Rsync - You'd be a fool not to!

Rsync is a rapid file copy function on steroids available in all flavors of Linux.

'On steroids' means that you can easily preserve all permissions and links, compress / decompress during transfer to speed up transmission between computers, use SSH to perform the copy, log outputs and many, many more options.

This the a basic / beginner cheat sheet so we'll leave some of the arcana out, below, just to keep it simple.

In general, rsync will only copy/transfer what it needs to. It reads the 'from', it reads the 'to', if any files in 'to' are missing or older than the files in 'from', it updates what's missing or old.

At its most basic, it's simply: rsync -options fromfolder tofolder

There are lots of options and combos, but I usually use it as

rsync -avPzh *from to*

a = archive mode...keep permissions, links, etc.
v = verbose...tell me what's going on (or, as you'll see later, log it)
P = keep partial files (i.e. if a break in transmission...why not?)
z = compress/decompress (for sending over a network)
h = show the screen or log in human readable form.

Truly, the only option that's critical is a, but verbose comforts me, etc.

If I'm going to a windows FAT type folder, i'll also include --modify-window=3601

so **rsync -avPzh --modify-window=3601 *from to***

This has something to do with how FAT keep timestamps of file dates versus linux and, I confess, I'm not entirely sure how but, in practice, I've verified this works well.

Lastly, with '-e ssh', i can send the transfer to another computer on my network.

so **rsync -avPzh -e ssh --modify-window=3601 *from to-another-computer***

Now, before getting any more sophisticated or fancy, let's try a few examples just to see how it works.

my home folder is lou....so in linux /home/lou...try this and replace /home/lou with your own home folder. (the touch command in linux simply creates a blank file)

we'll make a folder (tbs...to be saved) and another folder (svd...where we're saving) and run some tests....

```
mkdir tbs
mkdir svd
touch tbs/a.txt
touch tbs/b.txt
touch tbs/c.txt

rsync -avPzh /home/lou/tbs/ /home/lou/svd/
```

you should see that the three files we created, a.txt, b.txt and c.txt transferred to the /home/lou/svd folder. Check for yourself that the permissions are the same, etc.

Now we'll add another file to be saved....

```
touch tbs/d.txt
rsync -avPzh /home/lou/tbs /home/lou/svd
```

you should see d.txt and only d.txt got transferred.

Now we'll remove a file from the saved folder

```
rm svd/b.txt
rsync -avPzh /home/lou/tbs/ /home/lou/svd/
```

only b.txt was missing from svd and so only b.txt was transferred.

Now we'll change a 'to be saved' file

```
nano tbs/a.txt
```

then type something...ctrl-O to write out, ctrl-X to end.

```
rsync -avPzh /home/lou/tbs /home/lou/svd
```

Only a.txt should be transferred, because it changed.

See how easy that is?

In practice, at home, i backup from one computer to another...

```
rsync -avPzh -e ssh --modify-window=3601 /share/ music@192.168.17.5:/share/
```

Here's the bash script I use:

```
#!/bin/bash

#create a log file for today
LOGFILE=/home/music/logs/backup-$(date +%Y%m%d).log
echo $LOGFILE

#rsync any changed folders to the backup raspberry pi...log activiy
rsync -avPzh -e ssh --log-file=$LOGFILE --modify-window=3601 /share/
music@192.168.17.5:/share/
```

I'll leave it to you to look up how to use ssh, or make shell scripts executable or slam this script into cron...plenty of sources for that on the internet.

Hopefully, this shows you just how easy it is to get started with rsync. Once you've used it for a bit, try 'man rsync' and check out just how flexible and powerful it can be....this truly IS, just the beginner overview.