Introduction to the UNIX command line

Steven Abreu

Introduction to Computer Science (ICS) Tutorial
Jacobs University

s.abreu@jacobs-university.de

September 19, 2017

Overview

UNIX

UNIX is a family of operating systems. They are all:

- multitasking
- multiuser

The operating system / kernel manages the computer's resources.

The UNIX Shell

Users communicate with the kernel through the shell.
You enter your commands for the shell in the command line interpreter.
The command line interpreter translates your commands into a language that the kernel understands.

Basic Commands

- `man xyz` - displays a manual page for the command entered
- `login` - log in to your UNIX machine
- `whoami` - displays the current user name
- `cal` - displays a calendar
- `passwd` - change your password
- `clear` - clears your command line

To exit a `man` page, press `q`.

Navigating in your file system

- `pwd` - displays the current working directory
- `cd [x]` - changes the current working directory
- `ls` - list files in your directory
- `ls -l` - list files with more information
- `ls -a` - list all files (also hidden ones)

Press `tab` to complete your command.
`../` takes you into your parent directory.
`./` takes you to your current directory.
`/` specifies a sub-directory.
`~` is your home directory.

Interacting with files

- `touch` file_name - creates or 'touches' a file (new timestamp)
- `mkdir` directory_name - create a new directory
- `rm [-i]` file_name - remove file
- `cp` f1 f2 - copies f1 to path of f2
- `mv` f1 f2 - moves f1 to path of f2 (also renaming). Works for directories too.
- `chmod` - change permissions (later!)
- `grep` *pattern file* - returns all lines containing *pattern* in *file*
- `cat` file_name - display content of a file (as text)
- `vi` file_name - edit a file

Vi's modes

Vi has two modes:

1. command mode: cursor movement, text deletion, pasting. Any insert or change command puts editor into insertion mode.
2. insertion mode: inserting and changing. [ESC] returns editor to command mode.

Editing files using `vi`

1. open the file using the `vi` command
2. enter insert mode by pressing `i`
3. use cheatsheet to look up all the commands
4. enter command mode by pressing `[ESC]`
5. exit and save your changes with `:x`

Permissions in UNIX

Every file has permissions in UNIX. They are split up into three categories:

1. Owner permissions
2. Group permissions
3. Other (world) permissions

We can check files' permissions with `ls -l`

Changing file's permissions

We can change a file's permissions using `chmod`.
Categories:

1. u: changes owner's permissions
2. g: changes group's permissions
3. o: changes world's permissions

Options:

1. +: add permissions
2. -: remove permissions
3. =: set permissions

Permissions:

1. r: read
2. w: write
3. x: execute

Example - Changing file's permissions

- `chmod o-wx` - remove write and execute permissions from the world.
- `chmod o=rwx` - set world's permissions to read, write and execute.
- `chmod g+x` - add execute permission for the owner group.

The concept of environment

An environment is defined by the environment variables.
There are certain system variables saved in your UNIX environment.
You can output variables using the `echo` command.

Example:
Type `echo $TERM` into your terminal window. This returns the type of terminal you are using.

Setting variables

- You can set variables like this:
  `TEST = "ICS - UNIX Tutorial"`
- You can then access your variables with `$TEST`
- You can print your variables like this:
  `echo $TEST`

Wildcards

The UNIX command line knows two wildcards:

1. * represents any sequence of characters - also empty sequences
2. ? represents any one character

Introduction to Git

- Git is a version control system.
- Widely used in software development.
- A git repository is a directory.
- Git stores all versions of your data as snapshots.

The three states in Git

There are three main sections in a Git project:

1. Working Directory: when you modify your your data, you modify your working directory.
2. Staging Area: when you stage your data, you add snapshots of it to your staging area.
3. Repository: when you do a commit, Git takes all files from the staging area and stores them as one snapshot into your git repository.

Git from the command line

To use Git in your terminal, you first need to install Git.
Git commands:

- `git init` - initialize your current directory as a new git repository
- `git add .` - add all modified files to the staging area
- `git commit -m 'your commit message'` - commit your changes to the repository
- `git remote add ...` - add a remote repository
- `git push ...` - push your changes (all your commits) to the remote
- `git branch ...` - create a new branch in your repository
- `git checkout ...` - check out one specific commit or branch in your repository

Further Information

- UNIX and Shell Commands: here
- Git: here and the official website