Connecting to an FTP Server

The first thing we need to do is find an FTP server to connect to. There are many free ones you can use. For example, most Linux distributions have FTP mirrors that are publicly accessible. If you go to Fedora's website (https://admin.fedoraproject.org/mirrormanager/) you will find a long list of mirrors that you can use. They aren't just FTP though, so be sure that you choose the correct protocol or you will receive a connection error.

For this example, we will use **ftp.cse.buffalo.edu**. The official Python documentation uses **ftp.debian.org**, so feel free to try that as well. Let's try to connect to the server now. Open up the Python interpreter in your terminal or use IDLE to follow along:

```
from ftplib import FTP
ftp = FTP('ftp.cse.buffalo.edu')
print (ftp.login())
#'230 Guest login ok, access restrictions apply.'
```

Let's break this down a bit. Here we import the **FTP** class from ftplib. Then we create an instance of the class by passing it the host that we want to connect to. Since we did not pass a username or password, Python assumes we want to login anonymously. If you happen to need to connect to the FTP server using a non-standard port, then you can do so using the **connect** method. Here's how:

```
from ftplib import FTP
ftp = FTP()
HOST = 'ftp.cse.buffalo.edu'
PORT = 12345
ftp.connect(HOST, PORT)
```

This code will fail as the FTP server in this example doesn't have port 12345

open for us. However, the idea is to convey how to connect to a port that differs from the default.

If the FTP server that you're connecting to requires TLS security, then you will want to import the **FTP_TLS** class instead of the **FTP** class. The **FTP_TLS** class supports a keyfile and a certfile. If you want to secure your connection, then you will need to call **prot_p** to do so.