Installing Python Libraries on GoDaddy Shared Hosting

If you've got a GoDaddy shared hosting account, you're probably none too pleased with the Python support. After all, SSH in, type "python", and you get a message telling you your version is horribly out of date:

```
$ python
Python 2.4.3 (#1, Nov 11 2010, 13:34:43)
```

But after six years of hacking around on their boxes, I've learned that the trick to getting the most of your $9/month is just knowing what hoops to jump through:

```
$ /usr/local/bin/python2.7
Python 2.7.3 (default, May 25 2012, 16:43:26)
```

GoDaddy's Support Team is very responsive, but they're not so technically adept. You get a lot of boilerplate, shunted off to a lot of doc pages, and don't get a ton of specific instruction. This bit on Python libraries is pretty typical. But installing new libraries is relatively easy, so I'll walk you through it.

The package I want to install is BeautifulSoup, a web-scraping library. I started off by trying `easy_install` as suggested, but as you'll soon find out, shared accounts don't have the access to modify files in Python's packages directory.

This means you'll have to create your own directory and let Python know to use it. Start by downloading this virtual-python.py file (right-click, save link as...) and then uploading it to your GoDaddy account. I chose my home (~)

folder, though you can place it any file where you have the ability to create and modify files ("virtual Python" details).

Once the file's uploaded, you'll need to run it. virtual-python.py detects the version of Python you're using, and creates links to already-installed packages. This means to make your "virtual" version the best available, you'll need to use GoDaddy's "hidden" 2.7.3 installation as shown below:

```
$ /usr/local/bin/python2.7 virtual-python.py
```

This will create a bunch of new files and directories, and if successful, should end with the message telling you you're ready to download and install ez_python.py (right click, save link as…). After you've downloaded it, install it to the same directory, and then run it, being sure to use your the new "virtual" Python you created in the previous step:

```
$ ~/bin/python ez_setup.py -U setuptools
```

The "-U setuptools" reinstalls `setuptools` into your new directory—for some reason, ez_setup.py detects your system's existing installation (the one in the folder you can't modify) and decides that nothing new needs to be installed. Also, if you've installed virtual-python.py in a different folder, replace the "~" above with that path.

Finally, we tell our shiny new virtual Python to use our newly-installed easy_install.py to install the library we wanted (Beautiful Soup, in this example):

```
$ ~/bin/python ~/lib/python2.7/site-packages/easy_install.py
beautifulsoup4
```

Phew—not the easiest workaround, or the most concise package installation commands, I'll grant you. But for the price, I think it's well worth the effort. I hope you do, too.