Command Line Potpourri

CS14 - S26

Environment Variables

- Environment Variables are special variables maintained by the Bash shell that influence both the shell and programs run from it.
- Typically written in ALL CAPS
- Can be added/modified in the shell, by modifying `.bashrc`, etc..
- Current environment can be seen with `printenv`
- Many common variables can be seen in `man bash` under “Shell Variables”

Environment Variables – PATH

`PATH` is a colon separated list of directories that the shell uses for commands.

If an executable's directory is in `PATH`, you do not need to specify the full path to that executable to run it.

Exercise:
Open `~./bashrc` and add `/usr/games` to your PATH variable:

```
export PATH=$PATH:/usr/games
```

diff

- `diff` compares the contents of two files line by line and prints the differences
- `diff <file1> <file2>`
  - will show lines that are only in `<file1>` prepended with "`<`" and contents that are only in `<file2>` prepended with "`>`"

`diff -y` will output the files side-by-side

`diff -r` will compare the contents of directories recursively

`diff -u` will output "unified context" around differences, which is useful for patching

patch

`patch` can use output from diff to update files.

This is often used when developers want to distribute small updated to source code, e.g. via a message group or slack channel.

Historically, this is how updates to the Linux Kernel were distributed before source control.

- `patch -b <originalfile> -i <patchfile>`
- `patch -p<num> < <patchfile>`

diff and patch

Exercise:

Copy `/home/ckazer/public/word_lists/veggies.txt` .

Then, make a copy of `veggies.txt` called `modified_veggies.txt`, then add and remove some vegetables from `modified_veggies.txt` .

Using diff, try making a patch file that could upgrade `veggies.txt` to be the same as `modified_veggies.txt` .

Try applying the patch (using the `-b` option to make a backup), and ensure that update works.

tar and gzip

`tar` packages multiple files/directories into an “archive” file without compressing them.

`gzip` compresses files.

They are often used together to create compressed “tarball” `.tar.gz` or `.tgz` files.

These can be used for

- Simple backups
- To distribute packages
- Archive files that you don’t need immediate access to
- etc.

tar and gzip

tar

- `tar -cvf <archive_name> <directory>` creates an archive
- `tar -xvf <archive_name>` extracts an archive

gzip

- `gzip <file>` creates a compressed `.gz` file
- `gzip -d <file>` decompresses a `.gz` file

tar and gzip

Zipping and unzipping with `gzip` can be done as part of the `tar` command using the `-z` option.

- `tar -cvzf <archive_name> <directory>`
- `tar -xvzf <archive_name>`

Exercise:
On your `cs14vm`, turn your old navigation exercises into tarballs. Once you are certain that the tarballs contain all of the navigation exercise information, it is safe to remove the original navigation exercise folders!

at

`at` is used to schedule commands/scripts to be run at a later time. Standard output and standard error are sent to your email.

- `at <time> -f <script>`

`<time>` can be specified many different ways, e.g.,

- `now + 1 hour`
- `00:30 Thu`
- `09:15 PM 04/23/26`

cron

`cron` is a daemon that runs recurring commands/scripts.

`crontab` is used to add `cron` jobs.

`crontab -e` will open your crontab file for editing.