touch – change file timestamps

touch command is one of these little but extremely useful tools in Unix and Linux which you may have used for quite sometime before realizing their full potential. In short, it updates file timestamps – access and modification ones (**atime** and **mtime** respectively).

Why modify file timestamps?

There are quite a few legitimate reasons why you may want to update timestamps on a certain file. Ranging from source control approaches to storage usage analysis, there are processes out there which rely on the timestamps associated with each file and directory of yours.

After all, it's always useful to know when the file was last modified or when somebody tried to access its contents.

Changing timestamps of a time to the current system time

The default behavior of **touch** command is to change all three timestamps associated with a file to the current system time.

You simply specify the filename as a command line parameter, no oother options are needed. If there isn't a file with the specified name, touch command will create it for you if permissions allow it:

```
ubuntu$ ls try
ls: try: No such file or directory
ubuntu$ touch try
ubuntu$ ls try
try
ubuntu$ stat try
```

```
  File: `try'
  Size: 0               Blocks: 0          IO Block: 4096
regular empty file
Device: 801h/2049d      Inode: 655596      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   greys)   Gid: (
113/   admin)
Access: 2008-11-17 08:03:02.000000000 -0600
Modify: 2008-11-17 08:03:02.000000000 -0600
Change: 2008-11-17 08:03:02.000000000 -0600
ubuntu$ date
Mon Nov 17 08:03:05 CST 2008
```

As you can see from the example, the file which isn't originally found, gets created by the touch command and gets its timestamps set to the current system time and date.

Changing file timestamps to a specific date and time

If you have a specific time and date you would like to be used for all the timestamps of a file or directory, **touch command** will gladly accempt a timestamp template with **-t** command line option.

Template for the timestamp is quite thorough: **[[CC]YY]MMDDhhmm[.ss]**, but it's entirely up to you whether to specify the year (either two-digit or a full form) or not.

This example resets the date to October 16th:

```
ubuntu$ touch -t 10161000 ./try
ubuntu$ stat ./try
  File: `./try'
  Size: 0               Blocks: 0          IO Block: 4096
regular empty file
Device: 801h/2049d      Inode: 655596      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   greys)   Gid: (
113/   admin)
Access: 2008-10-16 10:00:00.000000000 -0500
Modify: 2008-10-16 10:00:00.000000000 -0500
Change: 2008-11-18 03:54:10.000000000 -0600
```

As you can see from the output, both access time and modification time got updated. The reason change time (ctime) is set to a different date is because this field reflects the last update to the inode behind a file, and always reflects the current time. In other words, it’s set to Nov 18th 2008 because of the date of writing this example.

If you fancy adding a year to the timestamp specification, you can specify something from both past and future.

Here’s how easy it is to set atime and mtime to the Oct 16th, 2010 date:

```
ubuntu$ touch -t 201010161000 ./try
ubuntu$ stat ./try
  File: `./try'
  Size: 0               Blocks: 0          IO Block: 4096
regular empty file
Device: 801h/2049d      Inode: 655596      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   greys)   Gid: (
113/   admin)
Access: 2010-10-16 10:00:00.000000000 -0500
Modify: 2010-10-16 10:00:00.000000000 -0500
Change: 2008-11-18 03:57:30.000000000 -0600
```

Modifying atime of a file in Unix

Similar to the commands above, you can use **-a** option to make **touch** only update the access time field of a file:

```
ubuntu$ touch -at 200010161000 ./try
ubuntu$ stat ./try
  File: `./try'
  Size: 0               Blocks: 0          IO Block: 4096
regular empty file
Device: 801h/2049d      Inode: 655596      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   greys)   Gid: (
113/   admin)
Access: 2000-10-16 10:00:00.000000000 -0500
Modify: 2010-10-16 10:00:00.000000000 -0500
Change: 2008-11-18 04:05:22.000000000 -0600
```

Modifying mtime of a file in Unix

If you’re interested in updating the modification date only, use **-m** option:

```
ubuntu$ touch -mt 200510161000 ./try
ubuntu$ stat ./try
  File: `./try'
  Size: 0               Blocks: 0          IO Block: 4096
regular empty file
Device: 801h/2049d      Inode: 655596      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   greys)   Gid: (
113/   admin)
Access: 2000-10-16 10:00:00.000000000 -0500
Modify: 2005-10-16 10:00:00.000000000 -0500
Change: 2008-11-18 04:07:12.000000000 -0600
```

Using a reference file to set atime and mtime

Finally, the really useful option for synchronizing access and modification time fields between multiple files is to use reference file. A reference file is the file which already has the timestamps you’d like to copy:

```
ubuntu$ stat /etc/lsb-release
  File: `/etc/lsb-release'
  Size: 97              Blocks: 8          IO Block: 4096
regular file
Device: 801h/2049d      Inode: 1278451     Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (
0/    root)
Access: 2008-11-14 05:30:09.000000000 -0600
Modify: 2007-04-12 01:02:52.000000000 -0500
Change: 2007-09-26 02:41:20.000000000 -0500
```

By specifying this file using a -r option, you can use the touch command to set the same atime and mtime values to any file of yours:

```
ubuntu$ touch -r /etc/lsb-release ./try
```

```
ubuntu$ stat ./try
  File: `./try'
  Size: 0               Blocks: 0          IO Block: 4096
regular empty file
Device: 801h/2049d      Inode: 655596      Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1000/   greys)   Gid: (
113/   admin)
Access: 2008-11-14 05:30:09.000000000 -0600
Modify: 2007-04-12 01:02:52.000000000 -0500
Change: 2008-11-18 04:09:02.000000000 -0600
```

See also:

- atime, ctime and mtime in Unix filesystems
- Advanced Unix commands
- Commands in Unix
- find command in Unix