Python Modules for Profitability

MIKE PAVLAK

Profound Logic

mikepavlak@gmail.com

Today's schedule

12:00 - Introduction to Python for RPGers (Mike)

1:30 - Python and Data Access (Mike)

3:00 - RPG Makes Friends with Open Source Apps (Richard)

4:30 - Python Modules for Profitability (Mike)

7:00 - Pass the Flask & Quickly Pour IBM I Python Web Apps & Services (Richard)

Agenda

Recent updates and announcements

Module Basics
- Passing arguments
- Paths

PSL – Python Standard Library

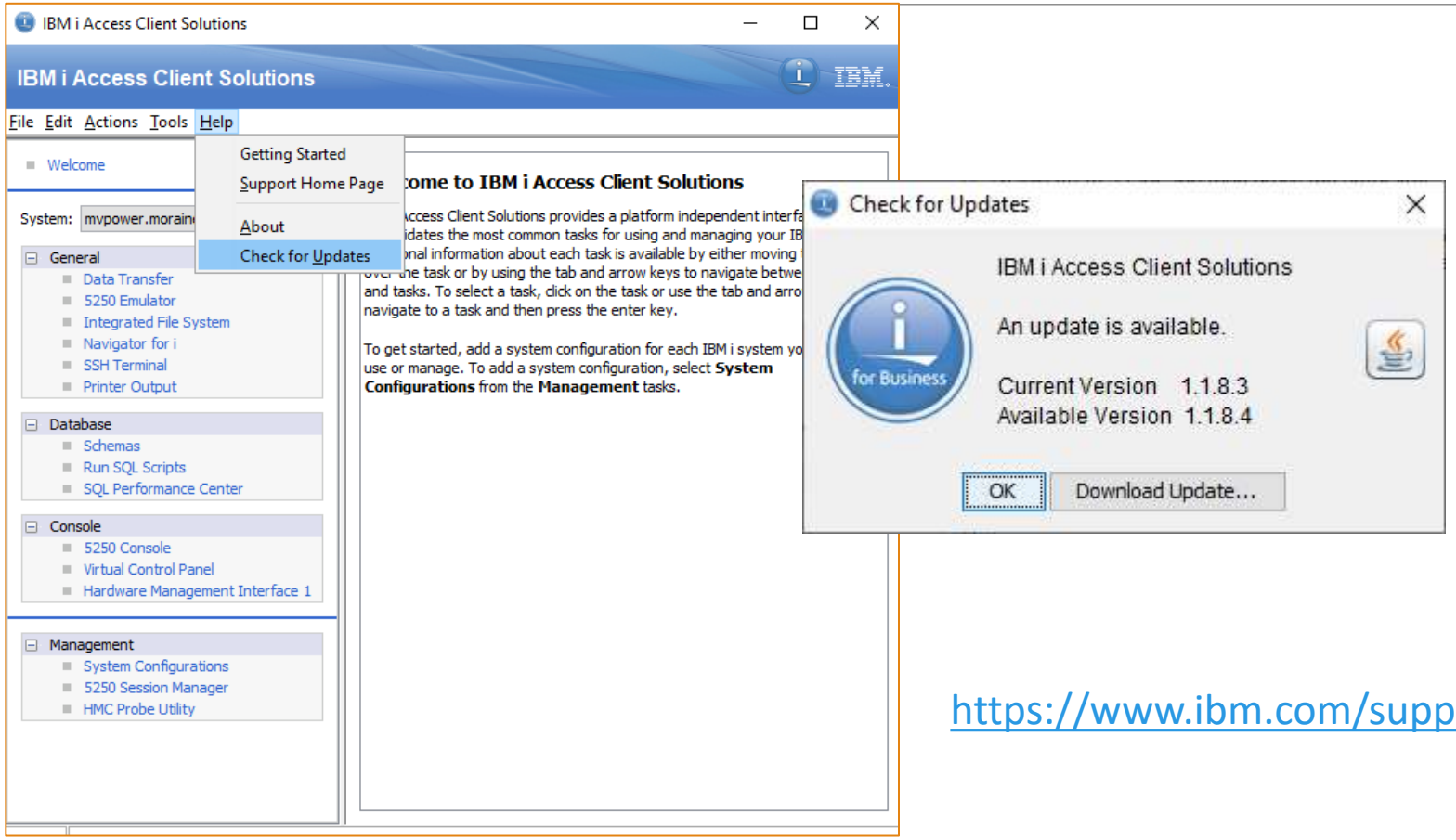
Examples
- Web service
- 3rd Party
- PDF & Spreadsheet
- Stocks

Flask

Recent Updates

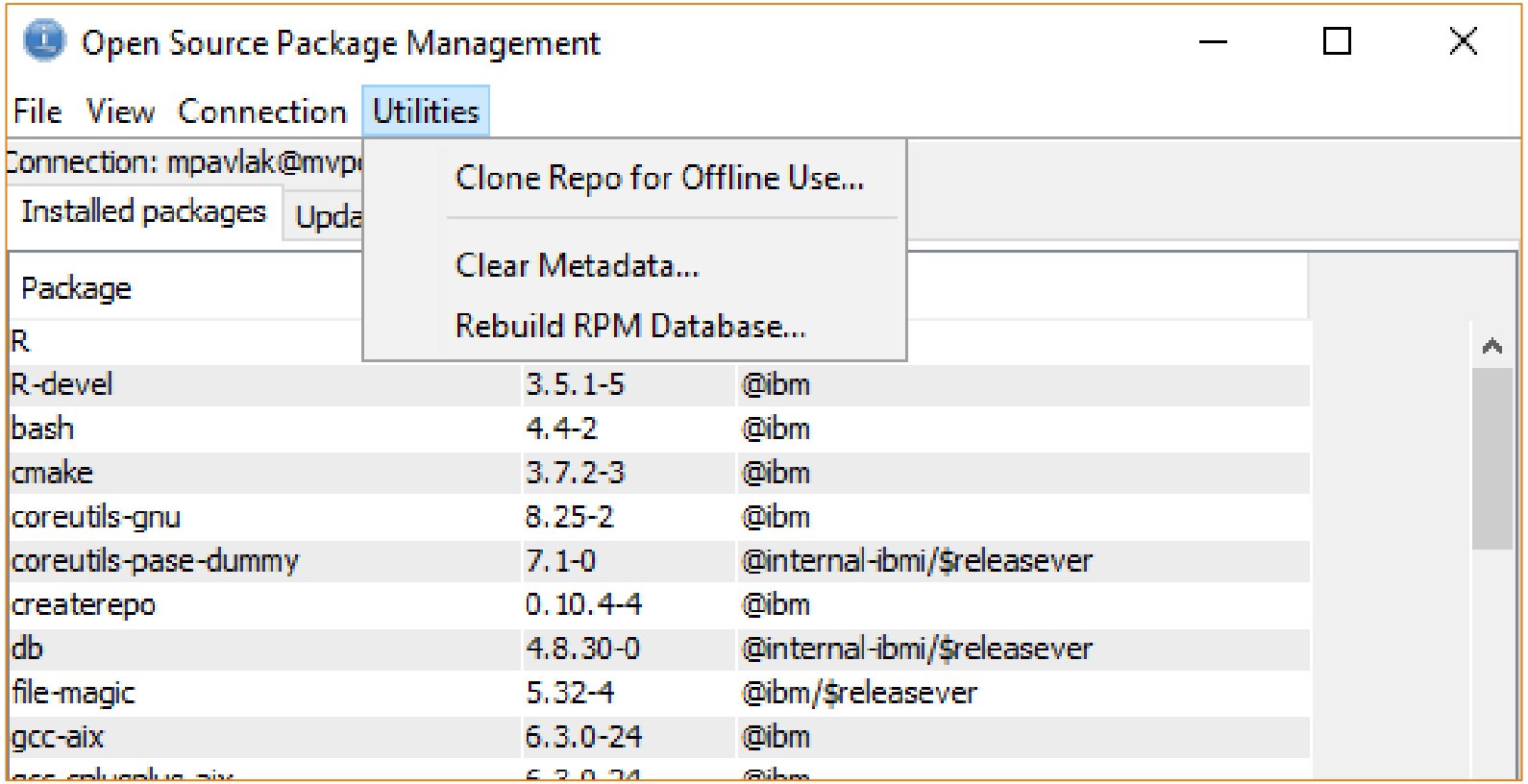
ACCESS CLIENT SOLUTIONS

PSA: Update ACS

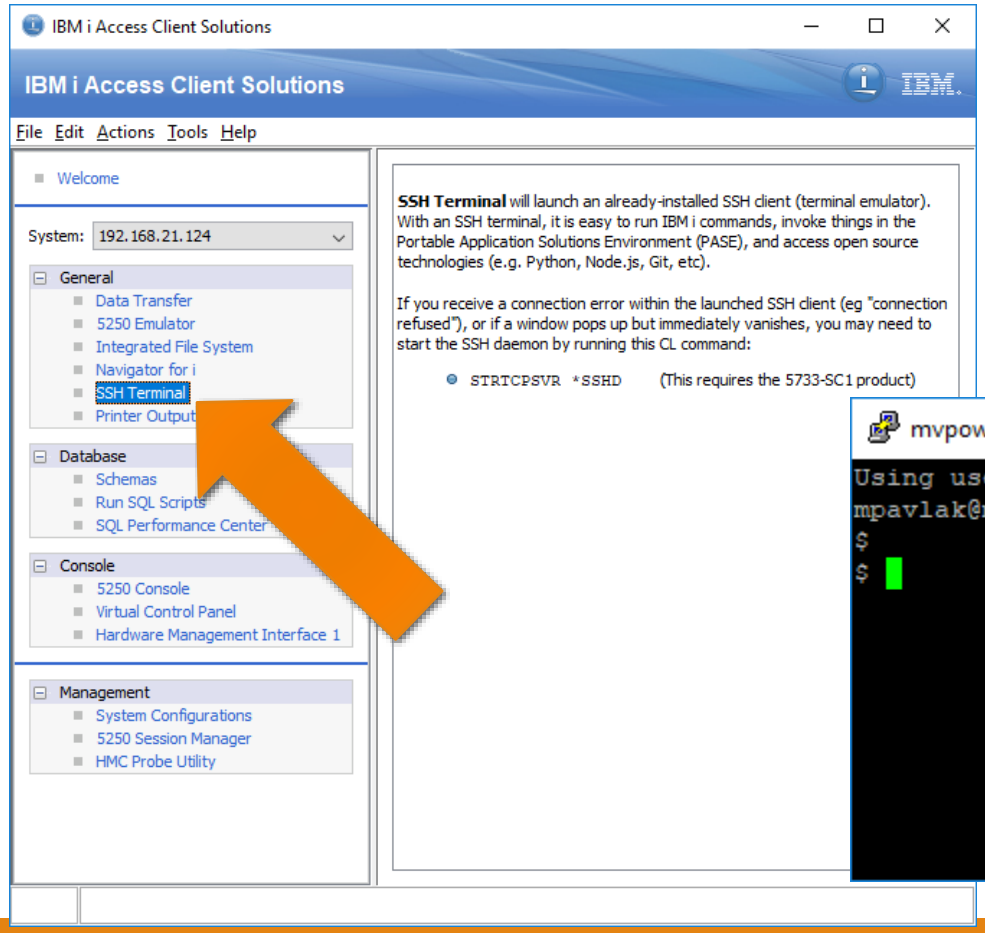


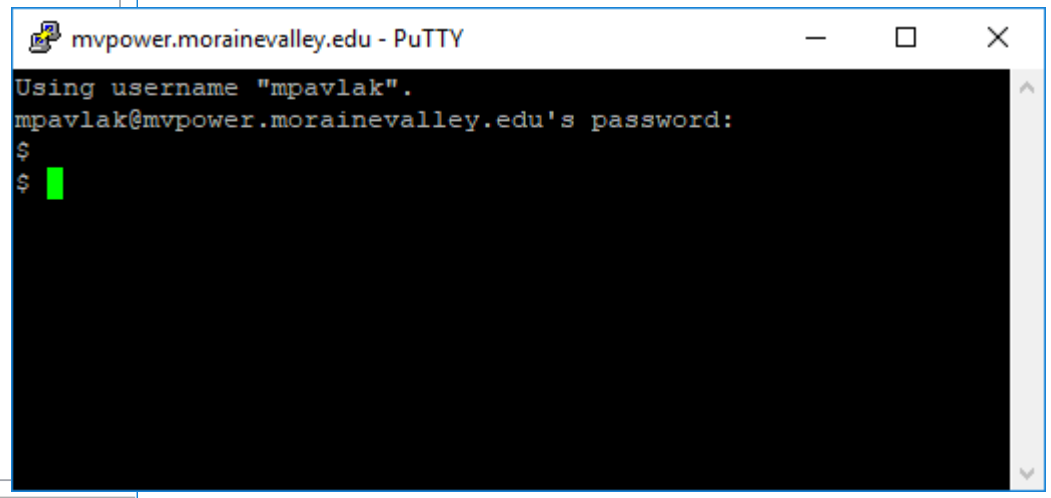
https://www.ibm.com/support/pages/ibm-i-access-client-solutions

ACS features in 1.1.8.4, 5



Shell available in ACS (short PuTTY)





Encrypted

BASH, etc.

Linux alignment

Module basics

PASSING ARGUMENTS, ETC.

sys module list: argv

Need to import

First argument is ALWAYS the script name

```
from sys import argv

print(argv)
```

```
bash-4.4$ python3 adv03argv.py
['adv03argv.py']
bash-4.4$
```

Something a little more practical

```
bash-4.4$ python3 adv03argv.py "Mikey" 1234 "IBM i"
['adv03argv.py', 'Mikey', '1234', 'IBM i']
bash-4.4$
```

https://docs.python.org/3/library/sys.html

Use the values…

```
from sys import argv

script, name, number, system = argv

print("\nThe script name is", script)
print("Your name is", name)
print("Your number is", number)
print("Your system is", system)
```

```
The script name is adv03argv.py
Your name is Mikey
Your number is 1234
Your system is IBM i
bash-4.4$
```

```
from sys import argv

script = argv.pop(0)

print("\nThe script name is", script)

for name in argv:
    print("\nHello", name)
```

Use many values…

```
from sys import argv

script = argv.pop(0)

print("\nThe script name is", script)

for name in argv:
    print("\nHello", name)

print("\n\nYou gave me", len(argv), "names")
```

```
bash-4.4$ python3 adv04argvmany.py "Pete" "Keith" "John" "Roger"

The script name is adv04argvmany.py

Hello Pete

Hello Keith

Hello John

Hello Roger


You gave me 4 names
bash-4.4$
```

Module Basics

PYTHON PATHING

Where does Python find stuff?

Current Directory

- os.cwd
- Can navigate

PYTHONPATH

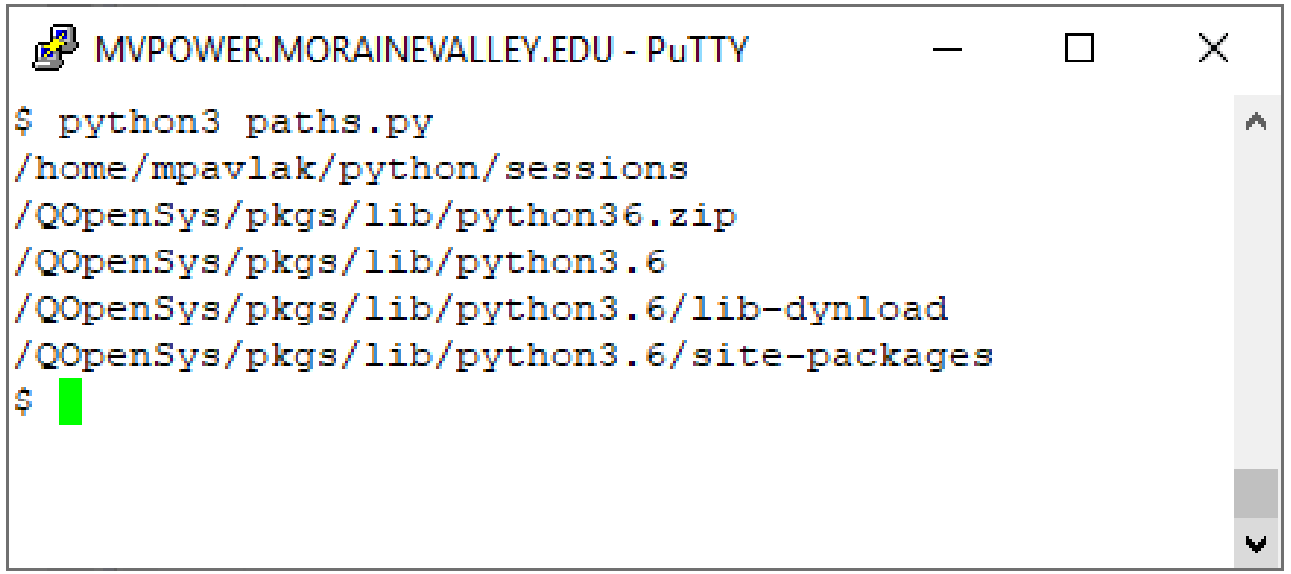
- Augment the search path for importing Python modules

Default Directory

- Where the script resides

Where does Python find stuff?

Dump the paths




```
import sys

pathList = sys.path

for path in pathList:
    print(path)
```

Modules and functions

os.path module

- Built in module for system related stuff

Pathlib

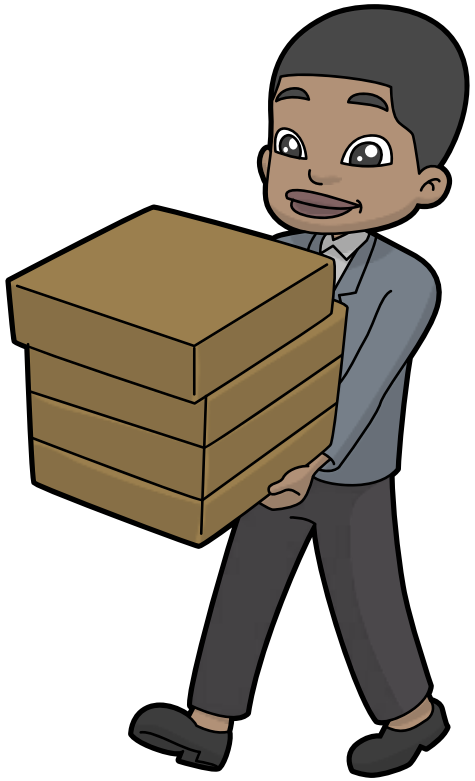
- Object Oriented path manipulation

os.cwd

- Display current working directory

Modules: The Secret Sauce of Python

If you invite Python to a party…



PHP

Python

What is a module

Code library – hunks of Python code

They come from the Python community

- PSL: Python Standard Library
- Home grown
- Third party

Usually Python, but not necessarily…

Designed to augment, not replace

Think of them as copybooks

Tested, trusted, useful…

Base Python

Python tends to be lightweight

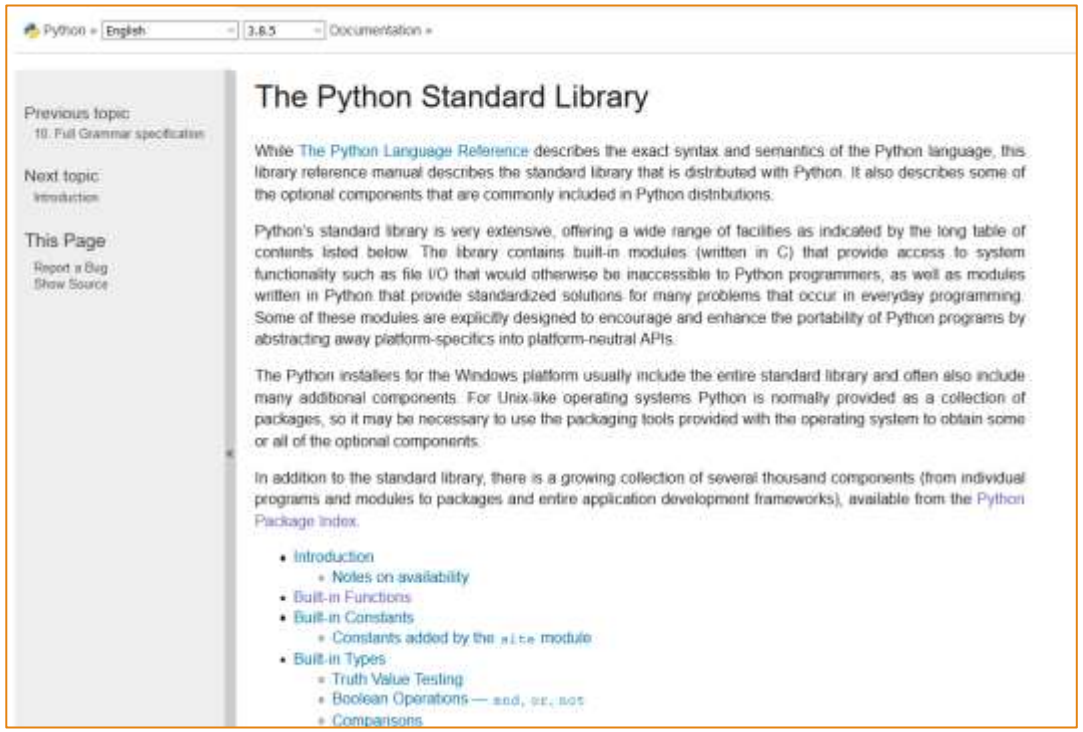
- Only about 60 built in functions
- print(), str(), round(), set(), etc.
- Keeps performance in check
- Minimalist or least access concept

If you need something else…use a module & bring it in only if/when you need it!

It all begins with the PSL

Python Standard Library

- Collection of over 200 sets of functions
- Math, os, decimal, random, etc.
- Shipped with Python
- Must be “imported” in order to use
- Simple, powerful, accepted…
- Assumed to be available

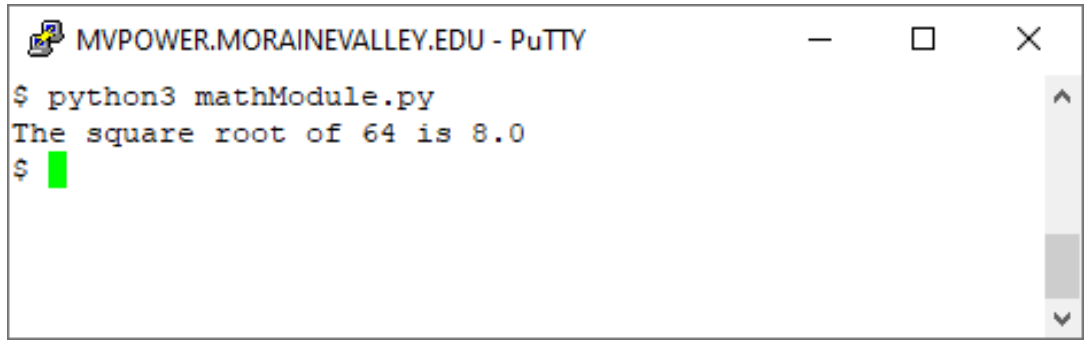


https://docs.python.org/3/library/

Math module example

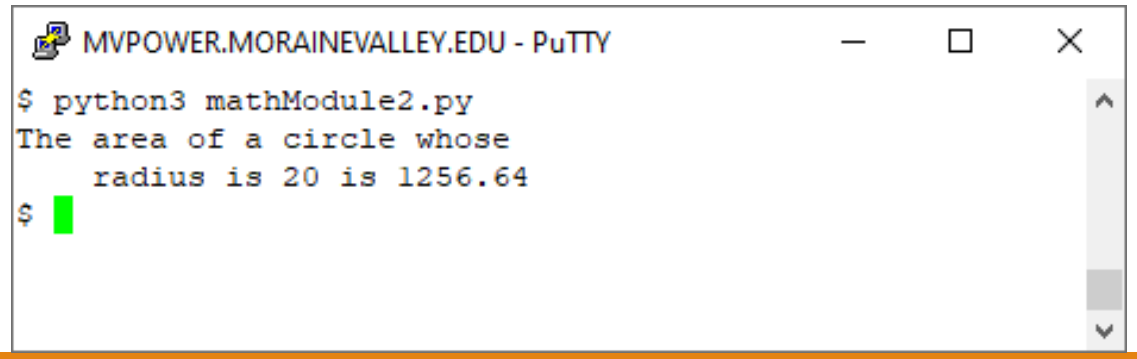
Let's say we want to do a square root or use pi

```
import math

x = 64
y = math.sqrt(x)

print("The square root of", x, "is", y)
```



https://docs.python.org/3/library/math.html

```
import math

# Calculate the area of circle
radius = 20
area = round((math.pi * radius**2),2)

print("""The area of a circle whose
    radius is""", radius, "is", area)
```



Other solid PSL candidates

OS

Sqlite3

Datetime

Decimal

Copy

Pprint

Random

Logging

Email

XML & JSON

Gettext

Doctest

Locale

Turtle

Pydoc

Sys

Site

Cmd

http

And many more

Module Examples

WEB SERVICES

Requests module

"Apache2 HTTP library written in Python for human beings."

Make REST calls from a single line in Python script

```
>>> r = requests.get('https://api.github.com', auth=('user', 'pass'))
>>> r.status_code
204
>>> r.headers['content-type']
'application/json'
>>> r.text
...
```

https://requests.readthedocs.io/en/master/

Example Access Watson

```
import requests
import json

textinput='Hello'
model='en-es'
apikey= 'NOT A REAL API KEY…'
url='https://gateway.watsonplatform.net/language-
translator/api/v3/translate?version=2018-05-01'
data={'text':textinput,'model_id':model}

r = requests.post(url,auth=('apikey', apikey),json=data)

jdata = json.loads(r.content)

translatedtext=jdata['translations'][0]['translation']
print(translatedtext, '\n', jdata)
```

Bring in some functionality

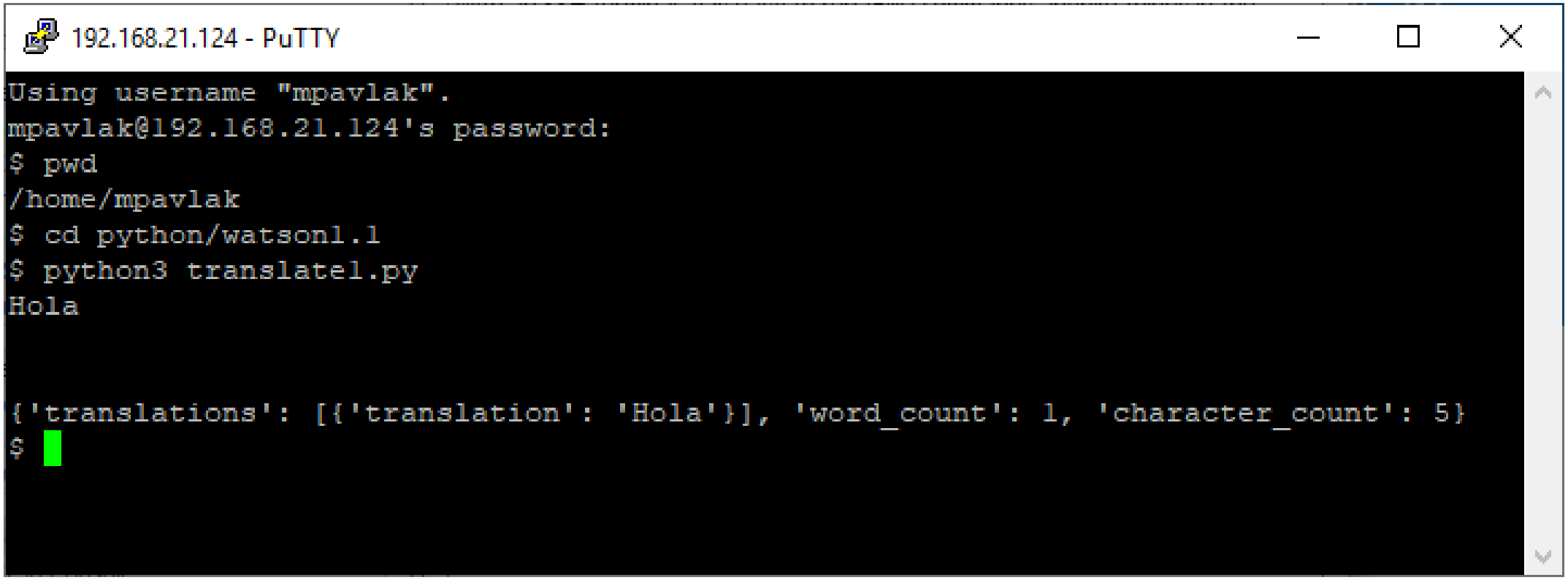
Set some values

Call service using requests

Call JSON to format data

Display something

Output


192.168.21.124 - PuTTY
Using username "mpavlak".
mpavlak@192.168.21.124's password:
$ pwd
/home/mpavlak
$ cd python/watson1.1
$ python3 translate1.py
Hola
{'translations': [{'translation': 'Hola'}], 'word_count': 1, 'character_count': 5}
$

Popular third party modules

Data Science
- Mumpy, Pandas, maPlotLib, nltk

Machine Learning
- Tensor Flow, Keras, Pytorch

Web
- Django, Flask, Bottle

Other
- Selenium, xlsxwriter,

Third parties require an extra step

PSL is shipped with python, already installed

Third parties need to be installed in the Python directories

pip == Python Installer Program ***Note pip3 on IBM i*

Reads from repositories

- Pypi.org == public repository (Python Package Index)
- Others can be added to pip.conf (PerForce with PHP)

Can point to private repositories at time of install

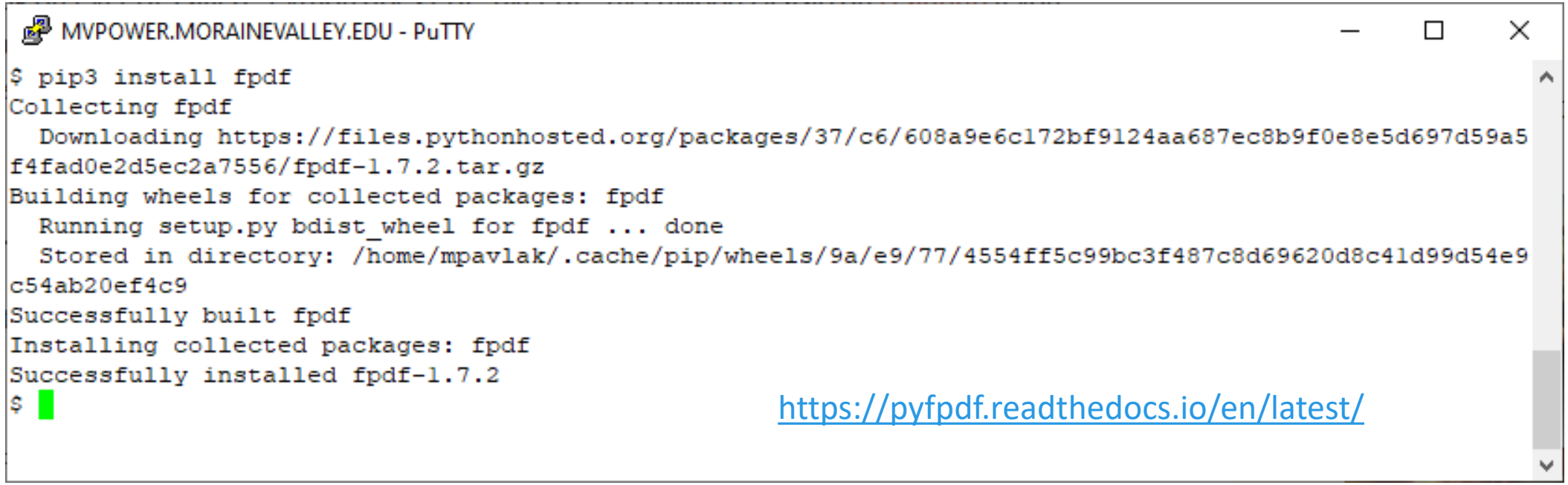
https://docs.python.org/3/installing/index.html

fpdf

Simple example of installing with fpdf.

- Create PDF files

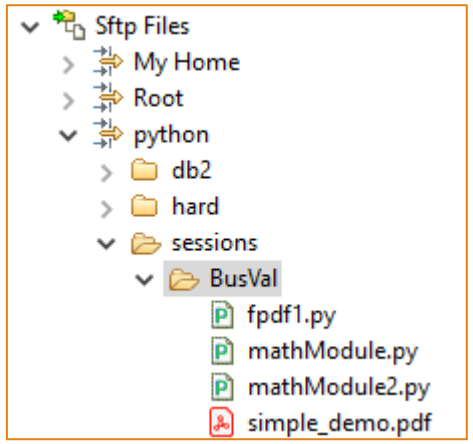
Command: **pip3 install fpdf**

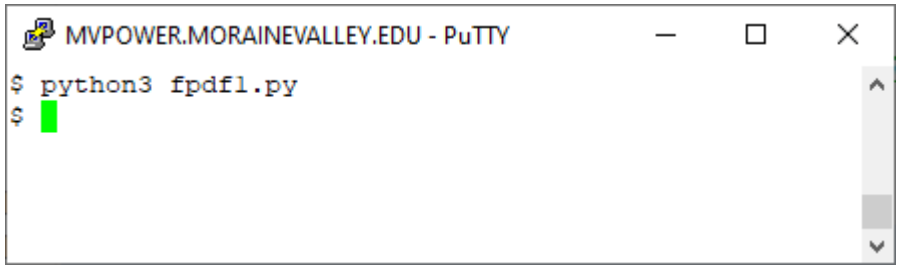


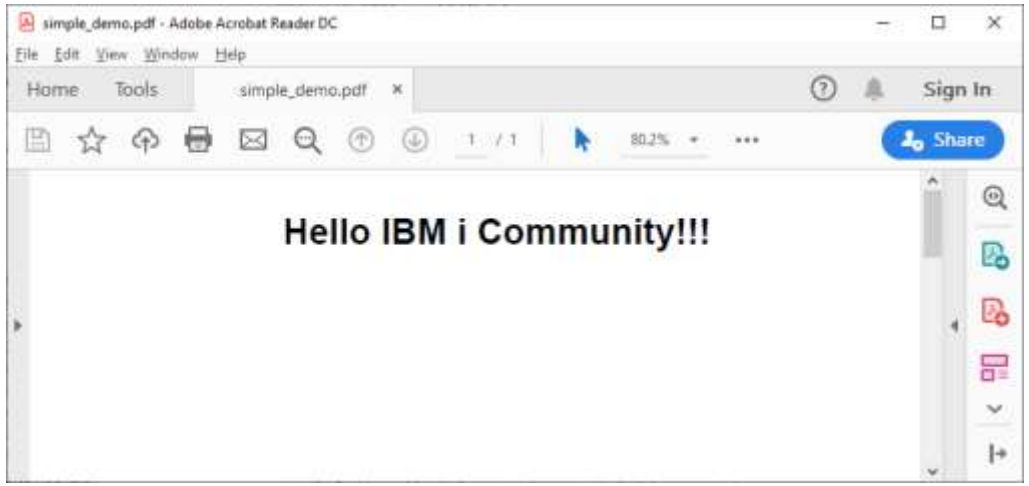
https://pyfpdf.readthedocs.io/en/latest/

Code & output

```
from fpdf import FPDF

pdf = FPDF()
pdf.add_page()
pdf.set_font("Arial", size=24)

pdf.cell(200, 10, txt="Hello IBM i Community!!!",
ln=1, align="C")

pdf.output("simple_demo.pdf")
```



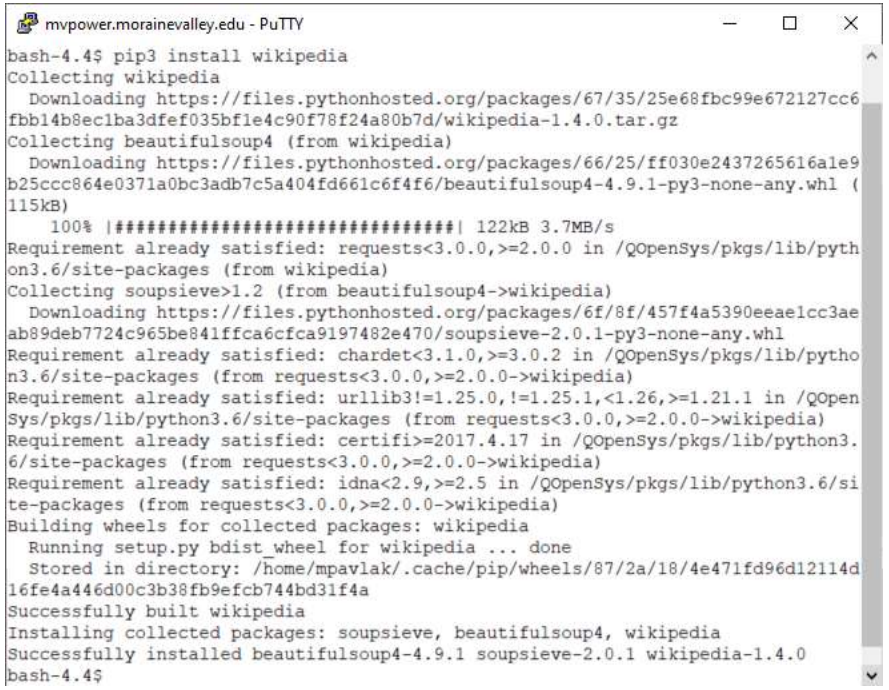




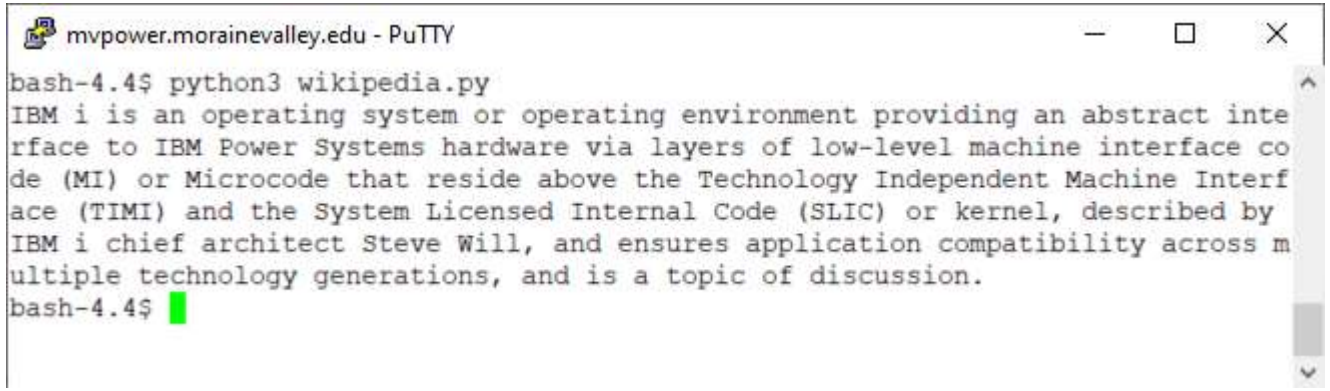
Wikipedia

Access from a command line…

- pip3 install wikipedia

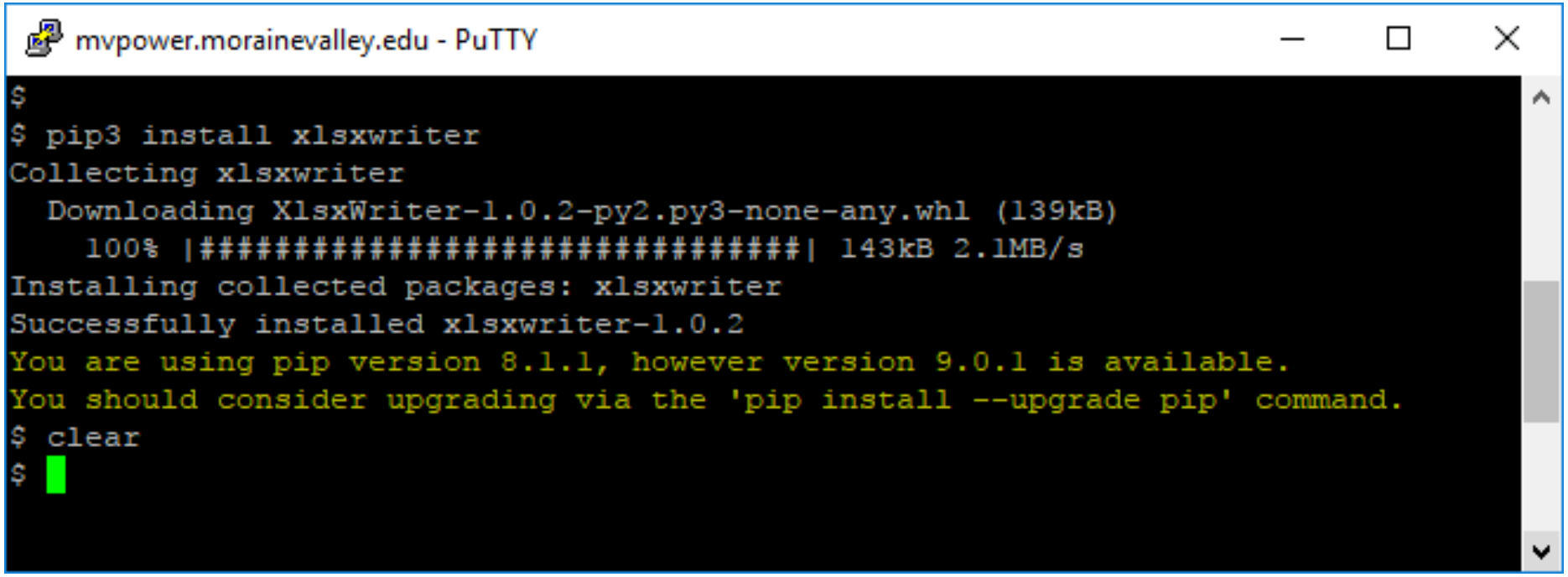



```
import wikipedia

searchSubject = "IBM i"
response = wikipedia.page(searchSubject)
print(response.summary)
```



https://pypi.org/project/wikipedia/

Spreadsheet? Install xlsxwriter



Got doc?

http://xlsxwriter.readthedocs.io/chart.html

The Chart Class

The `Chart` module is a base class for modules that implement charts in XlsxWriter. The information in this section is applicable to all of the available chart subclasses, such as Area, Bar, Column, Doughnut, Line, Pie, Scatter, Stock and Radar.

A chart object is created via the Workbook `add_chart()` method where the chart type is specified:

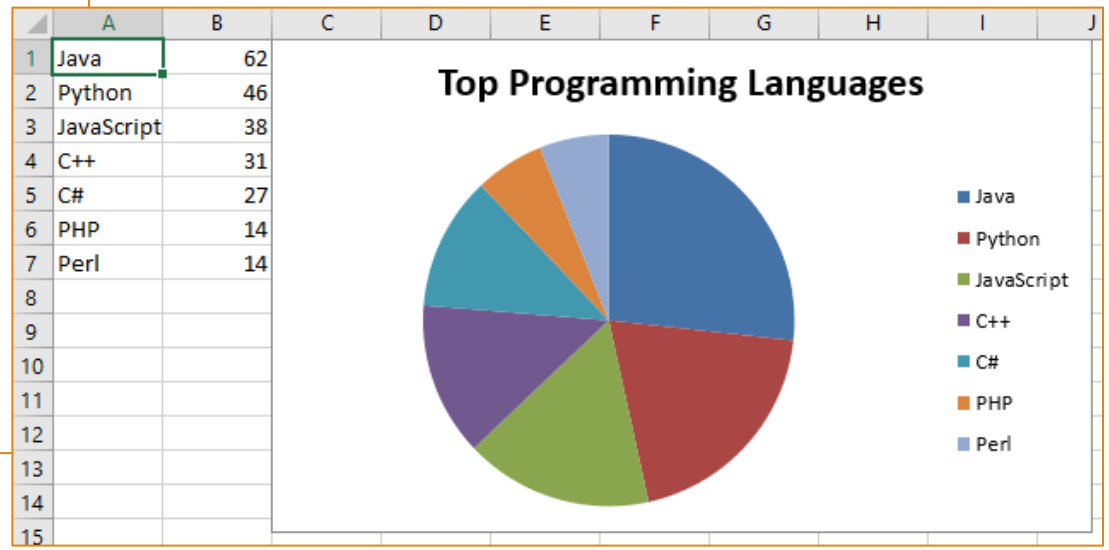
```
chart = workbook.add_chart({'type': 'column'})
```

It is then inserted into a worksheet as an embedded chart using the `insert_chart()` Worksheet method:

```
worksheet.insert_chart('A7', chart)
```

Now let's do something cool

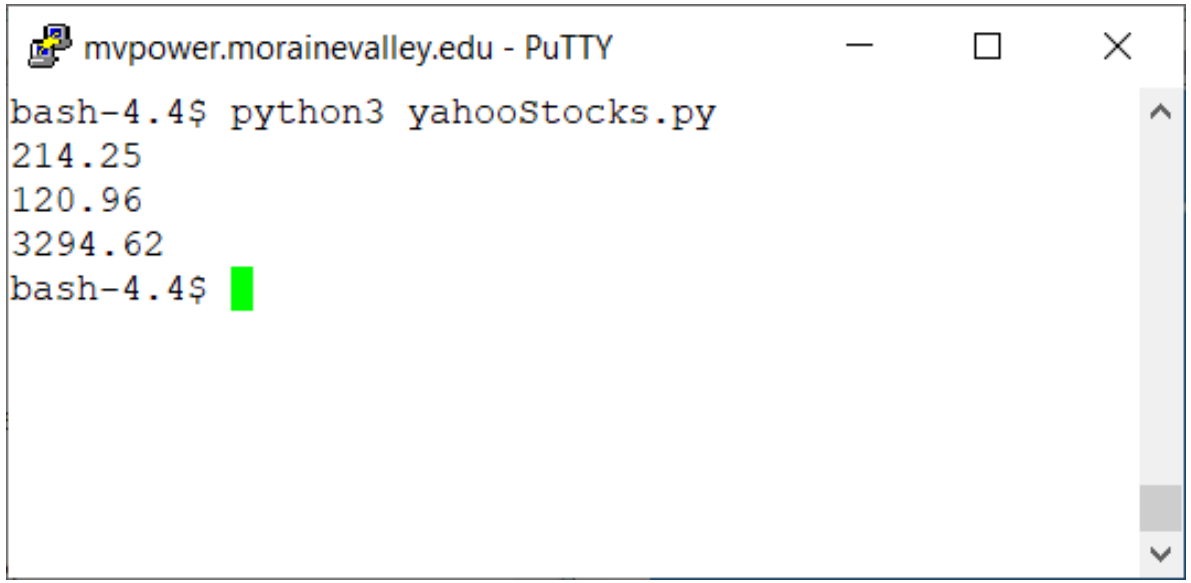
```
from xlsxwriter import Workbook


with Workbook('test.xlsx') as workbook:
    ws = workbook.add_worksheet()
    ws.write_column('A1', ['Java', 'Python', 'JavaScript',
                           'C++', 'C#', 'PHP', 'Perl'])
    ws.write_column('B1', [62, 46, 38, 31, 27, 14, 14])
    #chart = workbook.add_chart({'type': 'line'})
    chart = workbook.add_chart({'type': 'pie'})
    chart.set_title({'name': 'Top Programming Languages'})
    chart.add_series({'categories': '=Sheet1!$A$1:$A$7',
        'values': '=Sheet1!$B$1:$B$7'})

    ws.insert_chart('C1', chart)
```

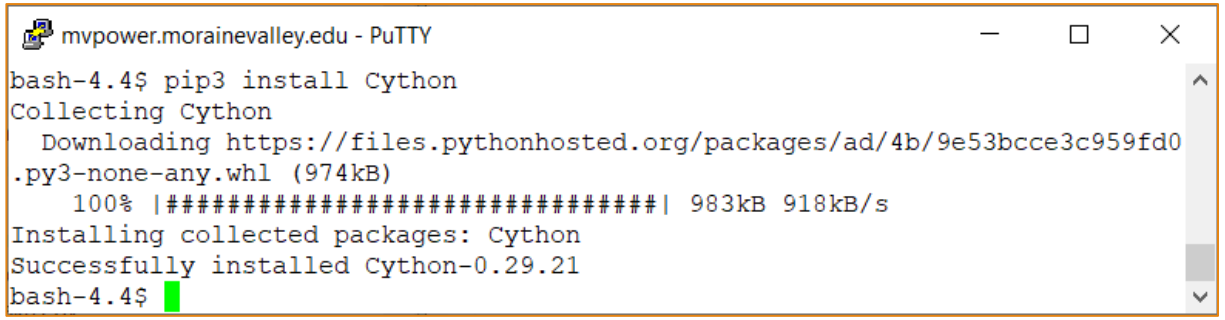


Stock prices

https://pypi.org/project/yahoo-finance/

```
#yFinance...
import yfinance as yf

#List of stocks

stocks = ['msft', 'aapl', 'amzn']

stocksParm = ' '.join(stocks)

tickers = yf.Tickers(stocksParm)


print(tickers.tickers.MSFT.history("1d")["Close"].values[0])

print(tickers.tickers.AAPL.history("1d")["Close"].values[0])

print(tickers.tickers.AMZN.history("1d")["Close"].values[0])
```



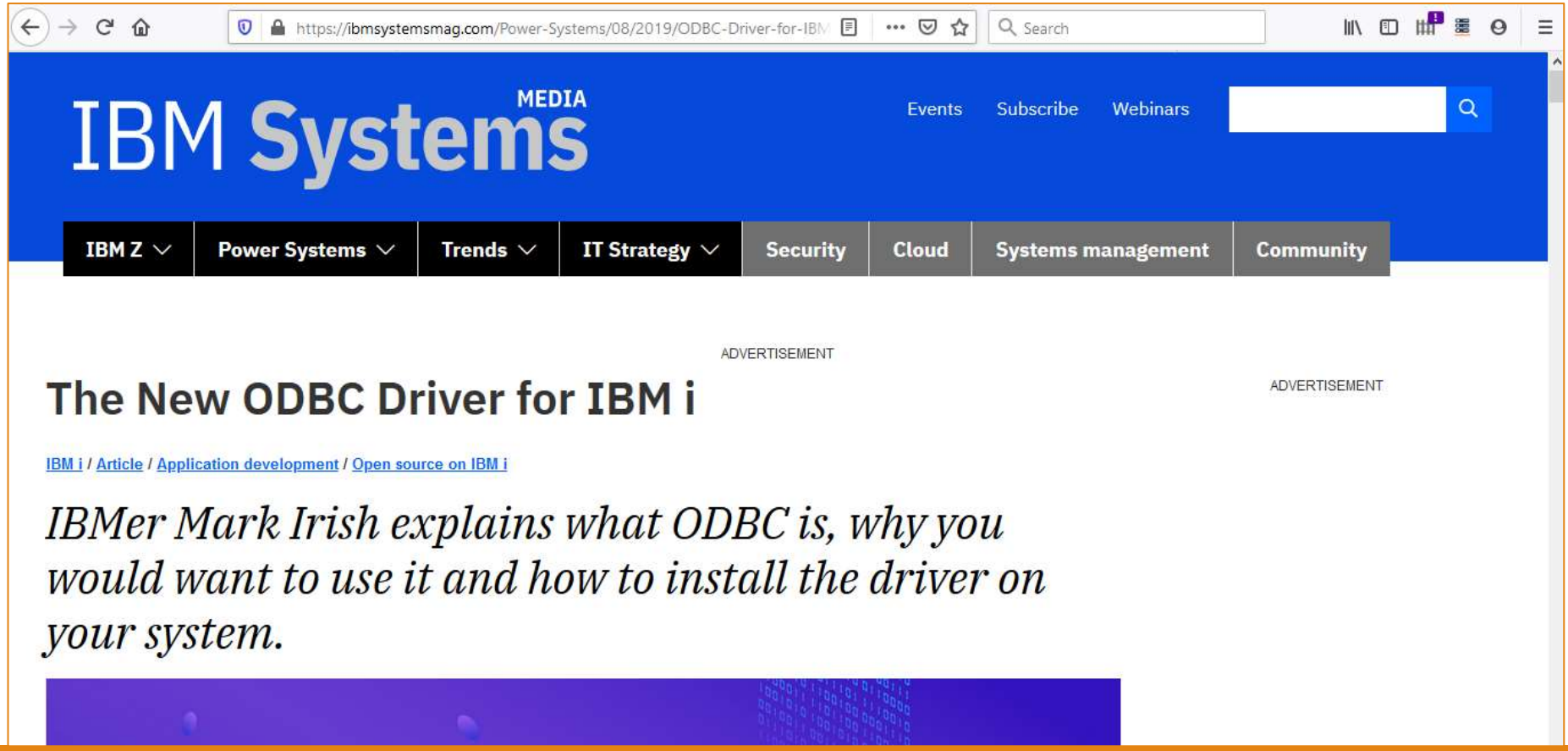


PYODBC

DATABASE

ODBC is where it’s at…

Read the article by Mark Irish https://ibmsystemsmag.com/Power-Systems/08/2019/ODBC-Driver-for-IBM-i



ODBC Needs 3 pieces to work

ODBC Driver

ODBC Driver Manager

Application specific connectors (adapter)

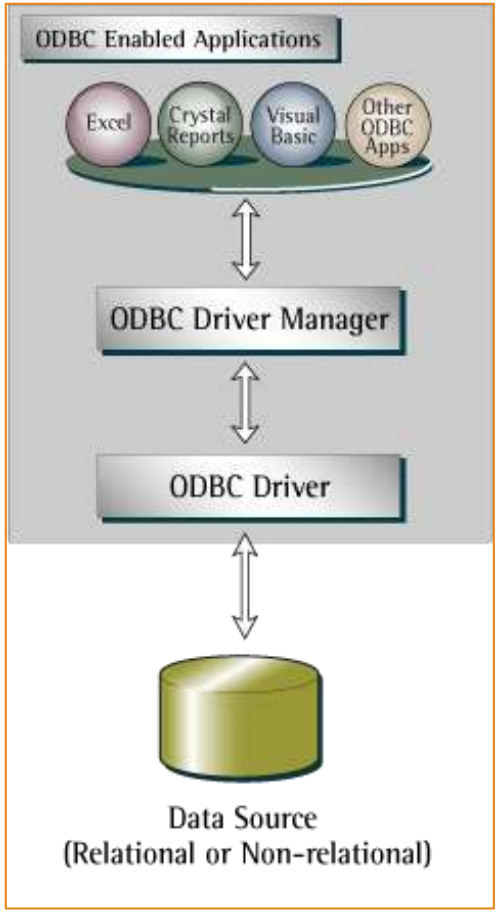
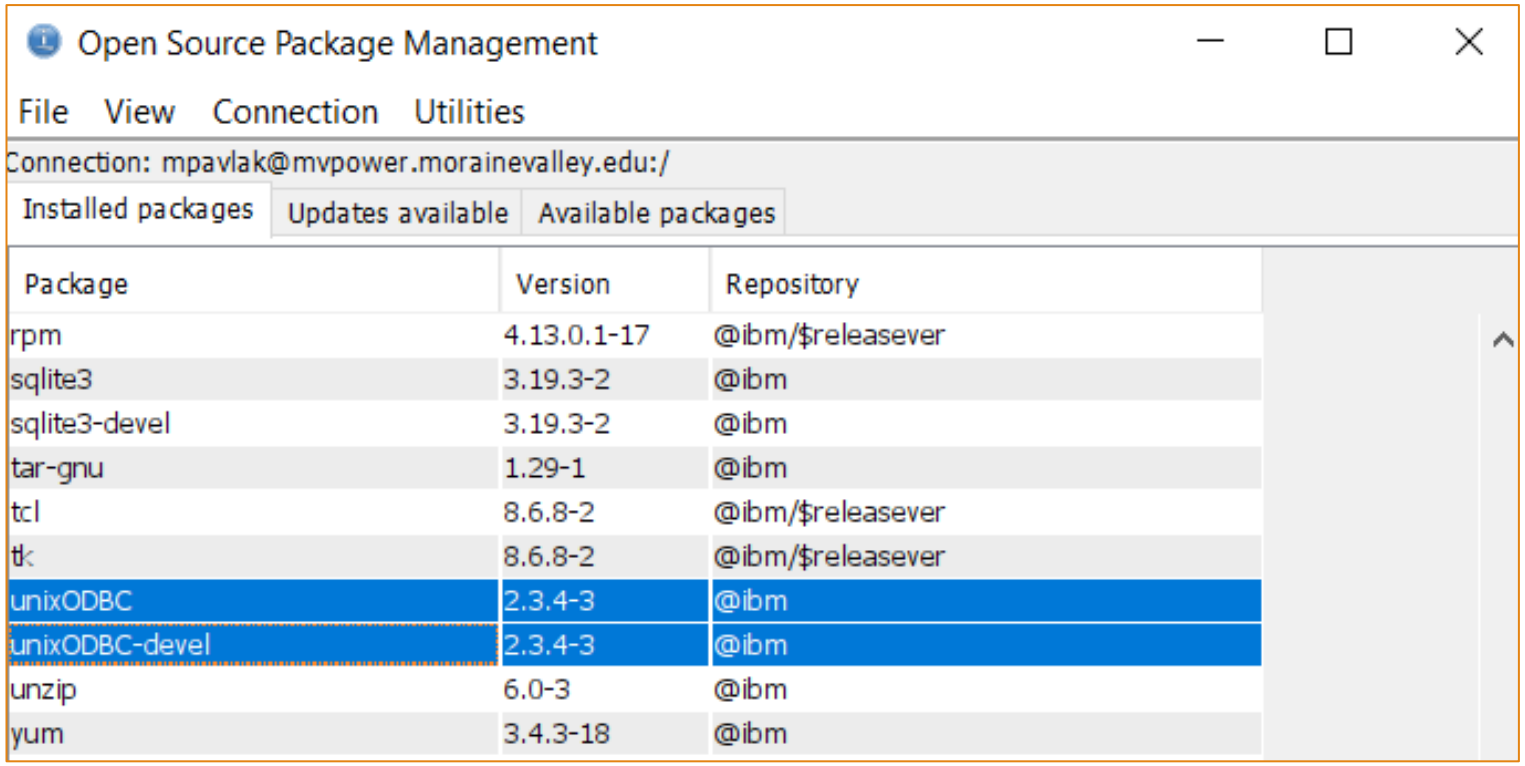


Image: Wikimedia Commons

ODBC: Driver Manager

Step 1

- Command line: **yum install unixODBC unixODBC-devel**
- GUI:



ODBC Driver is on ACS site

Step 2:

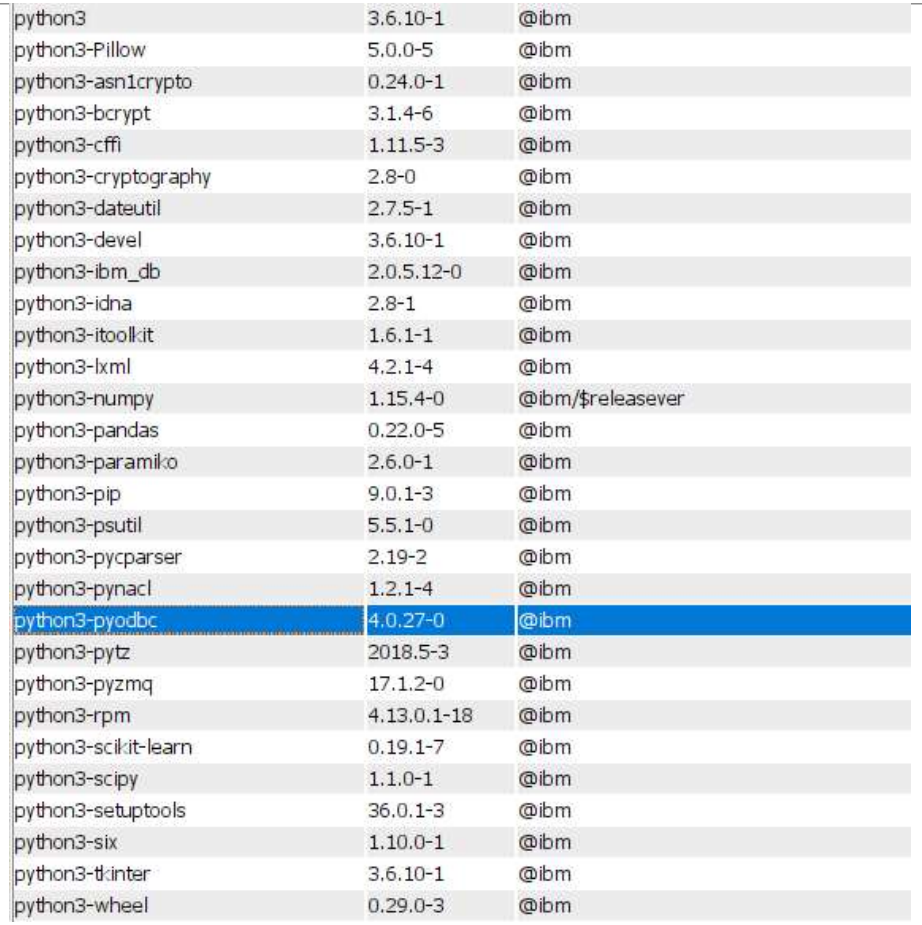
- Dowload
- Install

Description	Filename	Size	Action
IBM i Access Client Solutions	IBMiAccess_v1r1.zip	134107575 B	Download
Readme file for 5733-XJ1	Readme.txt	5 KB	Download
Quick Start Guide	QuickStartGuide_en.html	10 KB	Download
Getting Started	GettingStarted_en.html	119 KB	Download
ACS EHLLAPI - Version 8.20.104	acshllapi.exe	986862	Download
ACS Windows App Pkg English (64bit)	IBMiAccess_v1r1_WindowsAP_English.zip	53311376 B	Download
ACS Linux App Pkg	IBMiAccess_v1r1_LinuxAP.zip	20797233 B	Download
ACS PASE App Pkg	IBMiAccess_v1r1_PASE_AP.zip	8577229 B	Download

PyODBC: Application Adapter for Python

Step 3:

- Use IBM Open Source package Manager in ACS
  - Highlight
  - Click install
  - Type “y”
- Driver is strategic to python data access.
- Indirectly supported



python3	3.6.10-1	@ibm
python3-Pillow	5.0.0-5	@ibm
python3-asn1crypto	0.24.0-1	@ibm
python3-bcrypt	3.1.4-6	@ibm
python3-cffi	1.11.5-3	@ibm
python3-cryptography	2.8-0	@ibm
python3-dateutil	2.7.5-1	@ibm
python3-devel	3.6.10-1	@ibm
python3-ibm_db	2.0.5.12-0	@ibm
python3-idna	2.8-1	@ibm
python3-itoolkit	1.6.1-1	@ibm
python3-lxml	4.2.1-4	@ibm
python3-numpy	1.15.4-0	@ibm/$releasever
python3-pandas	0.22.0-5	@ibm
python3-paramiko	2.6.0-1	@ibm
python3-pip	9.0.1-3	@ibm
python3-psutil	5.5.1-0	@ibm
python3-pycparser	2.19-2	@ibm
python3-pynacl	1.2.1-4	@ibm
python3-pyodbc	4.0.27-0	@ibm
python3-pytz	2018.5-3	@ibm
python3-pyzmq	17.1.2-0	@ibm
python3-rpm	4.13.0.1-18	@ibm
python3-scikit-learn	0.19.1-7	@ibm
python3-scipy	1.1.0-1	@ibm
python3-setuptools	36.0.1-3	@ibm
python3-six	1.10.0-1	@ibm
python3-tkinter	3.6.10-1	@ibm
python3-wheel	0.29.0-3	@ibm

Steps for simple database Access

Import the class

Connect (with or without options)

Open the cursor

Set the SQL

Read

Simple script

```
import pyodbc

cnxn = pyodbc.connect('DSN=*LOCAL')

cursor = cnxn.cursor()

#Sample select query
cursor.execute('''SELECT CUST_ID, COMPANY, FIRSTNAME, LASTNAME, COUNTRY
    from zendphp7.sp_cust''')

row = cursor.fetchone()
while row:
    print(row)
    row = cursor.fetchone()
```

For DML type transactions

```
import pyodbc

cnxn = pyodbc.connect('DSN=*LOCAL;CommitMode=0')  #Turn off commitment control!

cursor = cnxn.cursor()

#Sample select query
cursor.execute("""update zendphp7.sp_cust
    set FIRSTNAME = 'Mike' where CUST_ID = 9841""")
```

Before:

CUST_ID	COMPANY	FIRSTNAME	LASTNAME	CIVIL	ADDRESS
9841	Neptune's Trident Supply	MIKE	Franks	2	PO Box 129

After:

CUST_ID	COMPANY	FIRSTNAME	LASTNAME	CIVIL	ADDRESS
9841	Neptune's Trident Supply	Mike	Franks	2	PO Box 129

FLASK

What is Flask?

Popular Micro web framework

Automates the boring “webby” stuff

Fast method to the web pages

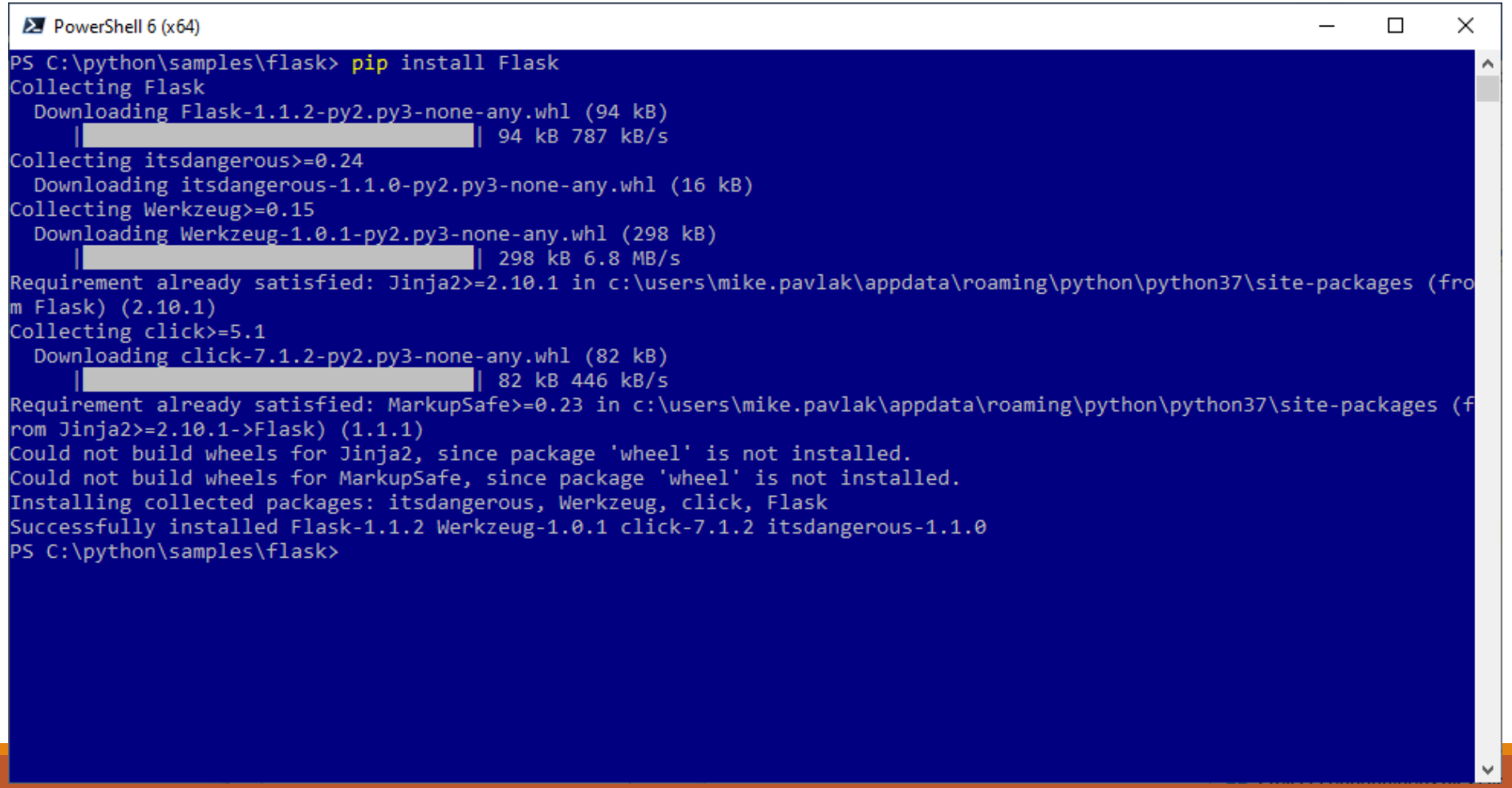
Productivity == $$$

Community and documentation

Very explicit code style, no large learning curve

https://pypi.org/project/Flask/

Installation on Windows

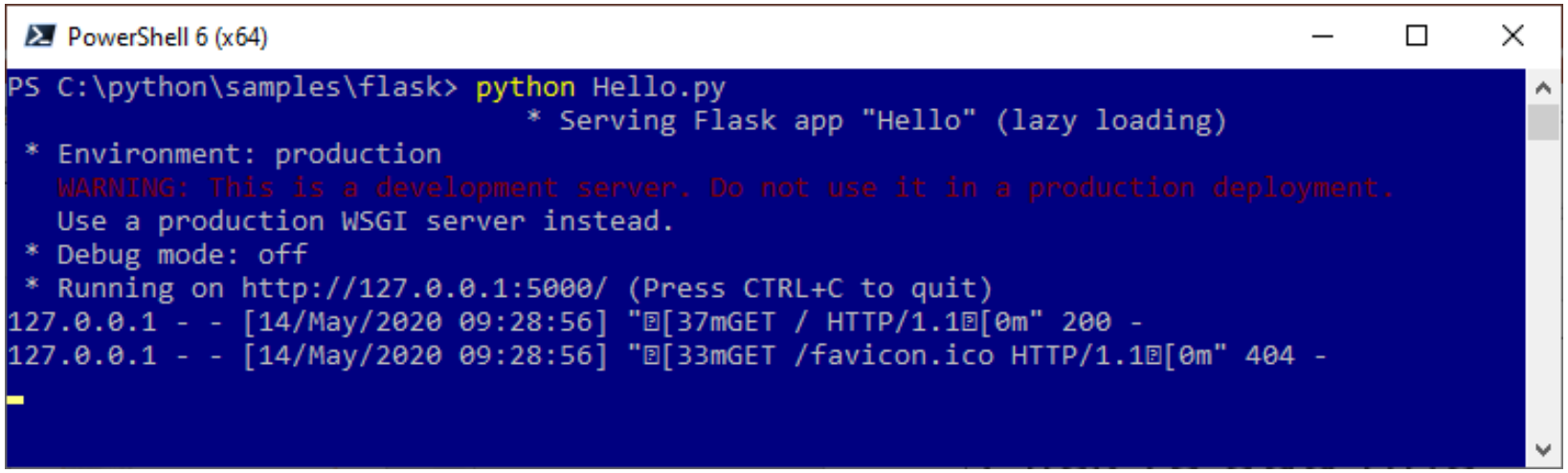


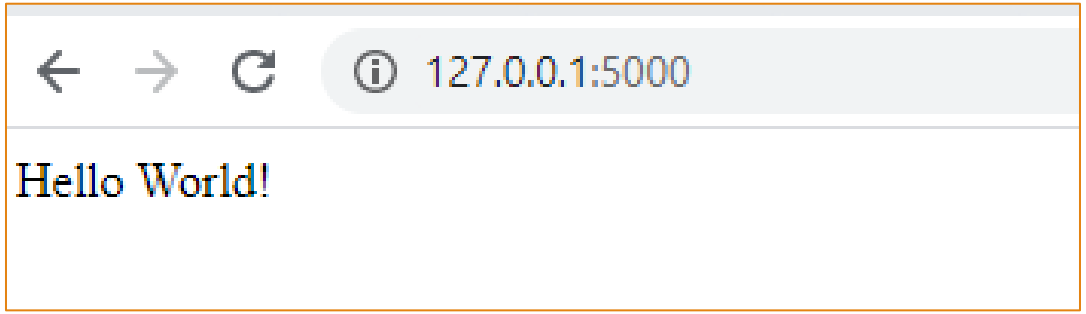
Installation on IBM i: pip3 install Flask

```
bash-4.4$ pip3 install Flask
Collecting Flask
  Using cached https://files.pythonhosted.org/packages/f2/28/2a03252dfb9ebf377f40fba6a7841b47083260bf8bd8e737b0c6952df83f/Flask-1.1.2-py2.py3-none-any.whl
Collecting Werkzeug>=0.15 (from Flask)
  Using cached https://files.pythonhosted.org/packages/cc/94/5f7079a0e00bd6863ef8f1da638721e9da21e5bacee597595b318f71d62e/Werkzeug-1.0.1-py2.py3-none-any.whl
Collecting click>=5.1 (from Flask)
  Using cached https://files.pythonhosted.org/packages/d2/3d/fa76db83bf75c4f8d338c2fd15c8d33fdd7ad23a9b5e57eb6c5de26b430e/click-7.1.2-py2.py3-none-any.whl
Collecting itsdangerous>=0.24 (from Flask)
  Using cached https://files.pythonhosted.org/packages/76/ae/44b03b253d6fade317f32c24d100b3b35c2239807046a4c953c7b89fa49e/itsdangerous-1.1.0-py2.py3-none-any.whl
Collecting Jinja2>=2.10.1 (from Flask)
  Using cached https://files.pythonhosted.org/packages/30/9e/f663a2aa66a09d838042ae1a2c5659828bb9b41ea3a6efa20a20fd92b121/Jinja2-2.11.2-py2.py3-none-any.whl
Collecting MarkupSafe>=0.23 (from Jinja2>=2.10.1->Flask)
  Using cached https://files.pythonhosted.org/packages/b9/2e/64db92e53b86efccfaea71321f597fa2e1b2bd3853d8ce658568f7a13094/MarkupSafe-1.1.1.tar.gz
Building wheels for collected packages: MarkupSafe
  Running setup.py bdist_wheel for MarkupSafe ... done
  Stored in directory: /home/mpavlak/.cache/pip/wheels/f2/aa/04/0edf07a1b8a5f5f1aed7580fffb69ce8972edc16a505916a77
Successfully built MarkupSafe
Installing collected packages: Werkzeug, click, itsdangerous, MarkupSafe, Jinja2, Flask
Successfully installed Flask-1.1.2 Jinja2-2.11.2 MarkupSafe-1.1.1 Werkzeug-1.0.1 click-7.1.2 itsdangerous-1.1.0
bash-4.4$
```

Hello on Windows

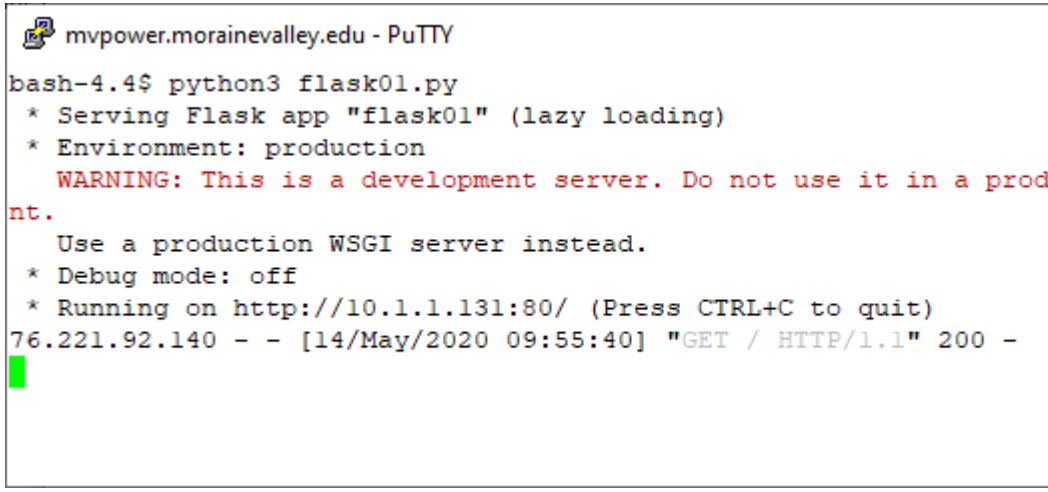
```python
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run()
```

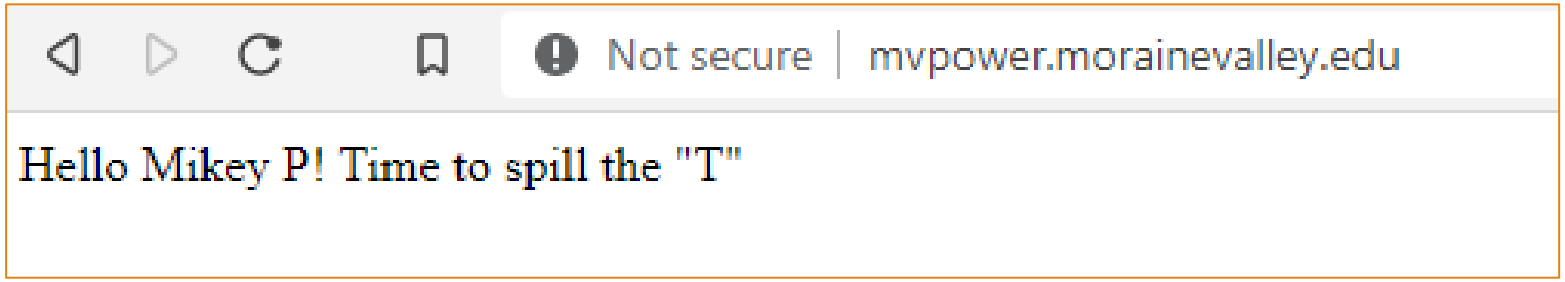




Hello on IBM i

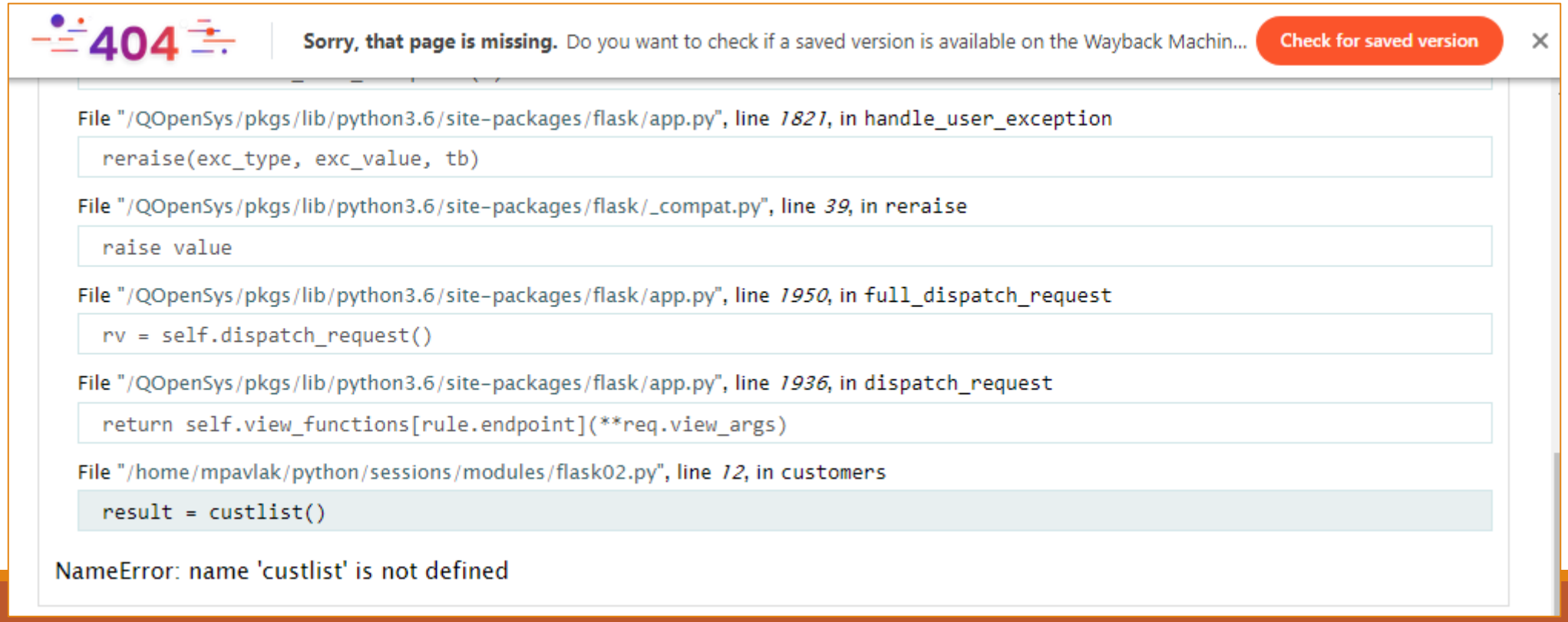
```
from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return 'Hello Mikey P!  Time to spill the "T"'

if __name__ == "__main__":
    app.run(host="10.1.1.131", port=int("80"))
```





Consider debug…

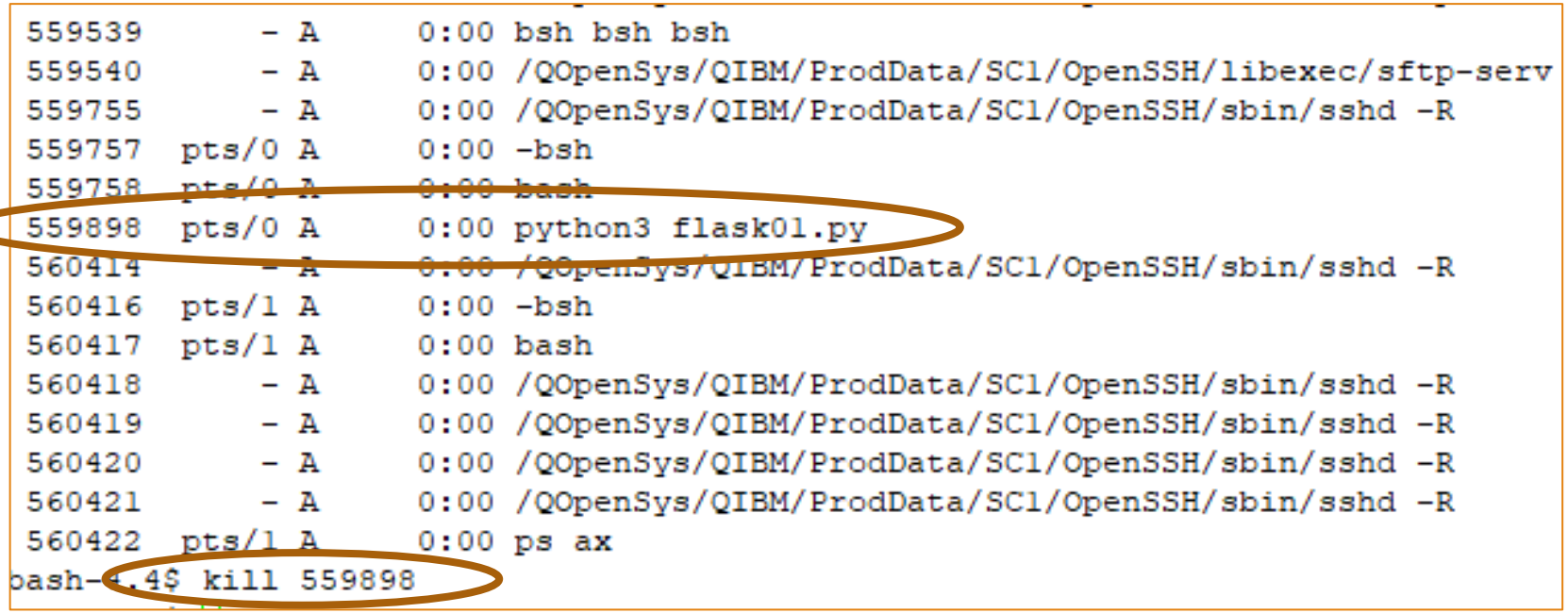
```
app.run(host="10.1.1.131", port=int("80"), debug=True)
```



Troubleshooting note…

Process runs in PASE

If you lose shell connection, you may need to go into PASE to “kill” the PID.



```python
from flask import Flask
import pyodbcfunc as pof

app = Flask(__name__)

@app.route("/")
def hello():
    return 'Hello Mikey P!  Time to spill the "T"'

@app.route("/customers")
def customers():

    result = list(pof.custlist())
    response = '''<table border="2"><tr><th>Number</th><th>Company</th>
        <th>First Name</th><th>Last
        Name</th><th>Country</th></tr>\n'''
    for row in result:
        response += '<tr>'
        for col in row:
            data = str(col)
            response += '<td>' + data.strip() + '</td>'
        response += '</tr>\n'
    response += '</table>'

    return response

if __name__ == "__main__":
    app.run(host="10.1.1.131", port=int("80"), debug=True)
```

```
bash-4.4$ python3 flask02.py
 * Serving Flask app "flask02" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://10.1.1.131:80/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 276-042-664
```

Not secure | mypower.morainevalley.edu/customers

Number	Company	First Name	Last Name	Country
1221	Kauai Dive Shoppe	LINA	Norman	US
1231	Unisco	George	Weathers	Bahamas
1351	Sight Diver	Phyllis	Spooner	Cyprus
1354	Cayman Divers World Unlimited	Joe	Bailey	British West Indies
1356	Tom Sawyer Diving Centre	Chris	Thomas	US Virgin Islands
1380	Blue Jack Aqua Center	Ernest	Barratt	US
1384	VIP Divers Club	Russell	Christopher	US Virgin Islands
1510	Ocean Paradise	Paul	Gardner	US
1513	Fantastique Aquatica	Susan	Wong	Columbia
1551	Marmot Divers Club	Joyce	Marsh	Canada
1560	The Depth Charge	Sam	Witherspoon	US
1563	Blue Sports	Theresa	Kunec	US
1624	Makai SCUBA Club	Donna	Sisus	US
1645	Action Club	Michael	Spurling	US
1651	Jamaica SCUBA Centre	Barbara	Harvey	West Indies
1680	Island Finders	Desmond	Ortega	US
1984	Adventure Undersea	Gloria	Gonzales	Belize
2118	Blue Sports Club	Harry	Bathbone	US
2135	Frank's Divers Supply	Lloyd	Fellows	US
2156	Davy Jones' Locker	Tanya	Wagner	Canada
2163	SCUBA Heaven	Robert	Michelind	Bahamas
2165	Shangri-La Sports Center	Frank	Panagua	Bahamas
2315	Divers of Corfu, Inc.	Charles	Lopez	Greece
2354	Kirk Enterprises	Rudolph	Claus	US

Some formidable candidates

Django

Pyramid

TurboGears

Dash

Microframeworks
- Flask
- Bottle
- CherryPy

Asynchronous Frameworks
- Sanic
- Tornado

Where to get more info and help?

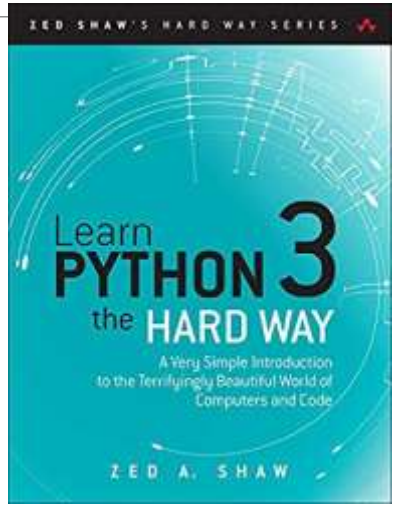
Club Seiden

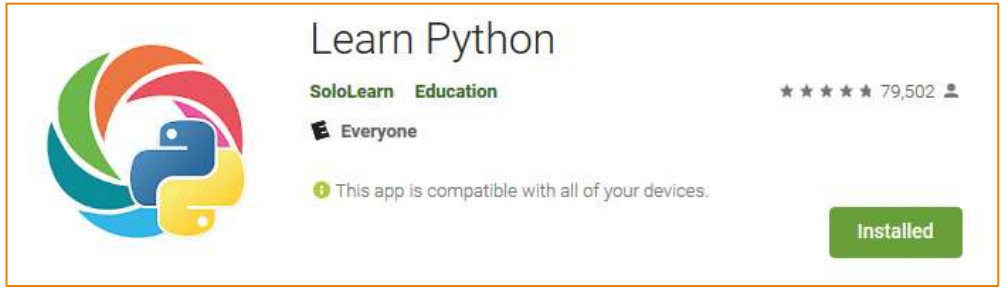
Learn Python 3 The Hard Way

SoloLearn: Learn Python (Android & Apple App)

https://ibmioss.ryver.com (send me a note for invite…)

Pythonweekly.com weekly newsletter





Python Modules

MISC

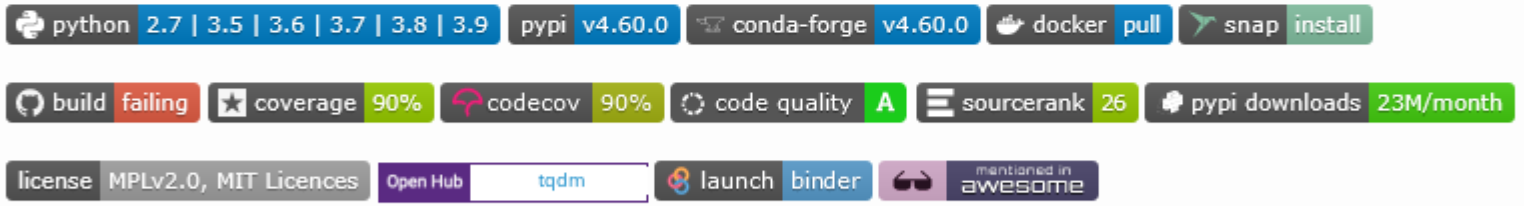
https://pypi.org/project/tqdm/

TQDM

Progress bar

Project description

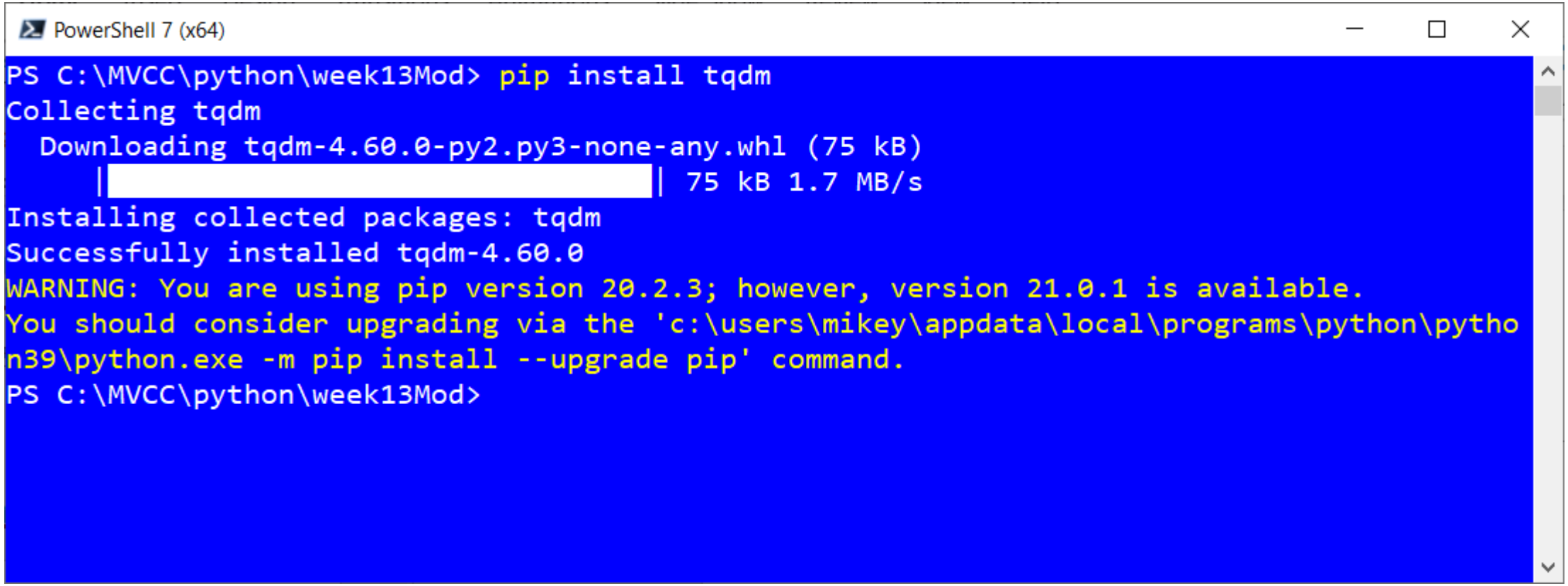
tqdm



`tqdm` derives from the Arabic word *taqaddum* (تقدّم) which can mean "progress," and is an abbreviation for "I love you so much" in Spanish (*te quiero demasiado*).

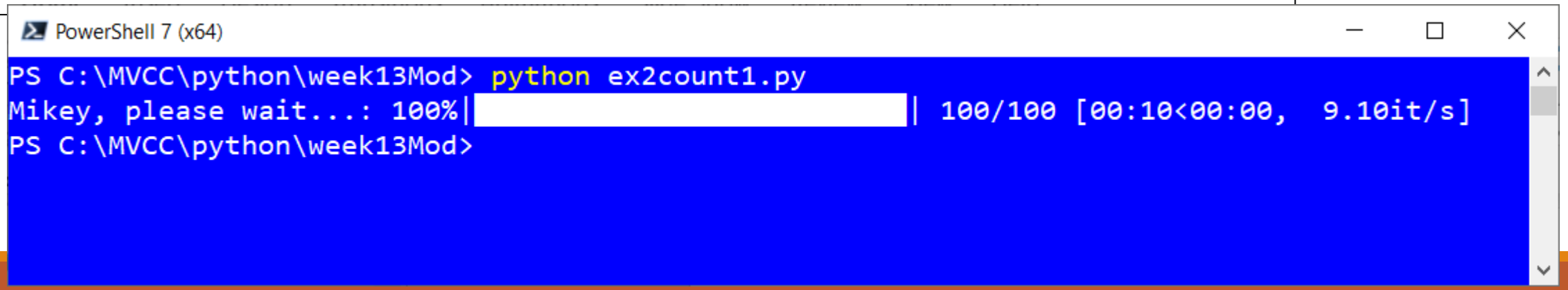
Instantly make your loops show a smart progress meter - just wrap any iterable with `tqdm(iterable)`, and you're done!

Progress bar



Live demo is best

```
# Countdown

from tqdm import tqdm
from time import sleep

for i in tqdm(range(0,100), desc = "Mikey, please wait..."):
    sleep(.1) #Long process here...
```



Needed to load data

```
for sequence in range(1,100):
    count = 0
    print("\n")
    print(datetime.datetime.now())
    for claim in range(1826, 1930):
        urlnew = url + str(claim)
        res = requests.get(urlnew)
        count +=1

    print("\nnumber of claims processed for claimcost is:",count)
    print(datetime.datetime.now())

    count = 0
    for claim in range(1826, 1930):
        urlnew = url2 + str(claim)
        res = requests.get(urlnew)
        count +=1

    print("\nnumber of claims processed for claims is:",count)
    print(datetime.datetime.now())
```

Site connectivity checker

```
import urllib.request

def connect(host) :
    try:
        urllib.request.urlopen(host)
        return True
    except:
        return False

print()
host = 'http://google.com'
print(('Found ' + host) if connect(host) else ('could not find' + host))

print()
host = 'http://googleapilove.com'
print(('Found ' + host) if connect(host) else ('could not find' + host))

print()
host = 'http://ibm.com'
print(('Found ' + host) if connect(host) else ('could not find' + host))
```

Summary

Python is the CL of Open Source

Useful beyond infrastructure

Virtually ANY root file system task

Supported and strategic

Full data access

Framework add productivity

Python Modules for Profitability

Questions?


MikePavlak@gmail.com


@MikeyPEI

IBM CHAMPION

Profound Logic