

Unlocking the Power of UNIX: tr Command

The `tr` command in UNIX is a handy tool to translate or delete characters. Whether you need to convert lowercase to uppercase or perform basic find-and-replace operations, `tr` can help you get the job done.

```
fa.wikipedia.org
g (208.80.152.2) 56(84) bytes of data.

ping statistics ---
sived, 0% packet loss, time 0ms
28/540.528/540.528/0.000 ms

5 Jul 30 22:43 .
5 Sep 14 20:42 ..
5 May 14 00:15 account
5 Jul 31 22:26 cache
5 May 18 16:03 db
5 May 18 16:03 empty
5 May 18 16:03 games
5 Jun 2 18:39 gdm
5 May 18 16:03 lib
5 May 18 16:03 local
1 May 14 00:12 lock -> ../run/lock
5 Sep 14 20:42 log
3 Jul 30 22:43 mail -> spool/mail
5 May 18 16:03 nis
5 May 18 16:03 opt
5 May 18 16:03 preserve
5 Jul 1 22:11 report
5 May 14 00:12 run -> ../run
5 May 18 16:03 spool
5 Sep 12 23:50 tmp
5 May 18 16:03 yp
arch wiki
resto, refresh-packagekit, remove-with-leaves
ry_db
```

Mastering the uniq Command in Linux

Remove duplicates

The ****uniq**** command is perfect for removing repeated lines from files. Add the '**-u**' option to display only unique lines.

Print duplicate lines

Use the '**-D**' option in combination with the '**-c**' option to print all duplicate lines.

Compare case insensitively

Make comparisons case insensitive by using the '**-i**' option.



```
atmosphere ~/apcb/intro/blast$ cat yeast_blastp_yeast_top2.txt | \
-v '#' | \
{ \
0 < 1e-30) {print "great",$1,$2,$10} \
if($10 < 1e-20) {print "good",$1,$2,$10} \
{print "ok",$1,$2,$10}\
```

Awk: Your Go-To Text Processing Tool

Awk is a powerful text processing tool. Whether you need to format output lines, perform arithmetic and string operations, or work with conditionals and loops, Awk is there to help. Here are just a few things you can do with Awk:

- Transform data files
- Produce formatted reports
- Search for patterns in file lines

Awk Operations: Counting Records

1

Command: NR

Use NR to keep a count of the number of input records.

2

Example:

Count the number of lines in a file called 'file.txt':

```
awk 'END {print NR}' file.txt
```



Made with Gamma

Awk Operations: Splitting a Line into Fields

Example:

Show the first and fourth fields of a file named 'file.txt':

```
awk '{print $1, $4}' file.txt
```

1

2

Command: \$n

For each record, awk splits the line delimited by whitespace by default, and stores it in \$n variables. If the line has 4 words, they will be stored in \$1, \$2, \$3, and \$4 respectively.

Awk Operations: Built-In Variables

Command: NR

NR stores a current count of the number of input records.

Command: NF

NF stores a count of the number of fields within the current input record.

Command: FS

FS contains the field separator character used to divide fields on the input line.

Command: \$0

\$0 represents the whole line.

Awk Options: Reading the AWK Program Source

```
# Sample configuration file for ISC dhcpcd
#
# option definitions common to all supported networks...
option domain-name "benet.com";
option domain-name-servers 192.168.235.132;

default-lease-time 600;
max-lease-time 7200;

# Use this to enable / disable dynamic dns updates globally.
#ddns-update-style none;

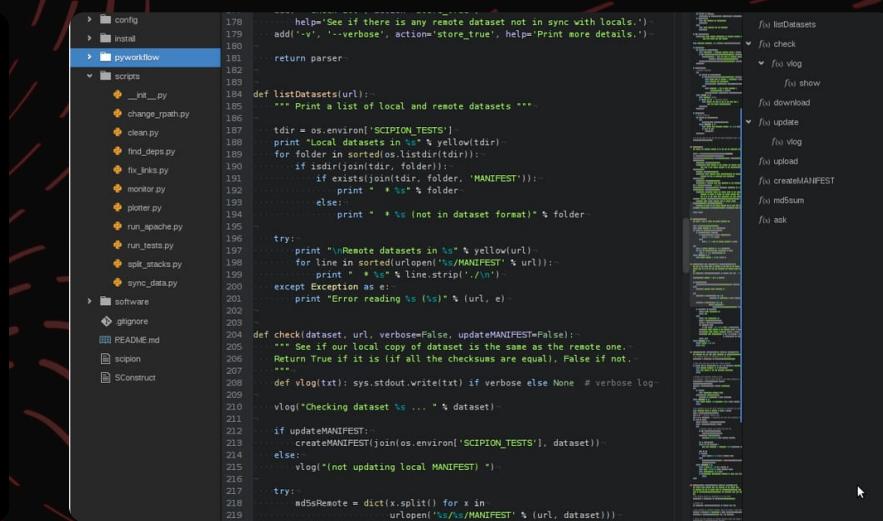
# If this DHCP server is the official DHCP server for the local
# network, the authoritative directive should be uncommented.
#authoritative;

# Use this to send dhcp log messages to a different log file (you also
# have to hack syslog.conf to complete the redirection).
log-facility local7;

# No service will be given on this subnet, but declaring it helps the
# DHCP server to understand the network topology.
```

Option: -f program-file

Use the -f option to read the AWK program source from a file, instead of from the first command line argument.



```
178     help='See if there is any remote dataset not in sync with locals.')
179     add('-v', '--verbose', action='store_true', help='Print more details.')
180
181     return parser
182
183
184 def listDatasets(url):
185     """ Print a list of local and remote datasets """
186
187     tdir = os.environ['SCIPION_TESTS']
188     print("Local datasets in %s" % yellow(tdir))
189     for folder in sorted(os.listdir(tdir)):
190         if os.path.isdir(folder):
191             if existsInManifest(folder, 'MANIFEST'):
192                 print(" * %s" % folder)
193             else:
194                 print(" * %s (not in dataset format)" % folder)
195
196     try:
197         print("\nRemote datasets in %s" % yellow(url))
198         for line in sorted(urllib.urlopen("%s/MANIFEST" % url)):
199             print(" * %s" % line.strip())
200     except Exception as e:
201         print("Error reading %s (%s)" % (url, e))
202
203
204 def checkDataset(url, verbose=False, updateMANIFEST=False):
205     """ See if our local copy of dataset is the same as the remote one.
206     Return True if it is (if all the checksums are equal), False if not.
207     """
208
209     def vlog(txt): sys.stdout.write(txt) if verbose else None # verbose log
210
211     if updateMANIFEST:
212         createMANIFEST(join(os.environ['SCIPION_TESTS'], dataset))
213     else:
214         vlog("(not updating local MANIFEST)")
215
216     try:
217         md5sRemote = dict(x.split() for x in
218                           urllib.urlopen("%s/%s/MANIFEST" % (url, dataset)))
219
220         for dataset in os.listdir(dataset):
221             if dataset != 'MANIFEST':
222                 md5sLocal = dict(x.split() for x in
223                                 open(join(dataset, 'MANIFEST')).readlines())
224
225                 if md5sLocal == md5sRemote:
226                     vlog("Dataset %s is up-to-date" % dataset)
227                 else:
228                     vlog("Dataset %s is NOT up-to-date" % dataset)
229
230             else:
231                 vlog("Dataset %s is up-to-date" % dataset)
232
233         vlog("All datasets are up-to-date")
234
235     except Exception as e:
236         vlog("Error reading %s (%s)" % (url, e))
237
238
239     if updateMANIFEST:
240         createMANIFEST(join(os.environ['SCIPION_TESTS'], dataset))
241     else:
242         vlog("(not updating local MANIFEST)")
```

Example:

Execute an AWK program stored in the file program.awk:

```
awk -f program.awk input-file.txt
```

Awk Options: Setting the Input Field Separator

1 Option: -F fs

Use '-F fs' to set fs as the input field separator. The default is whitespace.

2 Example:

Count the number of fields in a line, assuming the field separator is a comma:

```
awk -F ',' '{print NF}' some_csv_file.txt
```