COMP202 from Windows

Len Hamey
len.hamey@mq.edu.au

Revised 19 July 2017

Introduction

The unit COMP202 incorporates practical assignments that are performed in the Linux Unix environment on the servers ash.science.mq.edu.au and iceberg.science.mq.edu.au. This document outlines how you can work effectively on these servers from a Windows environment, both from your own Windows machine (assuming Internet connectivity) and from the Windows machines in the lab.

Putty – a command-line interface to ash or iceberg

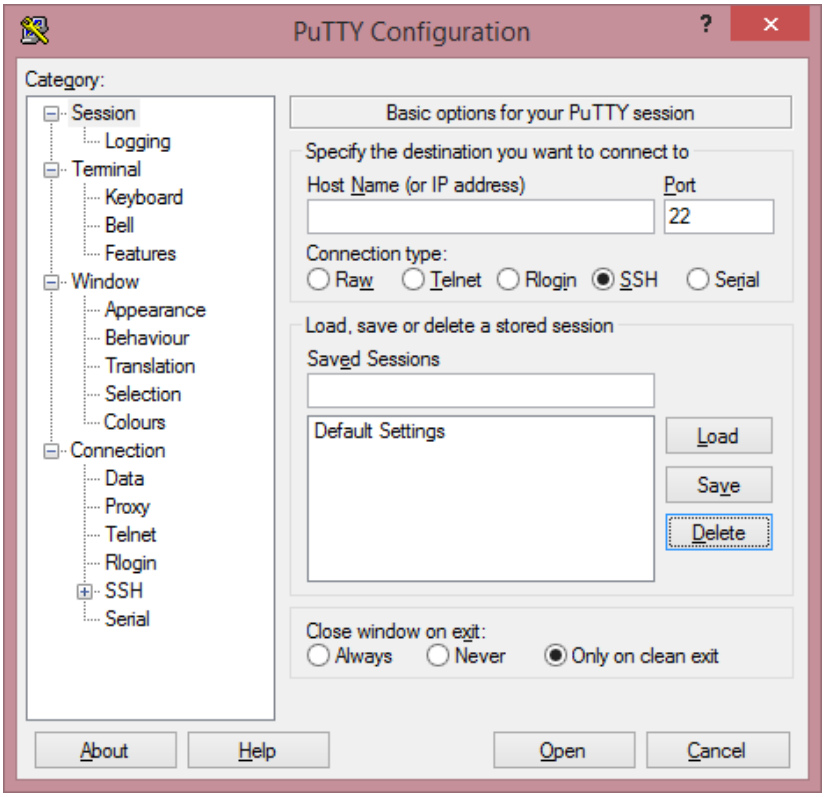
Putty is a Windows application that allows you to connect to a remote server and interact using a command-line interface. This is the basic interface for Unix, and you are expected to become familiar with it.

Putty is, in fact, a secure shell (ssh) client. Ssh is a network protocol that allows remote login to Unix servers. There are other ssh clients but we will focus on Putty because it is widely available, well supported and free.

> If you are wanting to connect from your own Windows machine, then you may need to install Putty yourself. You can find out about installing Putty on the Internet.

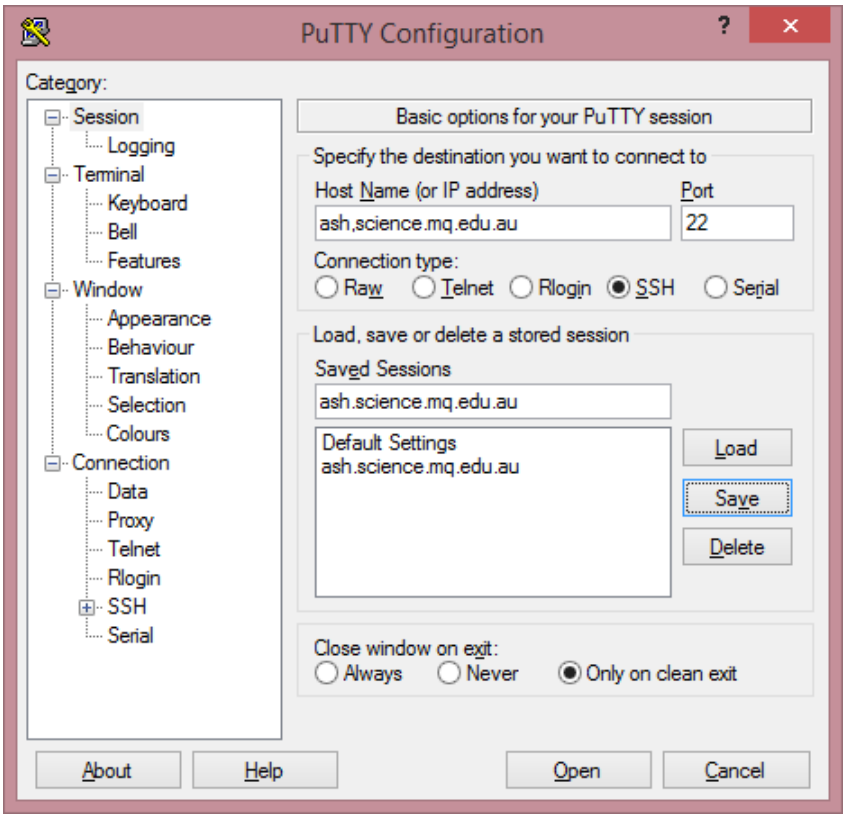
Connecting with Putty

Start Putty. You will see a screen similar to the following:



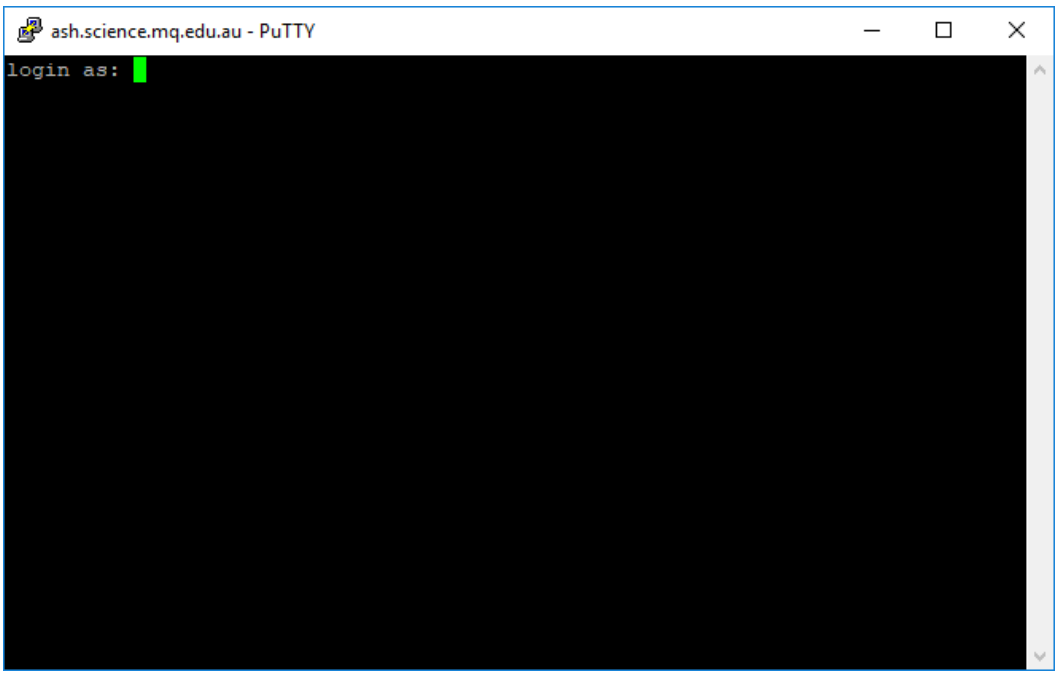
Enter the host name ash.science.mq.edu.au or iceberg.science.mq.edu.au.

Type the name ash.science.mq.edu.au or iceberg.science.mq.edu.au in the “Saved Sessions” box and click “Save”. This remembers the server so that in future you can click on the settings for ash or iceberg and click Load instead of typing the host name.



Now click “Open” to open a session with the server that you have selected.

You will see a “terminal screen” that can display lines of text. It will be asking you to enter your login name as in the following screenshot.

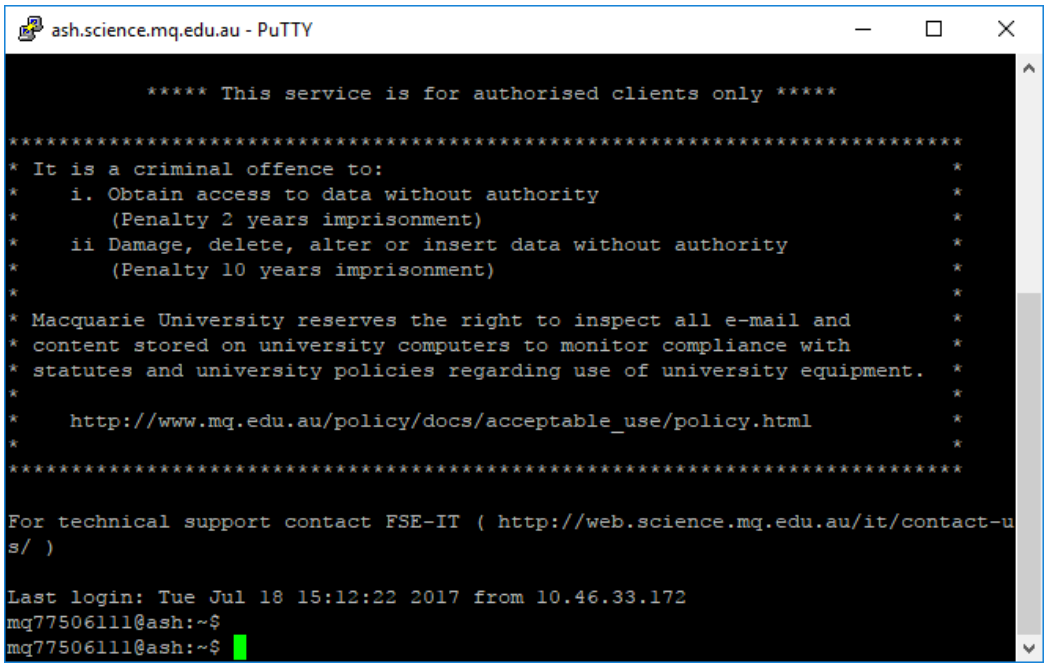


Type your user name (your student number) followed by the Enter key, and then type your password followed by the Enter key, and it will log you in. You will notice when you type your password that nothing that you type appears on the screen. This is a security feature. Modern systems show you dots or stars for each character of a password that you type, but Unix shows you nothing at all. If you make a typing mistake in your password, you will not be logged in.

> **Warning**: You have three attempts to type your password correctly before you are locked out, so be careful.

Unix command-line interface

Once you are logged in, you will see something similar to the following:



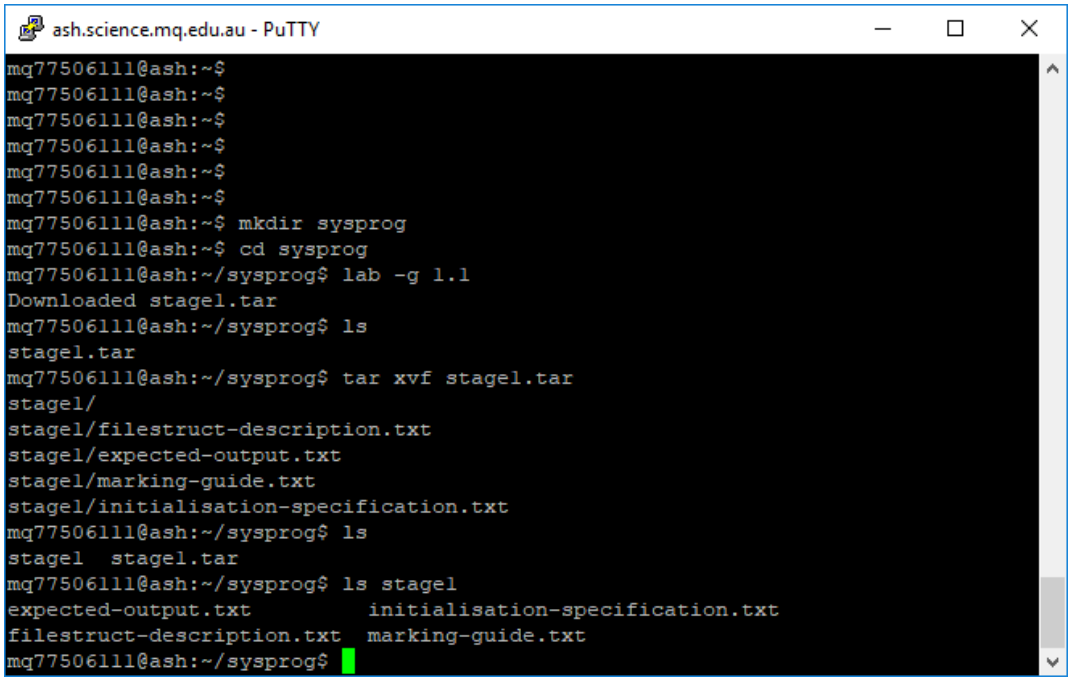
You can now type Unix system commands at the prompt "*user*@ash:~$ " and the Unix system will respond by displaying text in the terminal window.

> **An aside about the prompt.** The system prompt shows your logged in user ID, the short name of the host and the current working directory name. This can be helpful if you have a few windows open at once, working on different machines or in different directories.

For example, you may type the `ls` command to list files in your directory.

> **An aside about colours**: The `ls` command may use colours to highlight different types of files. Directories are shown in blue which can be hard to see, but you can change the colour of ANSI Blue in Putty's Change Settings dialog box. Set the Red and Green components to 70 to make the blue lighter. Then save those settings so that they are used every time that you log in to the same server. You will need to do this for each server that putty is set up for.

For example, the following screen shot shows the process of creating a working directory for the Systems Programming unit (`mkdir` command), changing into that directory (`cd` command), fetching the first stage of assignment 1 using the `lab` command and unpacking it with the `tar` command. For more information on using the `lab` command, see the *Lab Command Manual*.



Unix commands have a simple structure. The command name comes first – it is usually the name of a program that you want to run.

> Unix finds commands by looking through a collection of directories called a "path". The `ls` command actually lives in the file `/bin/ls`, while the C compiler gcc lives in `/usr/bin/gcc`. It does not matter because both `/bin` and `/usr/bin` are usually in your path. However, you can change the path environment variable, and that may mean that Unix will use a different program for `ls`, or it may not be able to find `ls` at all.

After the command name, you provide the command parameters. These usually consist of options, which commence with a hyphen ('−'), and file names. However, some commands behave differently. In the example above, the `tar` command accepts the options `xvf` without a leading hyphen, while the `lab` command requires the hyphen for the option `-g` but accepts a *lab.stage* parameter where a file name would be expected by other commands.

In fact, the command parameters are arbitrary text strings that are passed to the command program, and it is entirely up to that program how they are interpreted. Putting hyphens in front of options is only a convention that is followed for convenience and consistency.

Opening and editing text files with WinSCP and Notepad++

Your assignment task requires you to write a C program and it requires you to view information in the text files that are provided to you. You can do both of these tasks with a good editor.

The most direct way of editing files on Unix is with `vi`. The editor `vi` operates on a 'terminal screen' which is exactly what you have with Putty. However, `vi` has a rather steep learning curve and entirely keyboard driven. You can learn about `vi` from the Internet.

Editing in the lab using Notepad++ without WinSCP

If you are working in the Computing lab, your files on the file server (which happens to be called Claudius) can be access both on your PC and on the server ash or iceberg. This means that you can use a Windows text editor to edit your files, which you will find on the H: drive. Create a directory for your COMP202 lab work on the H drive, and edit the files in that directory on the PC. We suggest Notepad++ as your editor – the files should be in Unix text file format (see below) and Notepad++ can edit Unix text files safely. If you create new files in the PC using something other than Notepad++ then you may need to use Notepad++ to convert them to Unix format. This conversion can be done with the menu item Edit → EOL Conversion → UNIX/OSX Format in Notepad++.

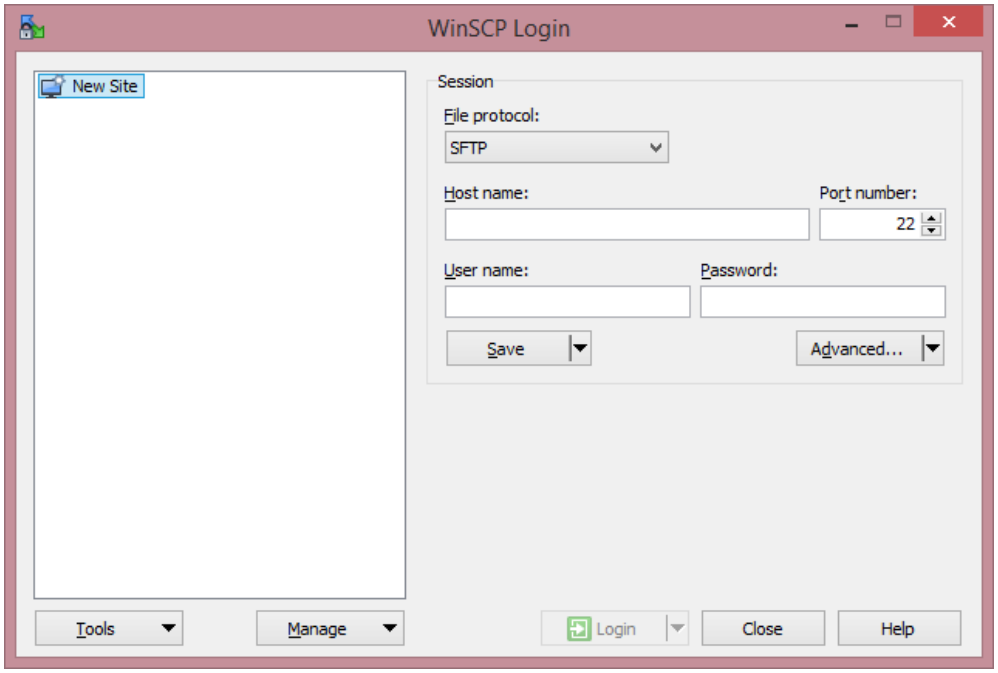
If you work in the lab but also on your own computer, you can ensure a consistent work environment by using WinSCP in the lab as well as on your own computer. See the following section.

Editing on your own Windows computer using WinSCP and Notepad++

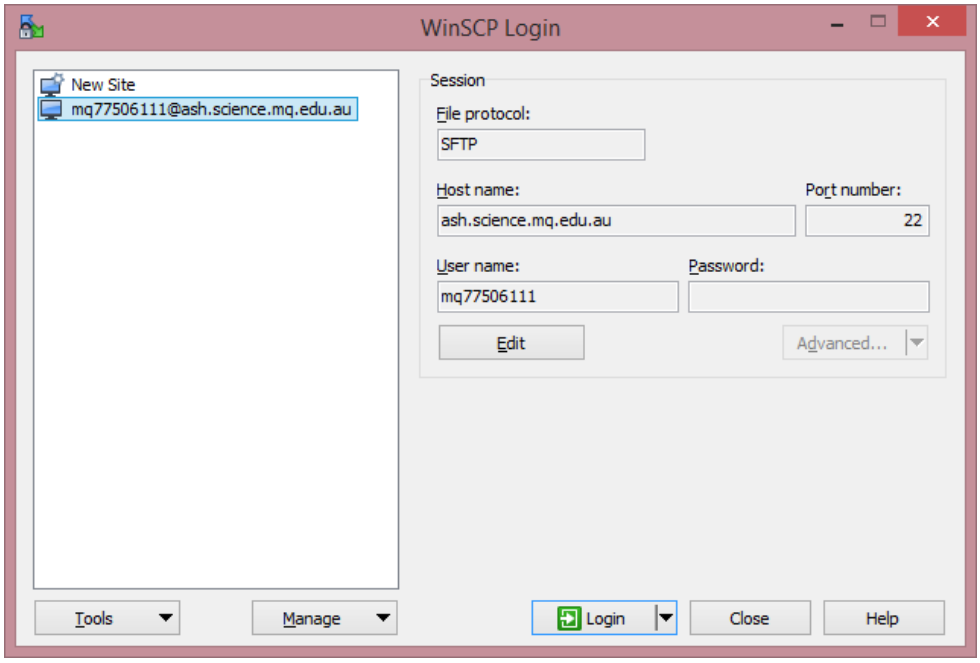
If you are working on your own computer, your files are not shared with the server ash or iceberg. However, the utility WinSCP can be used to remotely edit files on the server.

WinSCP (Windows Secure Copy) is a Windows client for the scp network protocol. Scp is a protocol that allows a computer to copy files to and from a remote server. You can use WinSCP to download files from ash or iceberg, then you can access them locally on your computer. However, WinSCP has an editing capability that makes editing remote files easier.

When you start WinSCP for the first time, you will see a screen similar to the following.

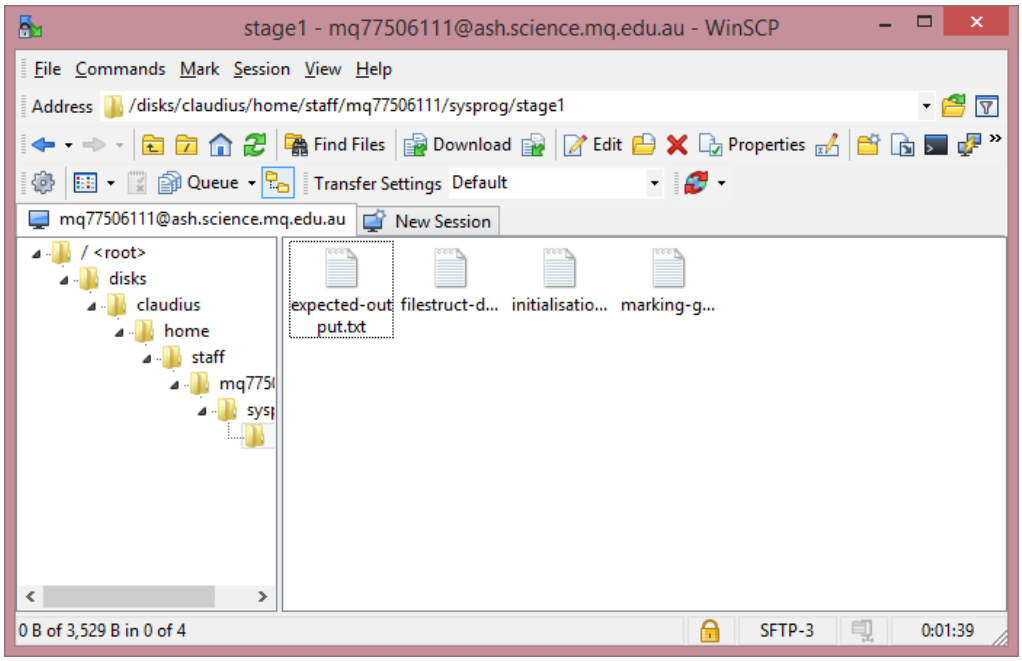


Type the name of the server iceberg.science.mq.edu.au or ash.science.mq.edu.au in the Host name box, and type your user name in the User name box. For security reasons, do not enter your password into the Password box. Click save, and click OK to the dialog box that appears. WinSCP will now have a record of how to log in as you to the server. In future, you will only need to select the site you want to log in to in the panel on the left.

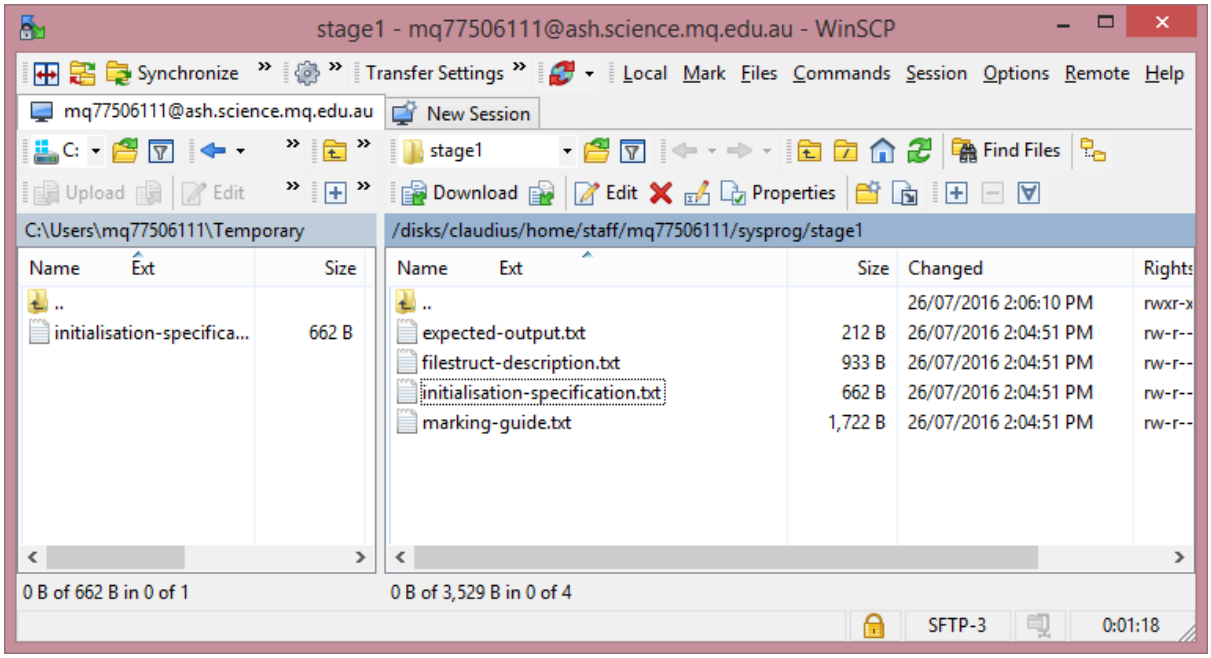


Now click the Login box. Enter your password into the dialog that appears and you will be logged in with WinSCP.

The appearance of the WinSCP window can be customised. If you are using the "Explorer" interface, you will see a windows styled after Windows Explorer, with the directory tree and the files as in the following example. You can edit a file by right clicking on it and selecting Edit. You can also drag and drop files between WinSCP and Windows Explorer (or other Windows programs) and WinSCP will copy them across the network to or from the remote server.



If you are using the “Commander” interface, you will see a window with two panels as shown below. The left panel shows the files in a local folder on your PC and the right panel shows the files in a directory on the remote server. You can change directories by double clicking on the Up folder icon to go to the parent directory, and you can copy files between the remote server and the PC by dragging them between the two windows.



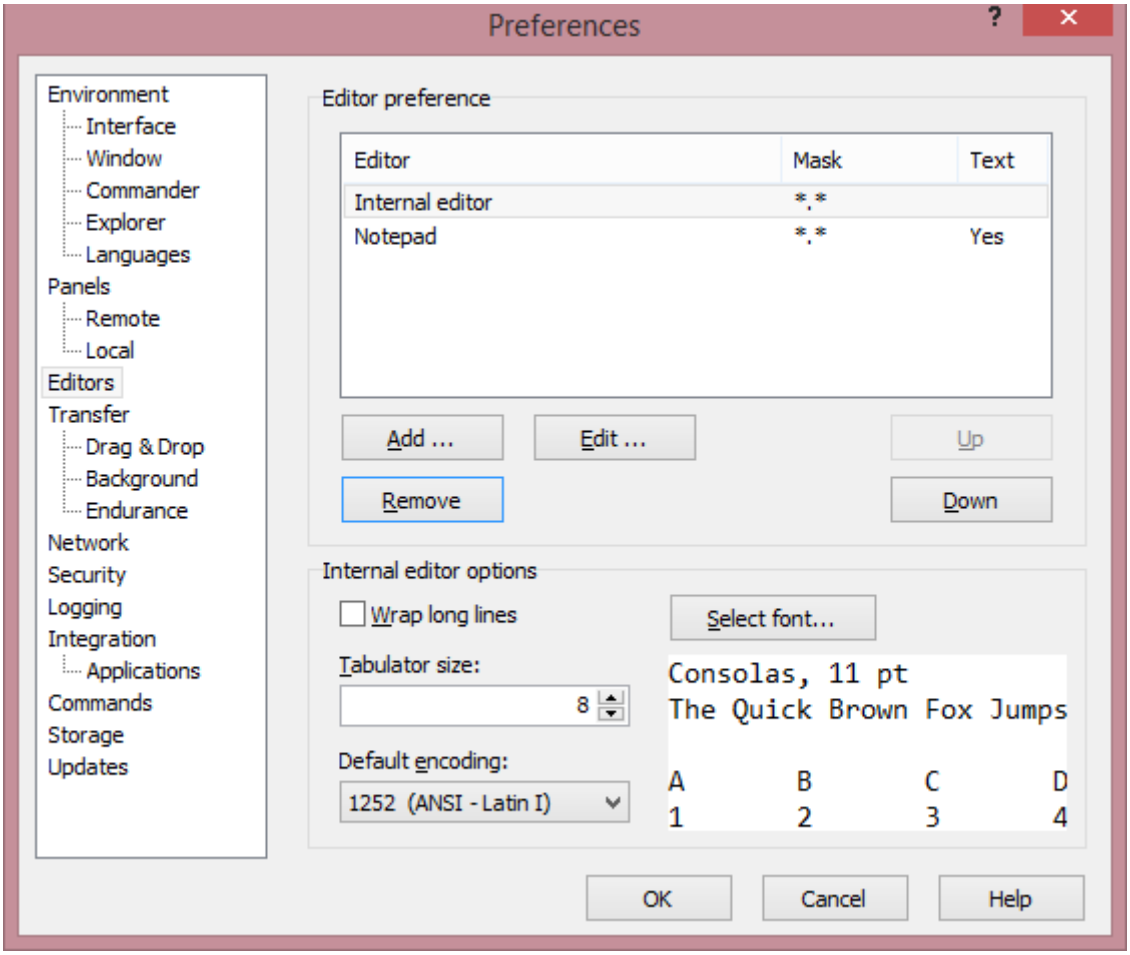
With either interface, you can edit a file on the remote server by right clicking on the file and selecting “Edit”. WinSCP downloads the file to a temporary folder on your PC, and whenever you save changes in the editor it automatically sends those changes back to the server.

Choosing an editor

The editor that you use on your PC must be capable of editing Unix text files. A good choice is Notepad++. Notepad++ has syntax highlighting that helps you with a variety of programming languages, and it can work with Unix formatted files. It can also convert MSDOS text files to Unix.

To select your editor for WinSCP, go to the Preferences dialog box. You will find it under Options in the Commander interface, or under View in the Explorer interface.

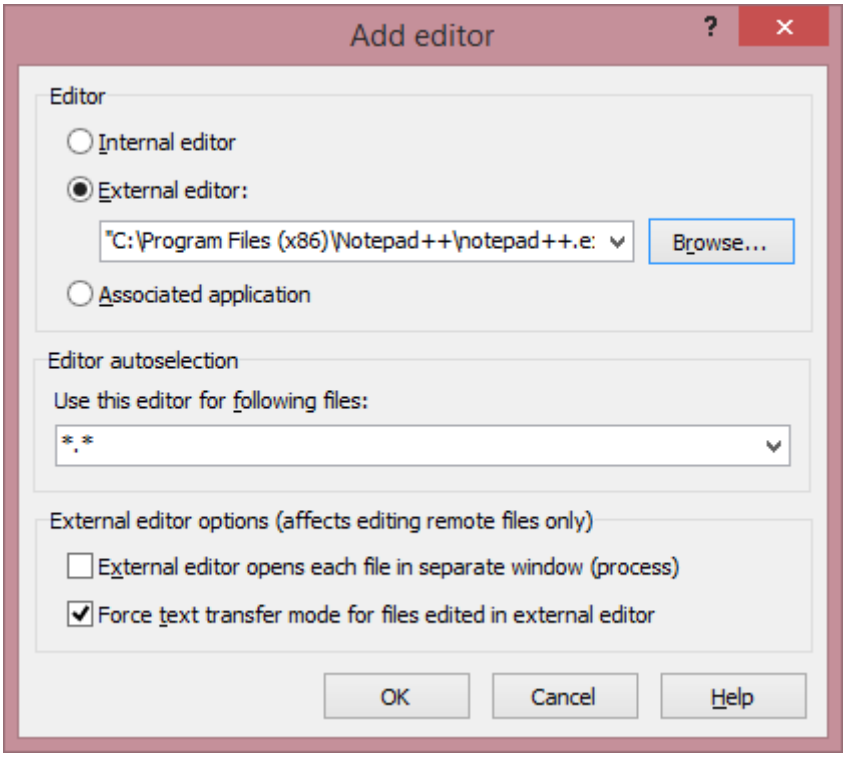
Select Editors in the menu on the left side of the Preferences dialog. Here is what you should see.



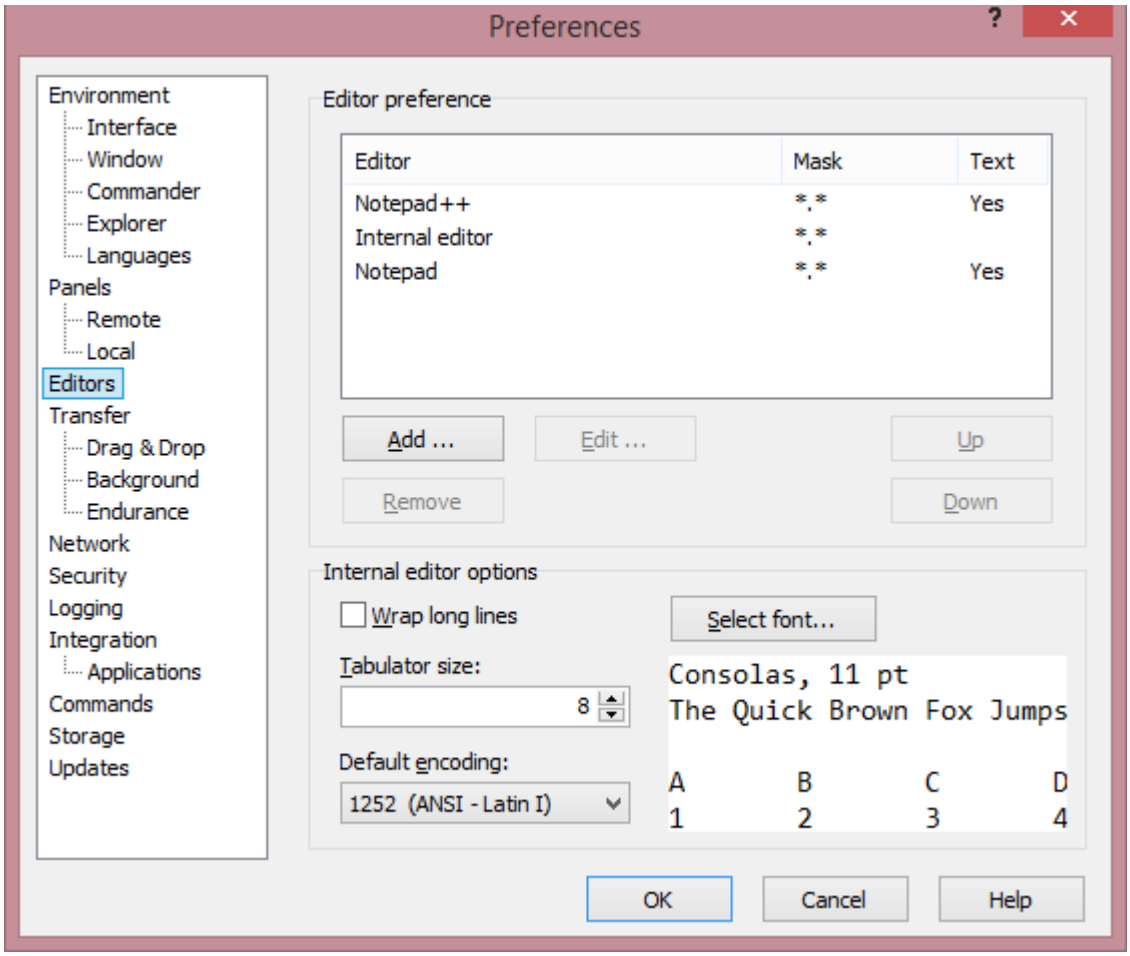
In order to use Notepad++, you need to have it installed on your system and you need to add it as the preferred editor in WinSCP's Preferences dialog.

To add Notepad++ to WinSCP:

1. Click Add… and a dialog box will appear.
2. Select External editor and Browse to find the Notepad++ executable.


Add editor
Editor
Internal editor
External editor:
"C:\Program Files (x86)\Notepad++\notepad++.e:
Browse...
Associated application
Editor autoselection
Use this editor for following files:
.
External editor options (affects editing remote files only)
External editor opens each file in separate window (process)
Force text transfer mode for files edited in external editor
OK
Cancel
Help

3. Click OK.


Preferences
Environment
Interface
Window
Commander
Explorer
Languages
Panels
Remote
Local
Editors
Transfer
Drag & Drop
Background
Endurance
Network
Security
Logging
Integration
Applications
Commands
Storage
Updates
Editor preference
Editor
Mask
Text
Notepad++
.
Yes
Internal editor
.
Notepad
.
Yes
Add ...
Edit ...
Up
Remove
Down
Internal editor options
Wrap long lines
Select font...
Tabulator size:
8
Consolas, 11 pt
The Quick Brown Fox Jumps
A B C D
1 2 3 4
Default encoding:
1252 (ANSI - Latin I)
OK
Cancel
Help

4. Notepad++ should be at the top of the list. If it is not, select it and use the Up button to move it to the top.
5. Now click OK to finish selecting Notepad++ as your editor.

Once you have done these steps, you can right click on a file in the main window, select Edit and you will be editing it with Notepad++.

If you edit a remote text file, WinSCP will convert it to MSDOS format while Notepad++ is editing it, and convert it back to Unix format when it is uploaded to the server.

A text file is a text file, right? Not quite. Way back in the dark ages (1981) when MSDOS was born, Microsoft decided that each line of text in a text file would have two characters at the end: CR (Carriage Return) and LF (Line Feed). These two characters were then important for terminals that printed on paper – the CR character sent the print head back to the start of the same line and the LF character scrolled the paper up one line. So, MSDOS files could be printed on such paper terminals easily.

Meanwhile, the inventors of Unix in the 1970's had decided that each line of a text file should have one character at the end. They called it "newline" (in C it is represented as '\n') but it was actually the LF (Line Feed) character. The Unix developers wrote a driver for paper terminals that would handle the LF character in a text file by actually sending CR to the terminal followed by LF. So Unix text files could be printed on paper terminals easily.

So a great divide was made that continues to this day: Text files on MSDOS have CR LF at the end of every line while text files on Unix have LF at the end of every line. Whenever a text file is copied from MSDOS to Unix, the software should remove the CR characters, and mostly it does so. And whenever a text file is copied from Unix to MSDOS, the software should add the CR characters before each LF, and mostly it does so. But sometimes things go wrong and an MSDOS format file ends up on Unix or a Unix format file on MSDOS.

If Unix encounters an MSDOS formatted text file, it usually objects to the CR characters at the end of each line. If you open such a file in vi, it will show you the CR characters as ^M which represents Ctrl-M.

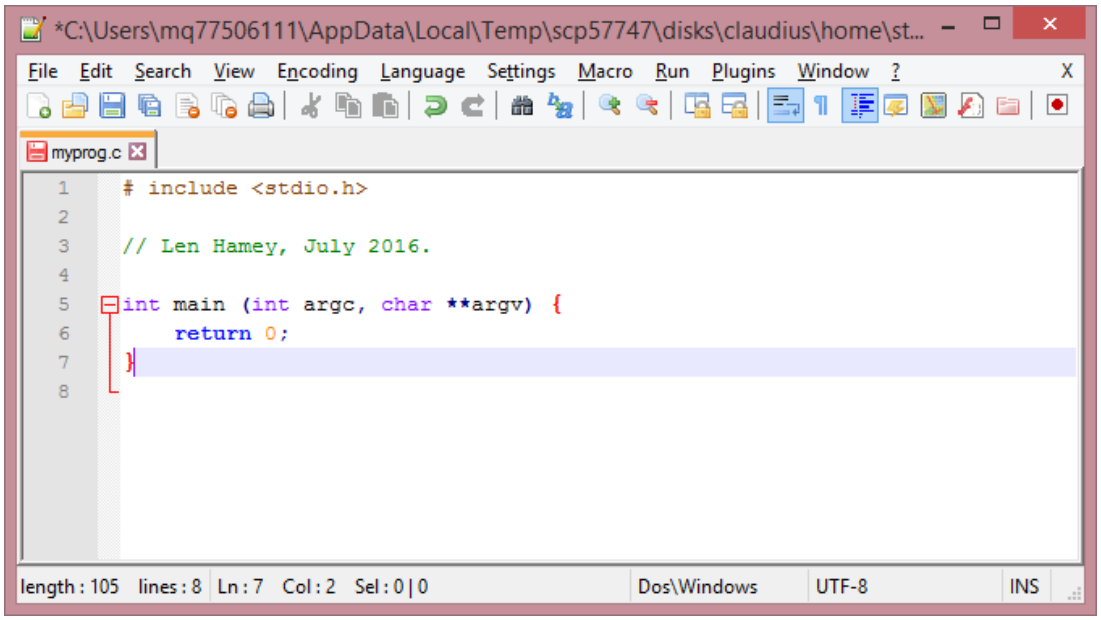
If MSDOS encounters a Unix formatted text file, it depends on the particular program as to what happens. Compilers are very forgiving because software often appears in Unix format. Many editors will also understand Unix format, but Windows standard Notepad will show the entire file as a single line with funny blocks where the Unix LF characters appear.

You should not think that this problem is only between Microsoft and Unix. Back at that time there were many different operating systems that had different ways of representing text files. Some systems even had different text file formats for different programming languages! Compatibility problems were widespread. Many of those systems have died out, and their file formats are no longer relevant. But Microsoft and Unix seem to be here to stay.

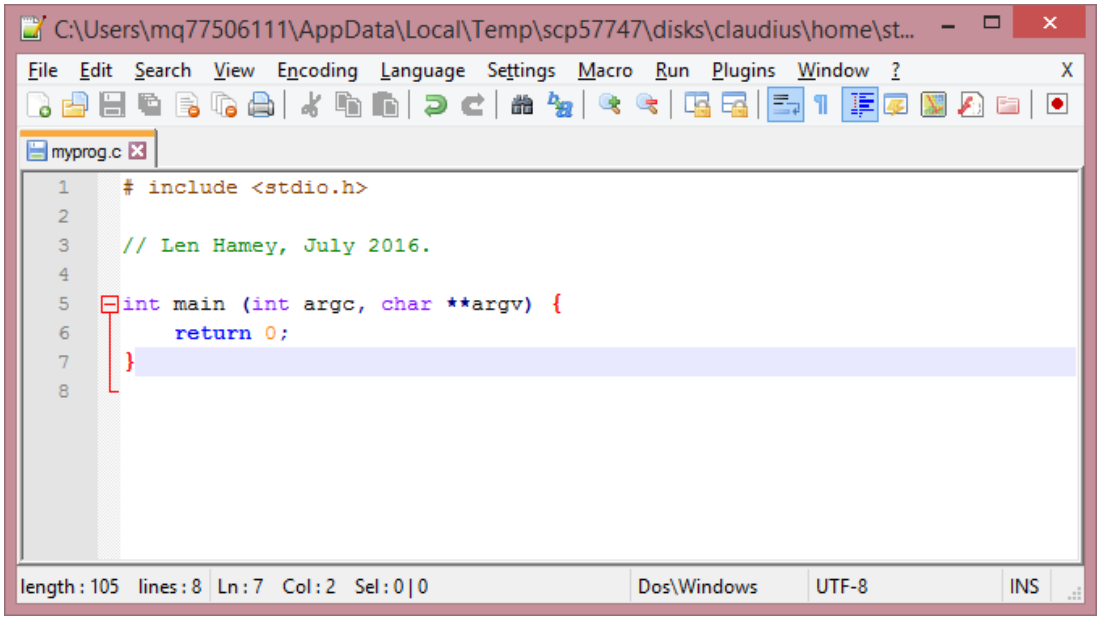
Compiling and running your program

There is nothing specific to Windows about compiling and running your program on the remote server. You use Putty as your terminal window to execute gcc for compilation and to run the program. There are a couple of things you should remember.

1. The files on the remote server are not updated until you save your edits from Notepad++. If you compile your program without saving your changes, there will be no warning. All that will happen is that you will compile and run an old version of the program. Then, when you look at the Notepad++ window you will see that Save and SaveAll buttons (second and third from the left) are blue, meaning that something needs to be saved. Also, the file that has not been save has a red save icon next to its name as shown in the following screenshot.



In that case, press the Save or SaveAll button and Notepad++ will save your changes and grey the buttons indicating that nothing needs to be saved. Then you should recompile your program to run the latest version.



2. Saving across the network to the remote server is not as fast as saving a file on your own computer. This is usually not a problem, but if the files are large or the Internet is slow you may find that your file is not updated immediately on the server. This means that you might need to allow some time before compiling your program. If you compile too early you may get the wrong code version compiled, or you may get syntax errors if the file is partially transferred. A good solution is to use the system utility `make` to compile your programs – it knows when your source file is more recent than the executable program and will recompile it.

Because there is nothing Windows-specific about using gcc or running your compiler through Putty, we won't discuss those topics here.

After you are finished, make sure that your files are saved and close Notepad++ before you exit WinSCP. If you exit WinSCP without saving edits from Notepad++, an error will result and you could lose your changes.

After you are finished, use the Unix `logout` command to terminate your terminal session in Putty.