Introduction to the Unix command line

History

Many contemporary computer operating systems, like Microsoft Windows and Mac OS X, offer primarily (but not exclusively) graphical user interfaces. The user interacts with files and programs by pointing and clicking with a mouse. Unix, which is a considerably older operating system than Windows or OS X, also offers a graphical user interface (more than one, actually), but Unix originally presented a *command line interface*, in which the user interacts with the operating system by typing commands at a text prompt and reading textual output from programs. The command line interface is alive and well in Unix today, and many Unix users consider it the most powerful and efficient system for basic operations like navigating through directories and handling files.

Terminology

In this document, we use the term Unix to refer to a family of operating systems. The look and feel of the graphical interfaces of these systems, as well as the file system layout, may differ significantly from one system to the next, but they all have their (conceptual) roots in the original Unix developed at Bell Labs. This family of operating systems includes Linux (also called GNU/Linux) distributions like Debian, Red Hat, and Ubuntu, as well as Sun Solaris. Mac OS X is a curious case; many of its users never touch the command line interface, but all the commands described below are all available in OS X. Many entertaining arguments about whether OS X is a true Unix can be found in programmer internet forums. Microsoft Windows offers a different command line interface than Unix, but a program called Cygwin provides a Unix interface for Windows.

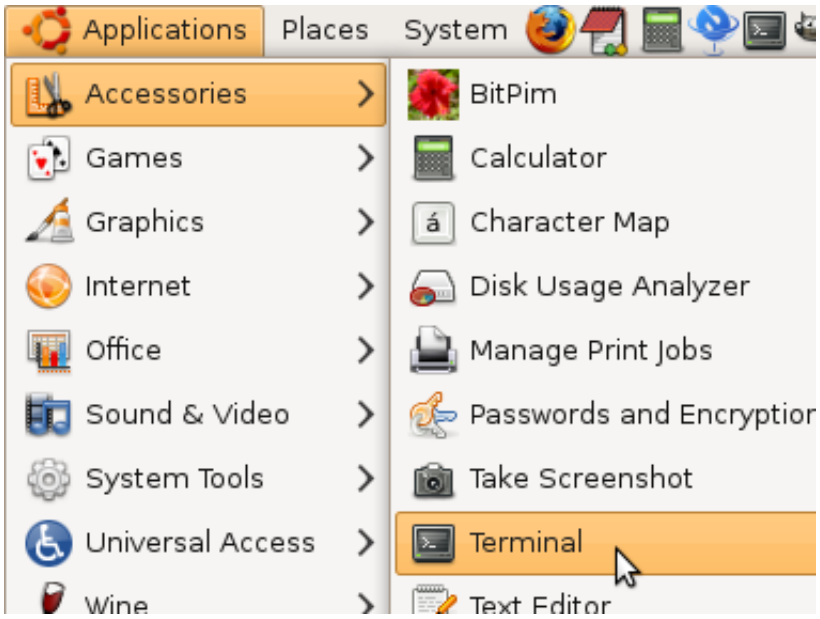
We also use the abbreviation OS to mean “operating system”.

Opening a Terminal

In a modern Unix OS (that is, one with a graphical user interface) the command line interface is accessed through a program called a Terminal. Below we show how to run a terminal program in some of the more popular graphical Unix environments, including GNOME and KDE, available in most Linux distributions, as well as Mac OS X.

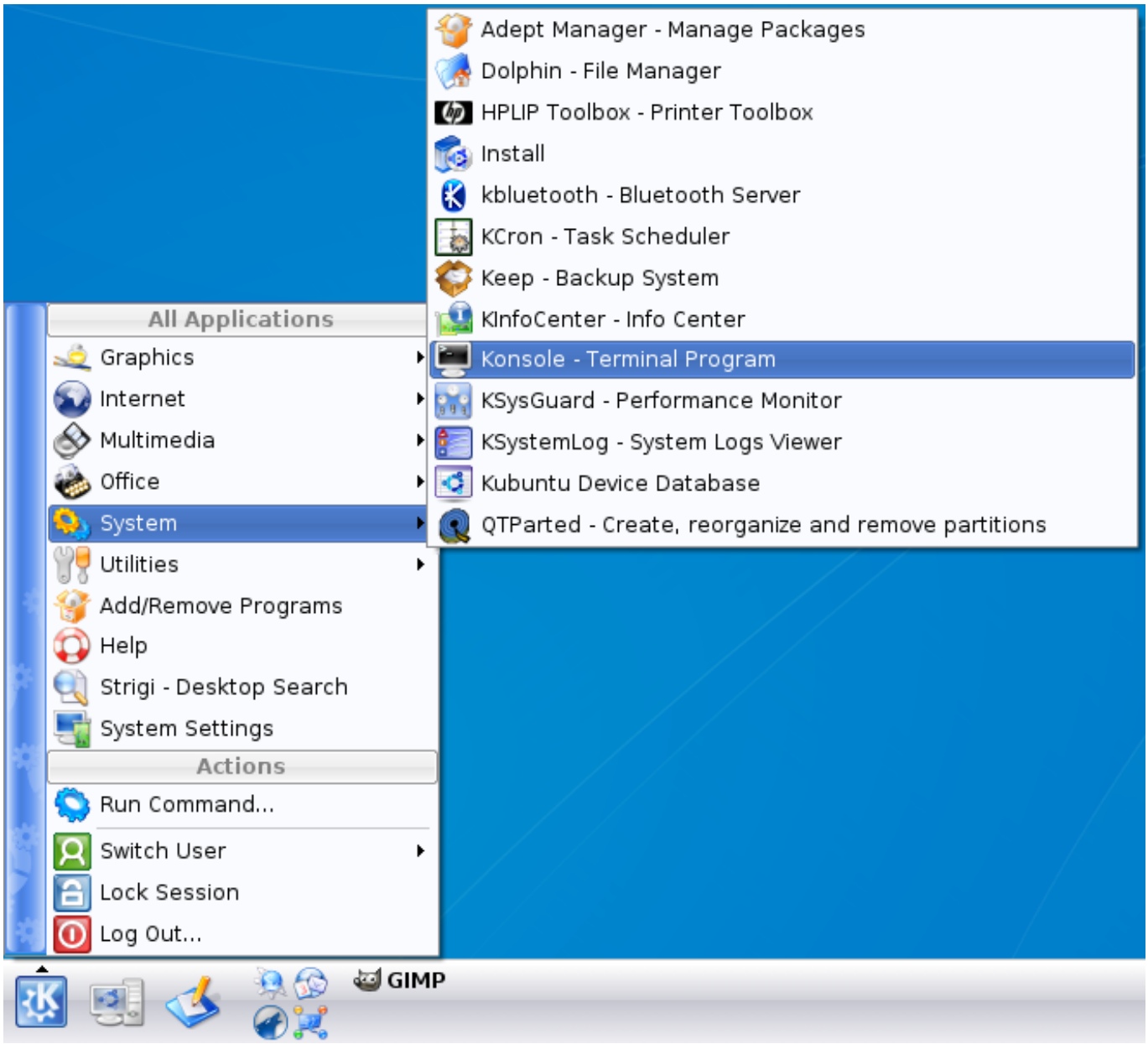
GNOME

In the GNOME desktop, click the Applications menu (typically in the upper left corner); in the Accessories menu, choose Terminal.



KDE

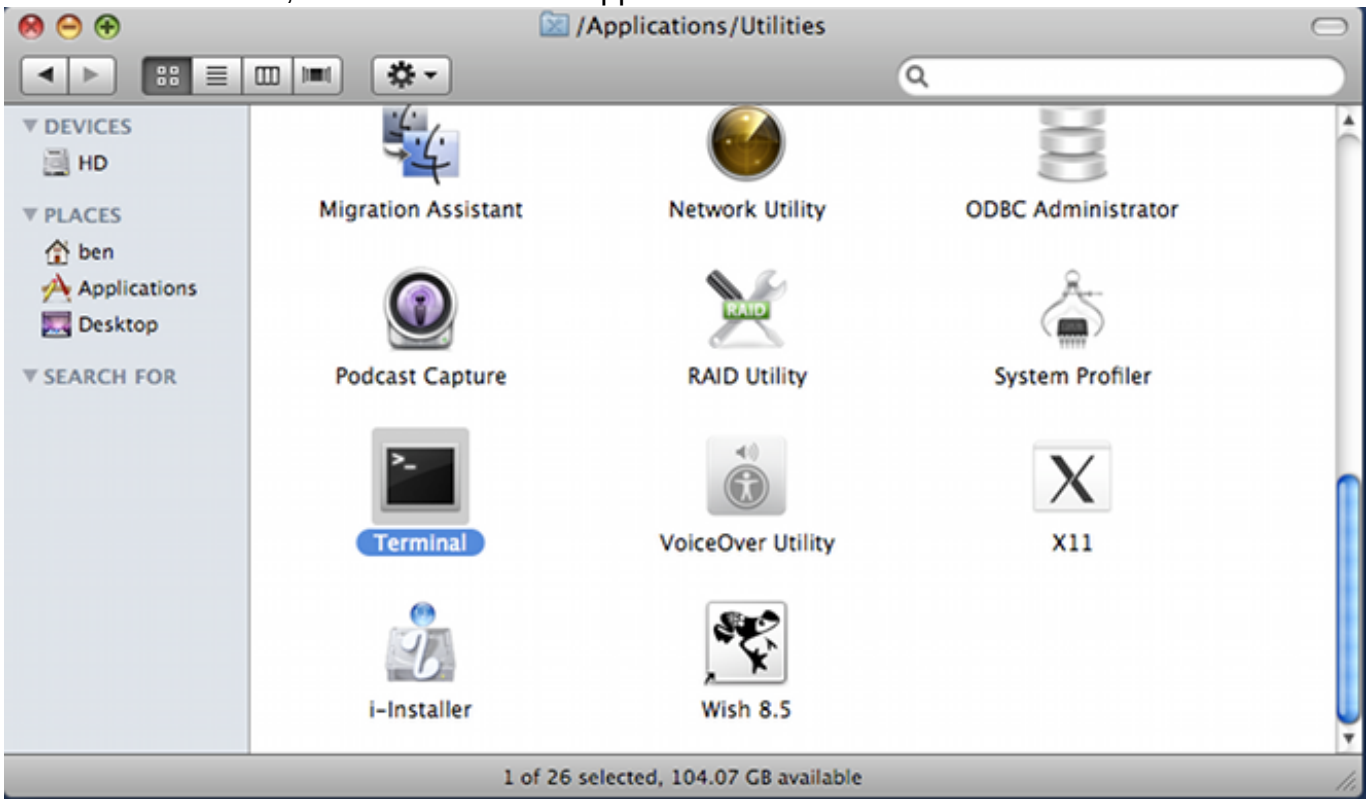
In the KDE desktop, click the main menu (with K superimposed over a gear), typically in the lower left corner; in the System menu, choose Konsole (the KDE terminal program).



Mac OS X

In a Finder window, choose the Utilities folder.

In the Utilities folder, select the Terminal application.



A new window

The steps listed above should produce a new window on the screen, with a prompt and possibly a cursor. The prompt might be a simple character, like **$** or **%**, or it might be the name of the computer and a number showing how many commands have been entered, like coates[1]. Regardless of the color of the window or cursor or the wording of the prompt, this terminal window is a Unix command line interface.

Shell

The prompt in a terminal window is the output of a program called the shell. The shell is the Unix command interpreter. (It is actually a separate program from the terminal window program, but the shell starts automatically when a terminal window opens and exits when the terminal window closes). Unix offers several different shells, but for most of what we cover in this document, the differences between the shells are negligible.

The shell reads the characters typed next to the prompt in the terminal, until the user hits either the Enter key or the Return key. After Enter or Return, the shell interprets all the input characters as a command, runs the command, returns any text output from the command to be displayed in the terminal, and, finally, displays a new prompt, waiting for the next command.

File names and directories

Since most of the commands below involve files and directories, we should consider the rules for naming files and directories. Unix file names may contain letters (a-z, A-Z), numbers (0-9), and a few other characters, like the dot (.) and underscore (_). Certain characters should *not* be used in a file name, like slash (/), asterisk (*), ampersand (&), percent (%), and space ( ). These characters have special meaning in the Unix file system, or to the shell, and should be avoided in file names. Files whose names begin with a dot (.) are handled specially by some of the commands considered below.

Unix files live in directories, which are analogous to folders in Microsoft Windows. A directory is simply a collection of files. All directories exist in a hierarchy, which begins at the *root directory*, which is denoted by a slash (/). The *full path* of a file indicates its position in the directory hierarchy. For example, the full paths /file1 and /file2 refer to two separate files, file1 and file2, which both reside in the root directory (hence the / in the full path). The full path always begins with /. As another example, consider the full paths /dir1/file1 and /dir1/dir2/file2. These paths refer, again, to two separate files, but in this case the files exist in different directories. The first file, file1 lives in the directory dir1, which lives in the root directory /. The second file, file2, lives in the directory dir2, which is a subdirectory of dir1. Note that the slash (/) separates directory names in the full path, as well as representing the root directory.

Directory names follow the same rules as file names.

Basic commands

This section introduces the basic Unix command line *utilities*, programs designed to do one specific task. The example images show a grey terminal window with black text and a prompt of the form coates[N], where N is the number of commands which have been entered already.

ls – list files

Perhaps the most basic of all Unix commands, *ls* lists the files in a directory. If no argument is given to ls, as in this example

```
coates[105] ls
bin  data  diff  NEMO_3D  old  OMEN_current  tmp
coates[106]
```

certain files in the current directory are shown. Which files are shown and which are not shown? Read on…

ls -a

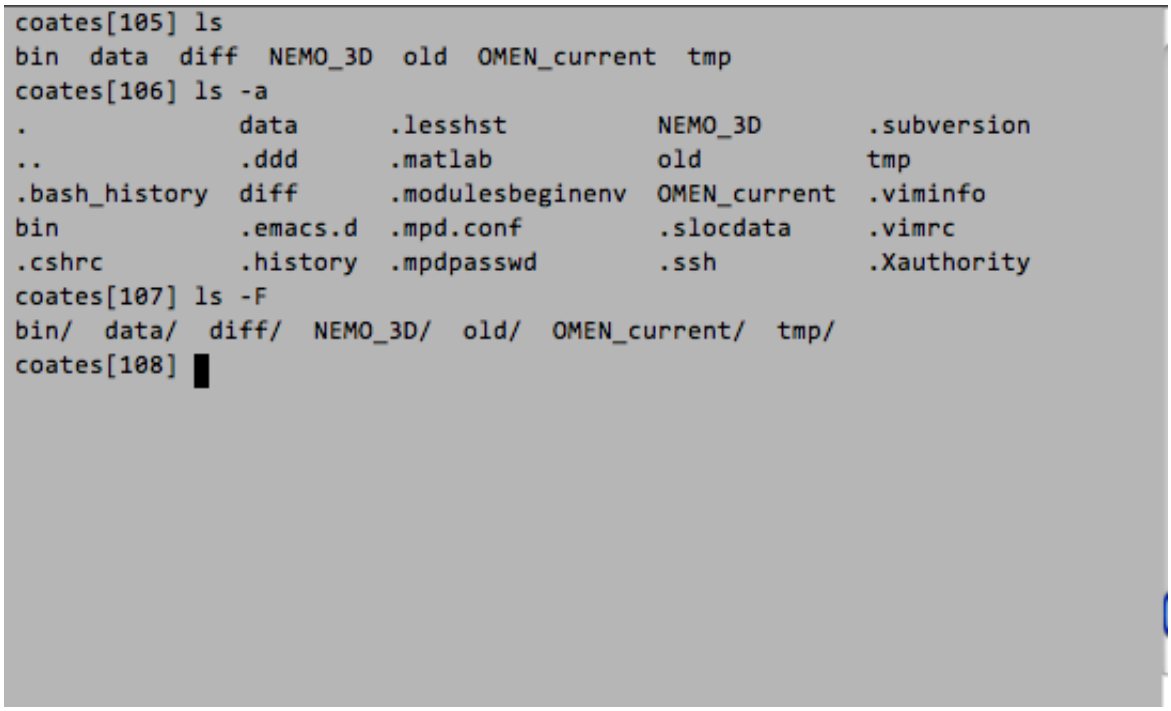
Files whose name begins with dot (.) are *not* shown by default in the output of ls. Such files are called *hidden files* or *dot files*. To see the hidden files in a directory, use the -a argument to ls. This example shows the additional output from ls, listing the hidden files which were not reported by the first invocation of ls.

```
coates[105] ls
bin  data  diff  NEMO_3D  old  OMEN_current  tmp
coates[106] ls -a
.              data       .lesshst           NEMO_3D       .subversion
..             .ddd       .matlab            old           tmp
.bash_history  diff       .modulesbeginenv   OMEN_current  .viminfo
bin            .emacs.d   .mpd.conf          .slocdata     .vimrc
.cshrc         .history   .mpdpasswd         .ssh          .Xauthority
coates[107]
```

Two entries always appear in the output of ls -a, even in empty directories which contain no files: . and .., which are shortcuts to the current directory and the parent directory, respectively.

ls -F

Some of the results listed in the output of ls are subdirectories, while others are simply files. In order to distinguish between files and directories we can use the -F argument (also called a *flag*) to ls.



Arguments can be grouped together. For example, ls -a -F and ls -F -a both list all files, including hidden files, and add a trailing slash (/) to directory names. The order of arguments (e.g. -a -F vs -F -a) does not matter.

```
coates[105] ls
bin  data  diff  NEMO_3D  old  OMEN_current  tmp
coates[106] ls -a
.              data      .lesshst          NEMO_3D       .subversion
..             .ddd      .matlab           old           tmp
.bash_history  diff      .modulesbeginenv  OMEN_current  .viminfo
bin            .emacs.d  .mpd.conf         .slocdata     .vimrc
.cshrc         .history  .mpdpasswd        .ssh          .Xauthority
coates[107] ls -F
bin/  data/  diff/  NEMO_3D/  old/  OMEN_current/  tmp/
coates[108] ls -a -F
./             data/     .lesshst          NEMO_3D/      .subversion/
../            .ddd/     .matlab/          old/          tmp/
.bash_history  diff/     .modulesbeginenv  OMEN_current/  .viminfo
bin/           .emacs.d/ .mpd.conf         .slocdata/    .vimrc
.cshrc         .history  .mpdpasswd        .ssh/         .Xauthority
coates[109]
```

ls *path*

The ls command also accepts a directory argument. This example shows a listing of the files in specific subdirectories of the current directory. Note that the -F flag adds a trailing slash (/) to the names of directories.

```
coates[114] ls -F
bin/  data/  diff/  NEMO_3D/  old/  OMEN_current/  tmp/
coates[115] ls -F OMEN_current
experimental/  NEMO/  stable/
coates[116] ls -F OMEN_current/experimental
cpp/  matlab/
coates[117] ls -F OMEN_current/experimental/cpp
GreenSolver/
coates[118]
```

The shortcut .. can be used to see the files in the parent directory.

```
coates[119] ls -F
bin/  data/  diff/  NEMO_3D/  old/  OMEN_current/  tmp/
coates[120] ls -F ..
aaksyuk/   bdgross/   cob/       frost/     jsheek/    lee399/    sbroui3/
aeislam/   bertodan/  cyarring/  glaze/     junkda/    lee509/    smin/
af5/       bflundgr/  cysung/    gren/      kelynch/   lsgw/      uscms01/
akerner/   bhaley/    ddubbeld/  hbao/      kidder/    mcriswel/  uscms02/
alberts/   billl/     dli/       hdemirel/  kim398/    mjconway/  uscms50/
alkaabi/   bowling/   dnuli/     hpal/      kknopfme/  mr/        vbalchan/
alpachec/  bseib/     ebaum/     hu19/      koziol/    mridilla/  wchayapr/
anant/     cbaumba1/  edavidso/  ijudson/   kumar34/   mtaba/     zetienne/
anavabi/   cgalvin/   ehowell/   jcwirth/   laliu/     rowston/   zhanwenb/
ansley/    cmsprod/   francisc/  jones30/   lamberss/  saul/
coates[121]
```

In the parent directory listing we can see the name of the current directory (bhaley, in this case) because the current directory is a subdirectory of the parent directory.

pwd – print working directory

The *pwd* command shows the full path of the current (working) directory. In the example above we saw the name of the current directory in the listing of the contents of the parent directory. In this example we see the full path of the current directory, bhaley.

```
coates[119] ls -F
bin/  data/  diff/  NEMO_3D/  old/  OMEN_current/  tmp/
coates[120] ls -F ..
aaksyuk/   bdgross/   cob/       frost/     jsheek/    lee399/    sbroui3/
aeislam/   bertodan/  cyarring/  glaze/     junkda/    lee509/    smin/
af5/       bflundgr/  cysung/    gren/      kelynch/   lsgw/      uscms01/
akerner/   bhaley/    ddubbeld/  hbao/      kidder/    mcriswel/  uscms02/
alberts/   billl/     dli/       hdemirel/  kim398/    mjconway/  uscms50/
alkaabi/   bowling/   dnuli/     hpal/      kknopfme/  mr/        vbalchan/
alpachec/  bseib/     ebaum/     hu19/      koziol/    mridilla/  wchayapr/
anant/     cbaumba1/  edavidso/  ijudson/   kumar34/   mtaba/     zetienne/
anavabi/   cgalvin/   ehowell/   jcwirth/   laliu/     rowston/   zhanwenb/
ansley/    cmsprod/   francisc/  jones30/   lamberss/  saul/
coates[121] pwd
/autohome/u101/bhaley
coates[122]
```

The name of the parent directory, then, is u101.

cd – change directory

The *cd* command changes the current (working) directory. In this example we use the parent directory shortcut .. as the argument to cd, which changes the working directory one level up in the directory hierarchy. This can be seen by the output of the pwd commands before and after the cd command.

```
coates[119] ls -F
bin/  data/  diff/  NEMO_3D/  old/  OMEN_current/  tmp/
coates[120] ls -F ..
aaksyuk/   bdgross/   cob/       frost/     jsheek/    lee399/    sbroui3/
aeislam/   bertodan/  cyarring/  glaze/     junkda/    lee509/    smin/
af5/       bflundgr/  cysung/    gren/      kelynch/   lsgw/      uscms01/
akerner/   bhaley/    ddubbeld/  hbao/      kidder/    mcriswel/  uscms02/
alberts/   billl/     dli/       hdemirel/  kim398/    mjconway/  uscms50/
alkaabi/   bowling/   dnuli/     hpal/      kknopfme/  mr/        vbalchan/
alpachec/  bseib/     ebaum/     hu19/      koziol/    mridilla/  wchayapr/
anant/     cbaumba1/  edavidso/  ijudson/   kumar34/   mtaba/     zetienne/
anavabi/   cgalvin/   ehowell/   jcwirth/   laliu/     rowston/   zhanwenb/
ansley/    cmsprod/   francisc/  jones30/   lamberss/  saul/
coates[121] pwd
/autohome/u101/bhaley
coates[122] cd ..
coates[123] pwd
/autohome/u101
coates[124]
```

If no argument is passed to cd, the working directory reverts back to the user's home directory. The home directory is where a Unix shell begins by default.

```
coates[120] ls -F ..
aaksyuk/   bdgross/   cob/       frost/     jsheek/    lee399/    sbroui3/
aeislam/   bertodan/  cyarring/  glaze/     junkda/    lee509/    smin/
af5/       bflundgr/  cysung/    gren/      kelynch/   lsgw/      uscms01/
akerner/   bhaley/    ddubbeld/  hbao/      kidder/    mcriswel/  uscms02/
alberts/   billl/     dli/       hdemirel/  kim398/    mjconway/  uscms50/
alkaabi/   bowling/   dnuli/     hpal/      kknopfme/  mr/        vbalchan/
alpachec/  bseib/     ebaum/     hu19/      koziol/    mridilla/  wchayapr/
anant/     cbaumba1/  edavidso/  ijudson/   kumar34/   mtaba/     zetienne/
anavabi/   cgalvin/   ehowell/   jcwirth/   laliu/     rowston/   zhanwenb/
ansley/    cmsprod/   francisc/  jones30/   lamberss/  saul/
coates[121] pwd
/autohome/u101/bhaley
coates[122] cd ..
coates[123] pwd
/autohome/u101
coates[124] cd
coates[125] pwd
/autohome/u101/bhaley
coates[126]
```

A tilde (~) is used as a shortcut to the home directory; thus cd and cd ~ will both change the working directory to the home directory and the path ~/mydir/myfile indicates a file myfile in a subdirectory mydir of the home directory.

mkdir – create new directory

cp – copy

mv – move (rename)

rm – remove (delete)

rmdir – remove directory

clear – clear screen

cat – display file

man – manual

Slightly more advanced topics

grep – search

| – pipe output

<<, >> – redirect

*, ? – wildcards

ssh – remote access

Links

Other useful Topics

- Software Development
- Using Subversion, a source code version management system

Other websites

- Unix cheat sheet short summaries of key commands
- Unix tutorial