole dans le sens de la réception et du traitement des demandes sera désigné comme étant un *serveur* SSH<sup>4</sup>.

Ce premier programme ne constituera cependant qu'une partie de la solution si on désire compiler un programme qui est déjà écrit: en effet, il faut pouvoir envoyer son/ses fichier(s) source sur la machine distante. Pour ce faire, nous utilisons le transfert de fichiers par SFTP (pour *SSH File Transfer Protocol*<sup>5</sup>), un protocole de transfert de fichiers basé sur SSH. A noter qu'il existe également un protocole FTP (pour *File Transfer Protocol*), mais qui ne fonctionne pas exactement de la même manière. Le protocole SFTP n'est donc pas une extension du protocole FTP. *FileZilla*, le programme que nous avons précédemment proposé pour le transfert de fichiers, implémente aussi bien le SFTP que le FTP.

En résumé, notre solution combine l'utilisation d'un *client* SSH et d'un *client* SFTP. De leur côté, les ordinateurs de l'ULg sont configurés pour répondre aussi bien aux requêtes SSH et SFTP, ils utilisent donc des *serveurs* SSH et SFTP.

---

<sup>3</sup>Plus de détails sur Wikipédia : [https://fr.wikipedia.org/wiki/Secure\\_Shell](https://fr.wikipedia.org/wiki/Secure_Shell)

<sup>4</sup>Plus de détails sur Wikipédia : <https://fr.wikipedia.org/wiki/Client-serveur>

<sup>5</sup>Cf. Wikipédia: [https://fr.wikipedia.org/wiki/SSH\\_File\\_Transfer\\_Protocol](https://fr.wikipedia.org/wiki/SSH_File_Transfer_Protocol)