

Auto Sync Local Changes to Remote Using rsync

Whenever I'm working on a new site, it's so incredibly tedious to change a few files, manually upload your changes via SFTP/SSH (if you're still using FTP, shame on you), reload your browser, check those changes, and repeat another 150+ times until you finally get it right.

For a while, I had a fairly good system where I would use [WinSCP](#) to auto-sync my changes to my production box using their "Keep Synchronized" feature. The problem is that WinSCP is only for Windows, not OS X. As a workaround, I installed WinSCP on a Windows VM, made my Mac filesystem visible to the VM, and would have WinSCP reflect those edits while I made code changes in Sublime on the Mac side.

Obviously this isn't a great solution, but it did the job. I soon got tired of having to spin up the VM everytime I wanted to work on a project, and I found myself eating through battery more quickly running two operating systems at once (making working remotely difficult, because I never have a charger).

I searched around for a native OS X solution that would do a better job. There isn't. At least, not something that does it well. And not for free. I'm a cheapskate.

Through some digging, I discovered a command line solution. It's a combination of rsync and fswatch. More or less, this is how I've implemented it:

[fswatch](#) is a great little tool that uses the Mac OS X FSEvents API to monitor a directory for changes. When a change event about any file or directory is received, a user-specified bash script is executed. I use fswatch to watch changes on my local project directory, and then invoke a bash script whenever a change event is received. A "change" includes file/directory creations, edits, and deletes. The bash script then calls rsync to push the changes.

If you're not familiar with [rsync](#), you should be. It works similarly to robocopy or xcopy, but minimizes the amount of data sent over the line by doing small, differential changes. It copies files via SSH.

Most will already have rsync installed by default on their computer, but fswatch won't be. You can either compile and install the source from scratch, or just grab it via Homebrew or MacPorts:

```
# MacPorts
$ port install fswatch

# Homebrew
$ brew install fswatch
```

Now, we can combine both utilities like so:

```
fswatch -o /Users/brycematheson/Sites | xargs -n1 -I{} ./sync.sh
```

For now, you can ignore the `xargs -n1 -I{}` part. Just know that it works. What we really want to pay attention to is the `fswatch -o /Users/brycematheson/Sites` part. We're telling fswatch to keep an eye on the "Sites" directory, and then run the "sync.sh" bash script, whenever a change is detected. "sync.sh" looks like this:

```
rsync -r -a -v -e "ssh -l root" --exclude "/phpmyadmin" --exclude "/images"
/Users/brycematheson/Sites/mysite/ 124.18.106.212:/var/www/public_html
afplay "/System/Library/Sounds/Blow.aiff"
```

The command above will copy the source directory of `"/Users/brycematheson/Sites/mysite/"` to `"/var/www/public_html"`, excluding the `"phpmyadmin"` and `"images"` folders. It will also log in via the username `"root"` at the IP address of `124.18.106.212`. **Note that the source directory ends in a slash, but the destination directory does not.** This is crucial, otherwise rsync will give you fits.

I would highly recommend [setting up private/public SSH keys](#) allowing you to use rsync without having to enter in a password each time.

Each time rsync successfully copies files to your destination, you'll be greeted with a short sound, just making it easier to audibly know that a change has happened.

That's it! Happy coding with auto-sync.