Installing Python 3 and Flask on GoDaddy

Jordan Ireland Oct 10, 2019 · 8 min read ★

Recently I decided to host my Python app on my GoDaddy hosting, and spent hours debugging since the documentation for both installing Python and Flask on something other than Heroku/Apache standalone server is hard to come by. The reason I wanted to set it up on my hosting platform, was due to Heroku and others not being able to

The Github for the following files can be found here

In the following weeks, I will update this article with links to expand on each topic for newer users and will also add more to the troubleshooting as newer issues appear.

Table of Contents:

- Prerequisites
- Installing Python
- Installing Flask
- Set Up Your Flask App
- Enable SSH
- Troubleshooting

Prerequisites

TOP^

Before beginning there are a few things you need to download or prepare yourself for.

These instructions have been tested on GoDaddy Shared and GoDaddy Business hosting.[1]

You will need:

- A GoDaddy Hosting account with at least 1 website hosted.
- PuTTy: PuTTy is a free SSH console that is used to connect to GoDaddy Servers. This will be used to install Python.
- SSH Access through GoDaddy. (Instructions below)

- Programming IDE. I use PyCharm for Python/CGI and Visual Studio for HTML, CSS, and Javascript.

Installing Python

TOP^

Login To PuTTy (or other SSH terminal):

Once PuTTy is installed, you can start installing Python. Open up PuTTy console.

PuTTy Icon. Don’t open PuTTyGen

[2]**Type** your hostname, which will be your IP address from when you enabled SSH on GoDaddy. This can be found in your hosting settings under the `Server` sub-menu. Your port will almost always be `22` , but double check to make sure. This can be found in the SSH enable menu. Click open and it will start to connect.

PuTTy will create a secure connection to GoDaddy servers; it will prompt you to enter the username to the **main** FTP account. This will be found under hosting settings, where you enabled SSH. It will be a string of alpha characters and numbers, all lower case. The password will be the same password used to log in to GoDaddy.

Once logged in to PuTTy, type the following to start the installation process.

```
## Download Python 3.5.7 (latest version GoDaddy supports as of
writing)
$ wget https://www.python.org/ftp/python/3.5.7/Python-3.5.7.tgz

## Once wget is done downloading

$ tar xvzf Python-3.5.7.tgz
```

```
## This should take you to the main Python Directory

$ ./configure --prefix=$HOME/.local

## This step sets up your configuration and makes files

$ make
$ make install

## This will create the Python files and folders, but will take a
long time. The last command will install pip and setuptools which
will be used to install any package you want
```

The steps above will download and install Python, pip, and setup tools. If you have any errors, see the Troubleshooting section below. (This will be updated as more errors are discovered)

The next steps are required to finish the installation of Python and make sure environment variables are set up.

```
$ cd $home
$ vi .bash_profile

## Press enter and change the file to equal this EXACTLY

PATH=$HOME/.local/bin/$PATH
export PATH

## Once done, Type :wq and press enter. This will save your file and
close it

$ python3 -V
>> Python 3.5.7
```

There you go! If you just wanted to install, Python3 onto GoDaddy, you can stop here! If you'd like to deploy a Flask or Heroku app on GoDaddy, continue the article.

Note: If you are using a subdomain, make sure the subdomain folder is inside public_html, and these instructions will be the same except you will need to add `subdomain_name/` to these instructions.

```
## Install Virtual Environment Package
$ python3 pip -m install virtualenv

## Navigate to your CGI-BIN folder inside public_html
$ cd $home
$ cd public_html/cgi-bin

## Create Virtual Environment. 'venv' can be named whatever you
want, but just change the other code accordingly.
$ virtualenv venv

## Activate the virtual environment
$ source venv/bin/activate
$ pip install Flask
$ deactivate
```

Flask is now installed and can be used in the next steps to set up your Flask App to run on your GoDaddy Hosting.

You can also install any required packages, just like a local machine with the above steps. Make sure you activate the virtual environment with `source venv/bin/activate` before installing, otherwise it won't work.

Set Up Your Flask App

TOP^

In order to publish an app with GoDaddy hosting, you will need to turn it into something GoDaddy can use. For this, we will have to use a package called `wsgiref`. There is no need to download anything, as this package is included in Python since 2.7.

understand before altering any options). The static folder is optional, it will allow you to use CSS and Javascript inside your Flask app, but that’s a different article.

```
public_html/
├── .htaccess
└── cgi-bin/
    ├── app.cgi
    ├── app.py
    ├── templates/
    │   └── home.html
    └── venv/
```

Make a new file inside `cgi-bin` and call it `app.cgi` . The following code should be placed inside this file and changed to match your website settings.[3]

The username will be the same as when you logged in to PuTTy or other SSH terminal.

```
#!/home/USERNAME/.local/bin/python3
import os
import sys

sys.path.insert(0,
                '/home/USERNAME/public_html/python/cgi-
bin/venv/lib/python3.5'
                '/site-packages')

from wsgiref.handlers import CGIHandler

from app import app

class ProxyFix(object):
    def __init__(self, app):
        self.app = app

    def __call__(self, environ, start_response):
        environ['SERVER_NAME'] = ""
        environ['SERVER_PORT'] = "80"
        environ['REQUEST_METHOD'] = "GET"
        environ['SCRIPT_NAME'] = ""
        environ['QUERY_STRING'] = ""
        environ['SERVER_PROTOCOL'] = "HTTP/1.1"
        return self.app(environ, start_response)
```

```
app.wsgi_app = ProxyFix(app.wsgi_app)
CGIHandler.run(app)
```

Let's break the code down a little bit. The first line is the called the `shebang` this tells the program what shell to use. In this case, we are running python so we need to point it to the python shell, which is located in the `/local` folder.

Next you import your app, just like a normal __init__.py file. `class ProxyFix` initializes the app and makes sure to set the correct environment variables. You can change and delete the environment variables as you need, but these should be working for most projects. The major ones that will be needed to change are `'QUERY_STRING' &` `'REQUEST_METHOD'` . If these are set, all calls to the Flask app will be GET requests and the Query String will be `NONE` .

Create a another new file called `app.py` inside `cgi-bin` and type the following code. This code is setting up a test app, so Flask has something to display.

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('home.html')
```

Inside `cgi-bin` create a folder named `templates` and a file inside that folder called `home.html`

```
<!DOCTYPE html>
<html>
    <head>
        <title>Home</title>
    </head>
    <body>
        <h2>It Works! Congrats!</h2>
    </body>
</html>
```

point all incoming requests to your app.cgi file and make sure the only thing that runs is your Flask app. Change username to main FTP account you used for SSH terminal.

```
Options +ExecCGI
AddHandler cgi-script .py
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ /home/USERNAME/public_html/python/cgi-
bin/app.cgi/$1 [L]
```

I will be completely honest here, I am no expert when it comes to .htaccess files, but I was able to get this working with some fiddling around. Make sure you copy paste this exactly. If it is one space or 1 letter off, this will not work.

There you go! Once this is all uploaded and finished, type your URL and cross your fingers! You can start expanding and make sure you import all of your required packages. You can most likely create a script that will check for installations and install them for you, but that's not for this article.

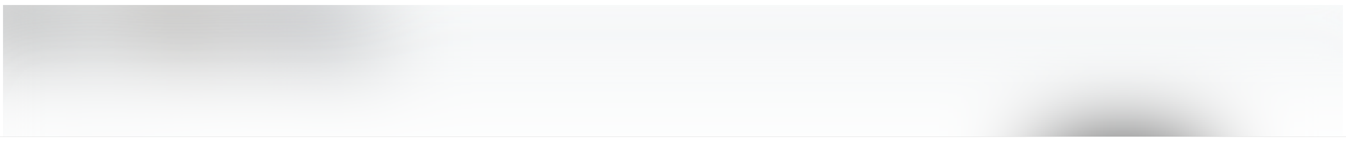
The most important thing is that your app doesn't print or output anything that isn't HTML format. That is one drawback about publishing any Flask app on any platform.

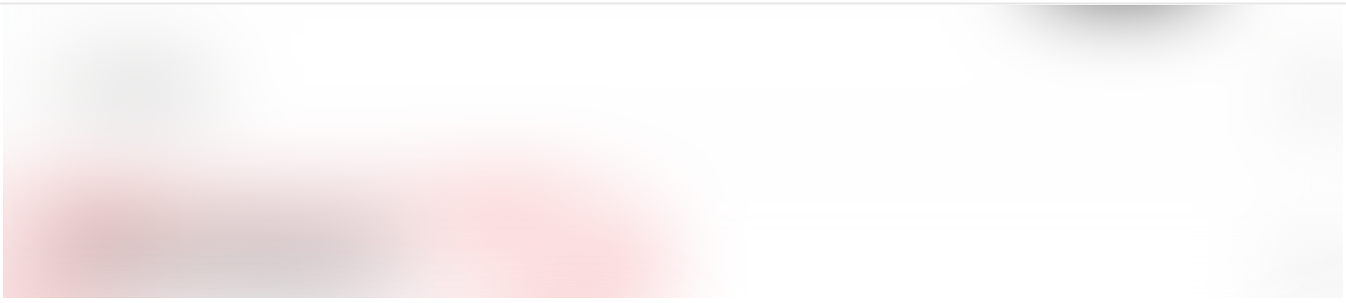
All Python and CGI files inside GoDaddy must have permissions set to 755. Make sure to check that before trying to open the URL.

How To Enable SSH

TOP^

- Login to GoDaddy Hosting
- Click your hosted website:





- On the right, click the `Server` button under your `Settings` Menu
- Click `Manage` next to `SSH Access` .
- From there all you have to do is Flip the switch to `On`

NOTE: Remember your port number (22), Username, and IP address. This will be on the Settings menu under the Account tab, but we will use these during Python Installation.

NOTE 2: The cPanel password is the password you use to log into GoDaddy.

Troubleshooting

TOP^

I'm getting Error 500 when trying to open the flask app through the URL.

- Check the error logs found in your GoDaddy cPanel. It should give you a good idea of what's going on. Most commonly a package isn't installed or the Python installation failed somewhere without throwing an error.
- Alternatively, make sure your app doesn't print or try to access the console at all, that will crash the app.
- The environment variables could also be incorrect. The most common is SERVER_NAME and SERVER PORT, these must be exactly how they are in the script above.

Python is failing to install. It's throwing an error during `make` or `make install`.

- Most likely, it's failing to install due to an error with `ctypes` . These were included in Python 2.7, but GoDaddy doesn't have them installed with their version of Python. Try downloading an older or newer version of Python. Flask is compatible with 2.7.* and up, not including 3.0.*–3.4.*
- Make sure you have read/write access to the folder you're trying to install Python in.

I'm getting a 404 error when trying to open the app.

- Check your .htaccess file, and make sure it's exactly like mine.
- Make sure the @app.route has a valid endpoint ie: /home, /, /index

[1] *These have not been tested on VPS hosting through GoDaddy or Windows hosting. As far as I know, GoDaddy Windows doesn't support Python.*

[2] *This code was taken from the GoDaddy blog post here and has been modified to work with Flask and has had some custom code added.*

[3] *Credit to Flask documentation for giving a starting point for users. Documentation Link*

By Towards Data Science

Every Thursday, the Variable delivers the very best of Towards Data Science: from hands-on tutorials and cutting-edge research to original features you don't want to miss. Take a look.

Get this newsletter

Python Godaddy Flask Tutorial Linux