

Python

Posted February 22nd, 2024

Python is a popular language used for web development

<https://developers.google.com/edu/python>

1. build python

```
./configure  
make  
make install
```

2. install gunicorn

```
pip3 install gunicorn
```

3. in apache enable proxy modules

```
LoadModule proxy_module modules/mod_proxy.so
```

```
LoadModule proxy_http_module modules/mod_proxy_http.so
```

4. add ProxyPass entry in vhost

```
ProxyPass "/pybase/" "http://127.0.0.1:8000/"
```

5. configure gunicorn.service file in /usr/lib/systemd/system/gunicorn.service

```
===== Start gunicorn.service File =====
```

```
[Unit]
```

```
Description=The Apache HTTP Server
```

```
After=network.target
```

```
[Service]
```

```
MAINPID = /run/cmdmb/cmdmb.pid
```

```
WorkingDirectory = /m/member/pybase
```

```
ExecStartPre = /bin/mkdir /run/cmdmb
```

```
ExecStart=/usr/local/bin/gunicorn --workers=4 myapp:app --chdir="/m/member/pybase" --pid /run/cmdmb/cmdmb.pid
```

```
ExecReload=/bin/kill -s HUP $MAINPID
```

```
ExecStop=/bin/kill -s HUP $MAINPID
```

```
ExecStopPost = /bin/rm -rf /run/cmdmb
```

```
PrivateTmp=true
```

```
[Install]
```

```
WantedBy=multi-user.target
```

```
===== End gunicorn.service File =====
```

A) definition of a function

```
def funcname(arg1,arg2):
```

B) spacing matters - indentation defines block

C) definition of if statement

```
if (a == 1):  
    a += 1;
```

D) for statement

```
for (x in environ):  
    data += x+environ[x];
```

E) Templates

Jinja2 is the most prevalent template engine

sample code to load from file:

```
import jinja2  
def display_template():  
    templateLoader = jinja2.FileSystemLoader(searchpath="./templates")  
    templateEnv = jinja2.Environment(loader=templateLoader)  
    templateFile = "template.html"  
    template = templateEnv.get_template(templateFile)  
    return template.render();
```

F) getting data from the database

- basic level method (cursor)

```
-----  
import mysql.connector  
def get_data(sql):  
    cnx = mysql.connector.connect(
```

```

user='membermaster',
password='XXXXXX',
host='127.0.0.1',
database='mainmemberdb')

```

```

data = "<table>\n"
if cnx and cnx.is_connected():
    with cnx.cursor() as cursor:
        result = cursor.execute(sql)
        rows = cursor.fetchall()
        for row in rows:
            data += "<tr>\n"
            for col in row:
                data += " <td>"+col+"</td>\n"
            data += "</tr>\n"
data += "</table>\n\n"
cnx.close()
return data
---- end code -----

```

-More sophisticated is to use pandas to load the data to a grid
pandas can output the data in a number of formats
pandas is a library that allows data to be loaded into a grid
that can then be displayed.

```

-----
def get_table(sql):
    try:
        cnx = mysql.connector.connect(
            user='membermaster',
            password='XXXXXX',
            host='127.0.0.1',
            database='mainmemberdb')
        results = pd.read_sql(sql,cnx)
        cnx.close()
        # html table has systables as a css class systables in addition to
        # dataframe... e.g. <table class='dataframes systables'>
        return results.to_html( classes= "systables")
    except Exception as e:
        cnx.close()
        return str(e)
---- end code -----

```

G) Get an argument from command line

```

import sys
def main():
    if (len(sys.argv)>=2:
        print("ARG 1:",sys.argv[1]);

if __name__ == '__main__':
    main()

```

H) repeat string... * syntax

```

print("foo"*3);
>>foofoofoo

```

I) From the command line - help(keyword)
help(len)

J)To convert numbers to strings
mystring = str(10);

K) numbers:

1. no ++ - can't do a++
2. integer division is two // e.g. 2.4 // 2 = 1

L) raw strings

```

raw= r'this\t\n and that'
multi= """ hello
world """

```

M) slices

```

s="hello";
something = s[1:4]
>>> 'ell'
s[-1]
>> 'o'

```

```
N) strings
[f strings]
value = 2.791514
print(f'approximate value = {value:.2f}')
```

```
O) if statements
if a==b:
    print("a equals b")
elif c==d:
    print("c equals d")
else:
    print("magic")
```

```
P) and/or ( there is no && or || )
if a and b:
    print(true)
if a or b:
    print(false)
```

```
Q) for/in
for key in list:
    iterate here
if "moe" in list:
    do if moe is in list
```

```
R) dict
mydict = {key1:value1,key2:value2}
mydict = {}
usage: myvar = mydict[key1];
mykeys = mydict.keys();
myvalues = mydict.values();
myitems = mydict.items(); // tuple
for k,v in myitems : print (k,"=",v)
s = : %(word)s %mydict
del mydict[0];
del mydict[-2:0]
```

```
S) files
#open file
f = fopen("filaneme.txt", "rt", encoding='utf-8')
for line in f:
    print (line, end="")
f.close();
```

```
T) regular expressions
import re
mystr = " hello 123 456";
mymatch = re.search(mystr, '\d+');
mymatch = re.findall(mystr, "\d+");
mymatch = re.sub(mystr, "123", "\d+");
```

```
U) major important libraries
os lib: os.listdir(dir), os.path.join(dir,filename), os.path.abspath(filename), os.path.dirname(filename)
        os.path.basename(filename), os.path.exists(filename), os.mkdir(path), os.makedirs(path1)
shutil lib: shutil.copy(source,target)
subprocess:
    output = subprocess.check_output(cmd, stderr=subprocess.STDOUT) -
    subprocess.Popen
    subprocess.call
urllib lib:
    ufile = urllib.request.urlopen(url);
    text = ufile.read()
    info = ufile.info()
    baseurl = ufile.geturl()
    urllib.request.urlretrieve(url, filename)
    urllib.parse.urljoin(baseurl, url)
    from urllib.request import urlopen
uriparse lib:
```