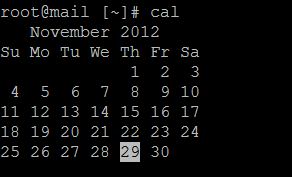
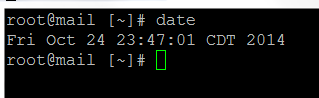
**Class 6**

The **cal** command displays a calendar

Use **cal 2013** to view the calendar for 2013 for example (**man cal** shows you the options)



**date** – this shows you the current date on the system:



If the date and time are not accurate, we need to do the following steps:

* **yum install ntp –y** (this installs the ntp aka Network Time Protocol service which is meant to sync to an NTP server.
* **chkconfig ntpd on** (with this the ntpd service will start when you boot-up your server). Note chkconfig was the default before Centos 7 came out. Although you can still use it, the new way is: **systemctl**
* **systemctl enable ntpd** - this enables ntpd
* **systemctl is-enabled ntpd** – this shows if ntpd is enabled
* **cp /usr/share/zoneinfo/America/Mexico\_City /etc/localtime** (with this command we copy the timezone file for Mexico City which is Central time and we copy it into **/etc/localtime** - note: type “y” for when it asks you if you want to overwrite the file /etc/localtime. Also, we could have selected any timezone we founded just look at the **/usr/share/zoneinfo** for the list and folders)
* If the date / time is still incorrect, we can set our server to point to a time server of our own choosing. We can use VI for that:
* **vi /etc/ntp.conf** (you will want to add one such as this to the pool: server 0.pool.ntp.org)
* **service ntpd restart** (we restart the service)
* date should now be accurate as well as the time zone

Change the **date, timezone, & time**: <http://www.cyberciti.biz/faq/unix-set-date-command/>

The **?** metacharacter is similar to the **\*** - it can be used to fill in for exactly one, two, or x characters. So for example, if you use ls file?? It will ls any files named fileXX.



**touch** – With touch, you can change the access times, or both for a file. You can also use it create an empty file.

<http://en.wikipedia.org/wiki/Touch_(Unix)>

The simplest [use case](http://en.wikipedia.org/wiki/Use_case) for touch is thus:

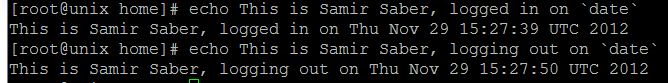
$ **touch** myfile.txt

Touch doesn't modify the contents of myfile.txt; it just updates the [timestamp](http://en.wikipedia.org/wiki/Timestamp) of the file to the computer's current date and time, whatever that happens to be. Or, if myfile.txt does not exist it is created, with zero length.

The **echo** is an important command to know as it is used quite often. In simple terms, it prints/returns the message that you echo. For example:



When something gets echoed, it gets written on the history file as well. This is one way that system administrators track whom logged in and at what time. I could do:



The echo command is commonly used for tech support purposes to keep track of when a given tech logged in and signed out.

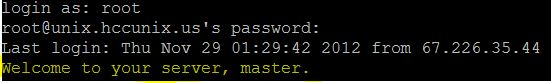
**echo** could also be used to show that a given script for example has completed. When you run a yum installation you will see the message “Complete!” at the end – this is an echo command inside of that script.

On this last example, you will notice the `date` command.

The **`** symbol is called a **grave accent mark** and it can be found above the “Tab” key (the ~ character is Tilde).

When utilizing two `` with a command inside of it, the command will be executed by itself and be separate from the rest.

The **motd** (message of the day) can be changed as a greeting for logging into a server. This is typically a place for administrators to put a disclaimer warning users that login that they could be prosecuted etc. The text file to change is in: **/etc/motd**

****

Also remember that UNIX was designed to be a multi-user environment where users could share the same applications and communicate. One of the ways users can keep track of daily occurrences could be through the message of the day (motd). Think big navy ships that use Linux and admins need to put a note such as “Doing backups at 9am – Don’t touch Apache” – or something like that. It’s a way of communicating.

**top** brings up the list of processes running – this is the same concept as Task Manager in windows

PID is the process ID

USER is the user tied to it

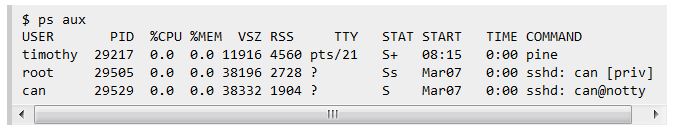
%CPU is the amount of CPU usage that process is taking

%MEM is the amount of ram used relative to the total system’s ram

TIME+ shows how long it’s been running

**ps** – shows current processes

**ps aux** (shows processes for all users)



USER = user owning the process  
PID = process ID of the process  
%CPU = It is the CPU time used divided by the time the process has been running.  
%MEM = ratio of the process’s resident set size to the physical memory on the machine  
VSZ = virtual memory usage of entire process  
RSS = resident set size, the non-swapped physical memory that a task has used  
TTY = controlling tty (terminal)  
STAT = multi-character process state  
START = starting time or date of the process  
TIME = cumulative CPU time  
COMMAND = command with all its arguments

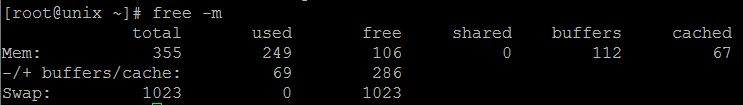
To kill a process: **kill -9 PID #** (replace PID # with the number you would get by using the ‘top’ command).

You could also do **killall -9 httpd** - this will kill all apache processes. Note: killall worked for version Centos 6 and previous ones. For Centos 7 you can use pkill

**pkill ntpd** – this will kill all processes related to the ntpd service.

**uptime** – shows how long the system has been powered up

**free –m** shows the total amount of ram being used (in megabytes)

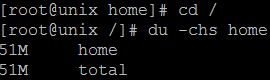


You want to look at the -/+ buffers/cache – that will show you how much you really have free

By default UNIX will utilize as much RAM as it can to process – it’s efficient like that

**df –h** (shows how much space is left on the system and the h makes it readable)

In order to estimate file space usage of a file or directory, the **du** command can be utilized. I like to use **du –chs** (there are other flags/options you can use, just man du and you will see what they do).

****

**service httpd start/stop/status – (httpd is for apache, it can be replaced with other services’ names) –**

**/etc/init.d/httpd start/stop/status/restart**

Start = starts the service

Stop = stops the service

Status = provides the status on the service

Restart = stops/starts the service

For Centos 7 we want to use **systemtcl start httpd** (or the full /bin/systemctl start httpd.service) – this is to start the Apache service.

**/proc/cpuinfo – text file that you can use cat to read (or less etc..) – it will show cpu info**

**/proc/meminfo – shows extensive memory-related information**

**/proc/version – shows details on the OS**

**/proc/partitions – shows how the drive has been partitioned**