Soft Links and Hard Links

Soft Links	**Hard Links**
Soft Links are invalid once the source or parent file is deleted	Removing source file won't affect Hard links because they share same inode number, permissions and data of the original file
Soft link is just a link that points to the original file yet with a different inode number and file permissions	Hard link and source file have same inode number and file permissions
Soft link is a shortcut to a cut, if you remove the source file, the shortcut becomes useless	If we change permissions on a source file, the same permissions will be applied to hard link.
If a soft link is removed, the original file is still available	If a hard link is removed, the original file is still available

Soft Links

Let's create two new files:

```
user$ touch file1.txt file2.txt
```

Now let's see the current directory

```
user$ ls –li
```

```
8607507145 -rw-r--r--   1 user  staff     23 Jul 29 15:40 cnt.txt
8607506955 -rw-r--r--   1 user  staff  29313 Jul 29 15:31 combined.txt
8607507120 -rw-r--r--   1 user  staff     23 Jul 29 15:38 count.txt
8607506444 drwxr-xr-x   4 user  staff    128 Jul 29 15:13 dir1
8607506764 drwxr-xr-x   4 user  staff    128 Jul 29 15:25 etcbackup
8607509213 -rw-r--r--@  1 user  staff     15 Jul 29 17:08 evenmore.txt
8607575793 -rw-r--r--   1 user  staff      0 Jul 31 16:20 file1.txt <—
8607575794 -rw-r--r--   1 user  staff      0 Jul 31 16:20 file2.txt <—
8607510585 drwxr-xr-x   3 user  staff     96 Jul 29 17:51 hardlink
8607509284 -rw-r--r--@  1 user  staff      0 Jul 29 17:07 hello
8607548672 -rw-r--r--   1 user  staff      0 Jul 30 21:30 hello world
8607506877 -rw-r--r--@  1 user  staff  26624 Jul 29 15:29 hello.txt
8607506425 -rw-r--r--   1 user  staff      0 Jul 29 15:10 hellocopy.txt
```

Now let’s soft link them using following cmd

```
user$ ln –s  file1.txt file2.txt
```

```
8607507145 -rw-r--r--   1 user  staff     23 Jul 29 15:40 cnt.txt
8607506955 -rw-r--r--   1 user  staff  29313 Jul 29 15:31 combined.txt
8607507120 -rw-r--r--   1 user  staff     23 Jul 29 15:38 count.txt
8607506444 drwxr-xr-x   4 user  staff    128 Jul 29 15:13 dir1
8607506764 drwxr-xr-x   4 user  staff    128 Jul 29 15:25 etcbackup
8607509213 -rw-r--r--@  1 user  staff     15 Jul 29 17:08 evenmore.txt
8607576596 -rw-r--r--   1 user  staff      0 Jul 31 16:38 file1.txt
8607576802 lrwxr-xr-x   1 user  staff      9 Jul 31 16:44 file2.txt -> file1.txt
8607510585 drwxr-xr-x   3 user  staff     96 Jul 29 17:51 hardlink
8607509284 -rw-r--r--@  1 user  staff      0 Jul 29 17:07 hello
8607548672 -rw-r--r--   1 user  staff      0 Jul 30 21:30 hello world
8607506877 -rw-r--r--@  1 user  staff  26624 Jul 29 15:29 hello.txt
8607506425 -rw-r--r--   1 user  staff      0 Jul 29 15:10 hellocopy.txt
```

Here, we see both file1 and file2 have different inode numbers and file permissions and file2 points to file1

Now, if we delete the source file i.e file1.txt

```
user$ rm file1.txt
```

The link between file2.txt and file1.txt still stays, but if we try to view th contents of file2.txt using cat we got the following error:

cat file2.txt
cat: file2.txt: No such file or directory

Hence, file2.txt becomes useless.

Hard Links
Let's create two new files:

user$ touch file1.txt file2.txt

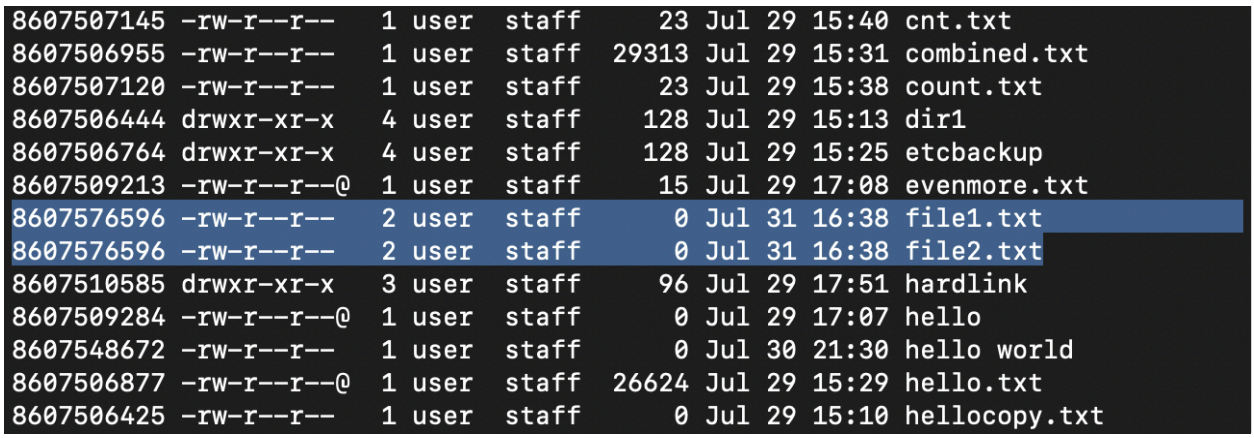
Now let's see the current directory

user$ ls –li

```
8607507145 -rw-r--r--   1 user  staff     23 Jul 29 15:40 cnt.txt
8607506955 -rw-r--r--   1 user  staff  29313 Jul 29 15:31 combined.txt
8607507120 -rw-r--r--   1 user  staff     23 Jul 29 15:38 count.txt
8607506444 drwxr-xr-x   4 user  staff    128 Jul 29 15:13 dir1
8607506764 drwxr-xr-x   4 user  staff    128 Jul 29 15:25 etcbackup
8607509213 -rw-r--r--@  1 user  staff     15 Jul 29 17:08 evenmore.txt
8607575793 -rw-r--r--   1 user  staff      0 Jul 31 16:20 file1.txt <—
8607575794 -rw-r--r--   1 user  staff      0 Jul 31 16:20 file2.txt <—
8607510585 drwxr-xr-x   3 user  staff     96 Jul 29 17:51 hardlink
8607509284 -rw-r--r--@  1 user  staff      0 Jul 29 17:07 hello
8607548672 -rw-r--r--   1 user  staff      0 Jul 30 21:30 hello world
8607506877 -rw-r--r--@  1 user  staff  26624 Jul 29 15:29 hello.txt
8607506425 -rw-r--r--   1 user  staff      0 Jul 29 15:10 hellocopy.txt
```

Now let's hard link them using following cmd

user$ ln file1.txt file2.txt



```
8607507145 -rw-r--r--   1 user  staff     23 Jul 29 15:40 cnt.txt
8607506955 -rw-r--r--   1 user  staff  29313 Jul 29 15:31 combined.txt
8607507120 -rw-r--r--   1 user  staff     23 Jul 29 15:38 count.txt
8607506444 drwxr-xr-x   4 user  staff    128 Jul 29 15:13 dir1
8607506764 drwxr-xr-x   4 user  staff    128 Jul 29 15:25 etcbackup
8607509213 -rw-r--r--@  1 user  staff     15 Jul 29 17:08 evenmore.txt
8607576596 -rw-r--r--   2 user  staff      0 Jul 31 16:38 file1.txt
8607576596 -rw-r--r--   2 user  staff      0 Jul 31 16:38 file2.txt
8607510585 drwxr-xr-x   3 user  staff     96 Jul 29 17:51 hardlink
8607509284 -rw-r--r--@  1 user  staff      0 Jul 29 17:07 hello
8607548672 -rw-r--r--   1 user  staff      0 Jul 30 21:30 hello world
8607506877 -rw-r--r--@  1 user  staff  26624 Jul 29 15:29 hello.txt
8607506425 -rw-r--r--   1 user  staff      0 Jul 29 15:10 hellocopy.txt
```

Here, both file1.txt and file2.txt have same inode number, same permissions and basically have same properties.

But the main difference is even we delete the source file here i.e. file1.txt file2.txt can still access to the contents that were inside of file1.txt and we won't get any error upon accessing the contents of fil2.txt as we did in soft links.
We can simply access contents using:

```
cat file2.txt
```