Downloading a File via FTP

Just viewing what's on an FTP server isn't all that useful. You will almost always want to download a file from the server. Let's find out how to download a single file:

```
from ftplib import FTP

ftp = FTP('ftp.debian.org')
print(ftp.login())
#'230 Login successful.'

print(ftp.cwd('debian')  )
#'250 Directory successfully changed.'


out = 'README'
with open(out, 'wb') as f:
    ftp.retrbinary('RETR ' + 'README.html', f.write)
```

For this example, we login to the Debian Linux FTP and change to the debian folder. Then we create the name of the file we want to save to and open it in write-binary mode. Finally we use the ftp object's **retrbinary** to call RETR to retrieve the file and write it to our local disk. If you'd like to download all the files, then we'll need to a file listing.

```
import ftplib
import os

ftp = ftplib.FTP('ftp.debian.org')
ftp.login()
ftp.cwd('debian')
filenames = ftp.nlst()

for filename in filenames:
    host_file = os.path.join(
        'ftp_test', filename)
    try:
        with open(host_file, 'wb') as local_file:
            ftp.retrbinary('RETR ' + filename, local_file.write)
    except ftplib.error_perm:
        pass

ftp.quit()
```

This example is fairly similar to the previous one. You will need to modify it to match your own preferred download location though. The first part of the code is pretty much the same, but then you will note that we call **nlst** which gives us a list of filenames and directories. You can give it a directory to list or just call it without and it will assume you want a listing of the current directory. Note that the nlst command doesn’t tell us how to differentiate between files and directories from its results. For this example though, we simply don’t care. This is more of a brute force script. So it will loop over the list returned and attempt to download them. If the “file” happens to actually be a directory, then we’ll end up creating an empty file on our local disk with the same name as the directory on the FTP server.

There is an MLSD command that you can call via the **mlsd** method, but not all FTP servers support this command. If they do, then you might be able to differentiate between the two.