UQ Computing Society

Python Beyond CSSE1001

Libraries for real-world Python projects

UQ Computing Society - 2017

Who Are We?

Mitchell Grice

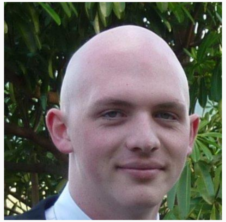
UQCS Committee
6th Year Software Engineering / Business
2 Years Professional Python

Lewis Bobbermen

UQCS Member
2nd Year Software Engineering
1 Year Professional Python

Who Are We?

Tom Manderson



UQCS President
4th Year Software Engineering / Math
2 Years Professional Python

Nicholas Lambourne

UQCS Member
2nd Year Software Engineering
BCom/BSc ‘15
1 Year Professional Python

What is UQCS?

UQ Computing Society

The University of Queensland Computing Society

The only society that focuses on

- Programming
- Computers & Technology
- Arguments about tabs & spaces

What is UQCS?

UQ Computing Society

Events every week

- Talks
- Programming Sessions

And also..

- Huge online chat (Slack)
- Competitions, Study Help, Mentors, Pizza

Why Python?

- Simple
- Powerful enough
- Taught at UQ to every IT / SwEngg student
- Lots of great libraries
  - Mature
  - Documented
  - Good communities

JSON

Data Serialisation Format

JSON

What it does and why you should care

Java**S**cript **O**bject **N**otation

The JSON library can:

- Turn Data into Text
- Turn Text into Data

This JSON string is:

- Cross language
- Cross platform
- Reasonably small
- Perfect for Python
- Widely used standard

JSON

Code Example

JSON

Find out more

Install:
None, it’s in the standard Python libraries

Docs:
https://docs.python.org/3/library/json.html

Requests

Makes HTTP Requests

Requests

What it does and why you should care

HTTP For Humans

Lets you make HTTP requests without building the whole request manually

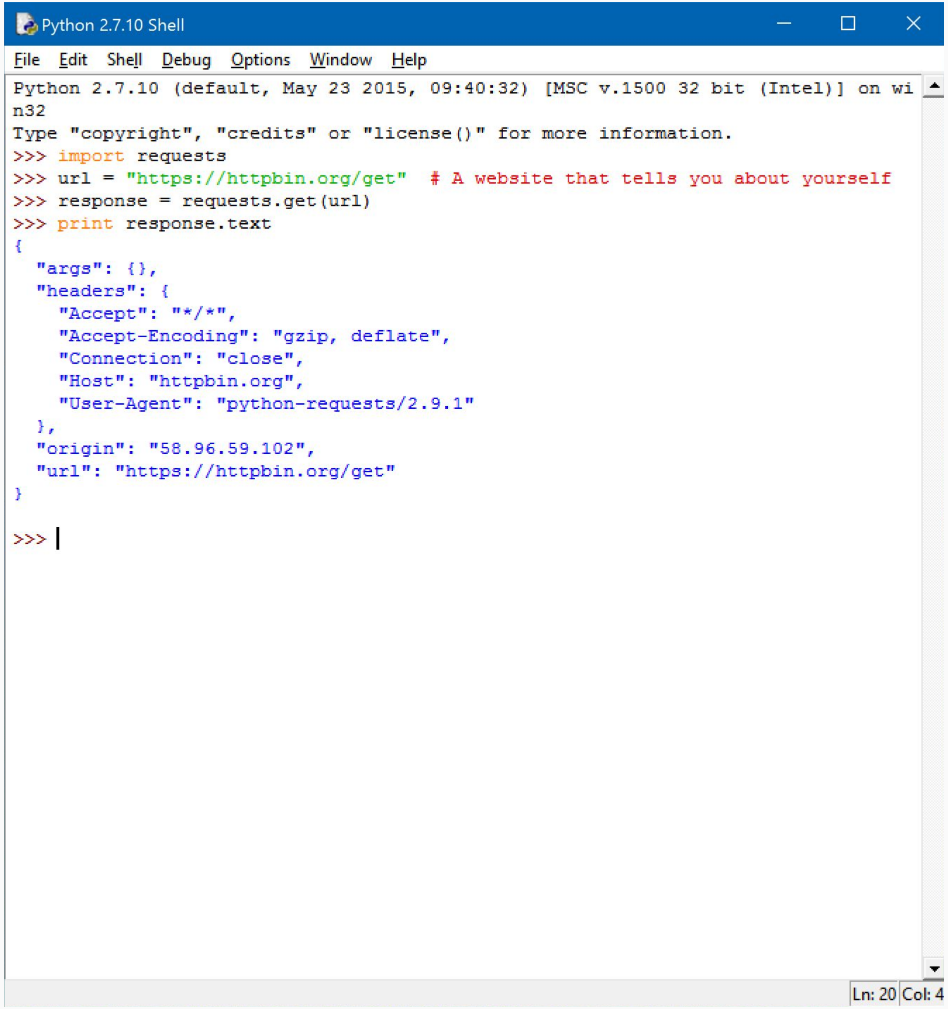
This is awesome because:

- Use APIs in your software to get data
- Send data to other systems

Requests

Code Example

Requests
http for humans



```
Python 2.7.10 (default, May 23 2015, 09:40:32) [MSC v.1500 32 bit (Intel)] on wi
n32
Type "copyright", "credits" or "license()" for more information.
>>> import requests
>>> url = "https://httpbin.org/get"  # A website that tells you about yourself
>>> response = requests.get(url)
>>> print response.text
{
  "args": {},
  "headers": {
    "Accept": "*/*",
    "Accept-Encoding": "gzip, deflate",
    "Connection": "close",
    "Host": "httpbin.org",
    "User-Agent": "python-requests/2.9.1"
  },
  "origin": "58.96.59.102",
  "url": "https://httpbin.org/get"
}

>>> 
```

Requests

Find out more

Install:

```
pip install requests
```

Docs:

http://docs.python-requests.org

Serves a Website

Flask

What it does and why you should care

- Create a website in Python
- Simple to start
  - Write functions which return strings
  - Wrap functions in route decorators

Flask

Code Example

Flask

Find out more

Docs:
http://flask.pocoo.org/docs

Numpy

Numbers & Maths

Numpy

What it does and why you should care

- Mathematical equations
  - Useful
  - Powerful
  - Fast
- Data-types that other libraries use

Numpy

Code Example

Numpy

Find out more

Docs:
https://docs.scipy.org/doc/numpy-dev/

Pandas

Bulk Data Manipulation

Pandas

What it does and why you should care

- Manipulate large amounts of data very quickly
- Very easy to use, and a consistent API
- Choose between mutating an object or returning a new one
- Great for Excel/CSV files

Pandas

Code Example

Pandas

Find out more

Docs:
http://pandas.pydata.org/pandas-docs/stable/

Plotly

Data Visualization

Plotly

What it does and why you should care

- Creates amazing graphs
- Easy to use
- Supported by multiple languages (JavaScript, R, MATLAB)

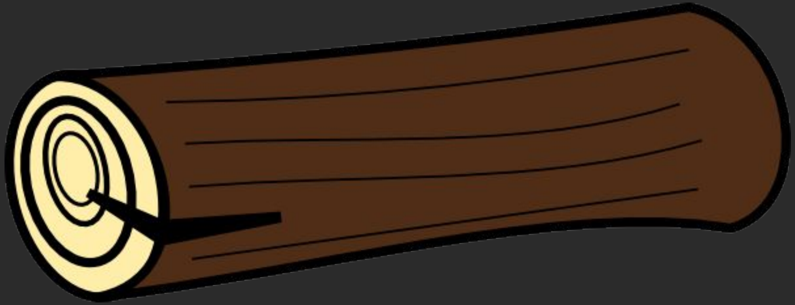
Plotly

Code Example

Plotly

Find out more

Docs:
https://plot.ly/python/

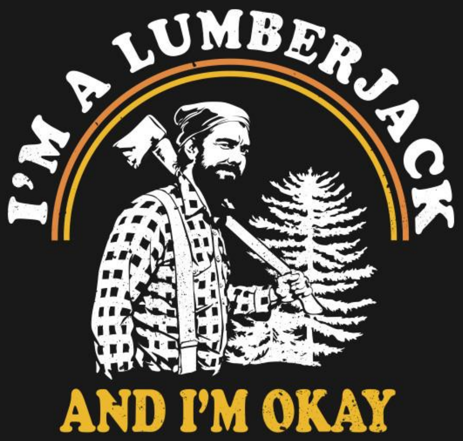


Logging

Better than Printing

Logging

What it does, and why you should care



What is logging?

- *The* way of recording what your program is doing
- Part of the Python standard library
- Unglamorous
- Completely and utterly necessary

When to use logging vs. print()?

print()	logging
● When prompting in a command line application.	● Literally every other occasion.

Basic Logging

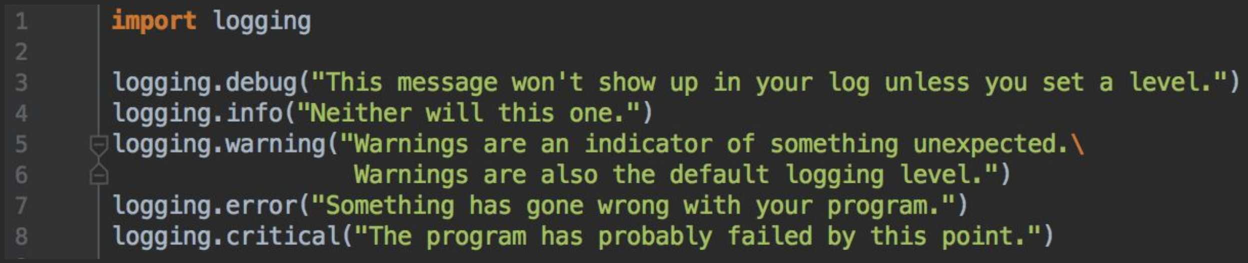
The idea of logging:

- Basic Logs = Basic Setup
- Only customise what you need
- Most functionality is provided

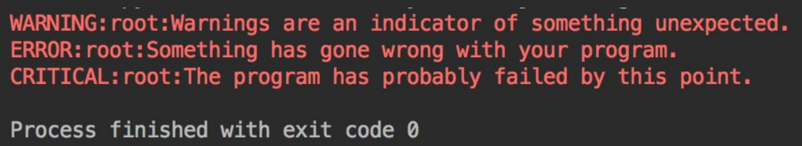
Common Gotcha:

- Default logging level is logging.WARNING

To the command line (no setup):

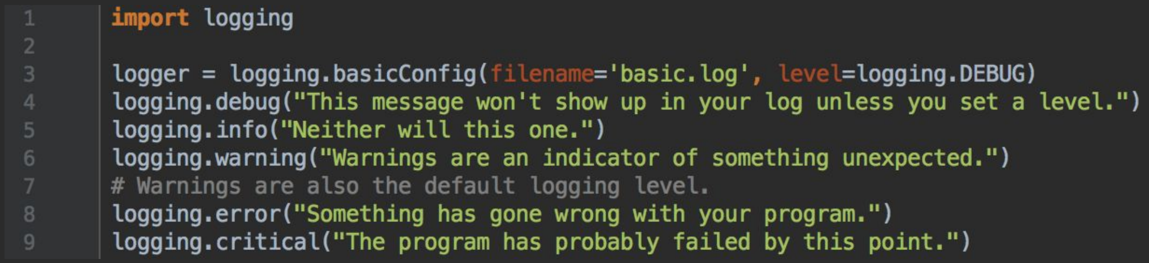


```
import logging

logging.debug("This message won't show up in your log unless you set a level.")
logging.info("Neither will this one.")
logging.warning("Warnings are an indicator of something unexpected.\
                 Warnings are also the default logging level.")
logging.error("Something has gone wrong with your program.")
logging.critical("The program has probably failed by this point.")
```

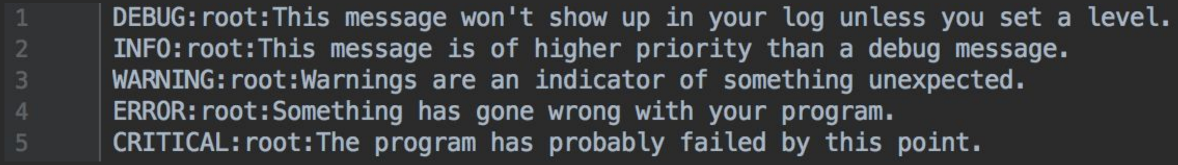


```
WARNING:root:Warnings are an indicator of something unexpected.
ERROR:root:Something has gone wrong with your program.
CRITICAL:root:The program has probably failed by this point.

Process finished with exit code 0
```

To a log file (one line setup):



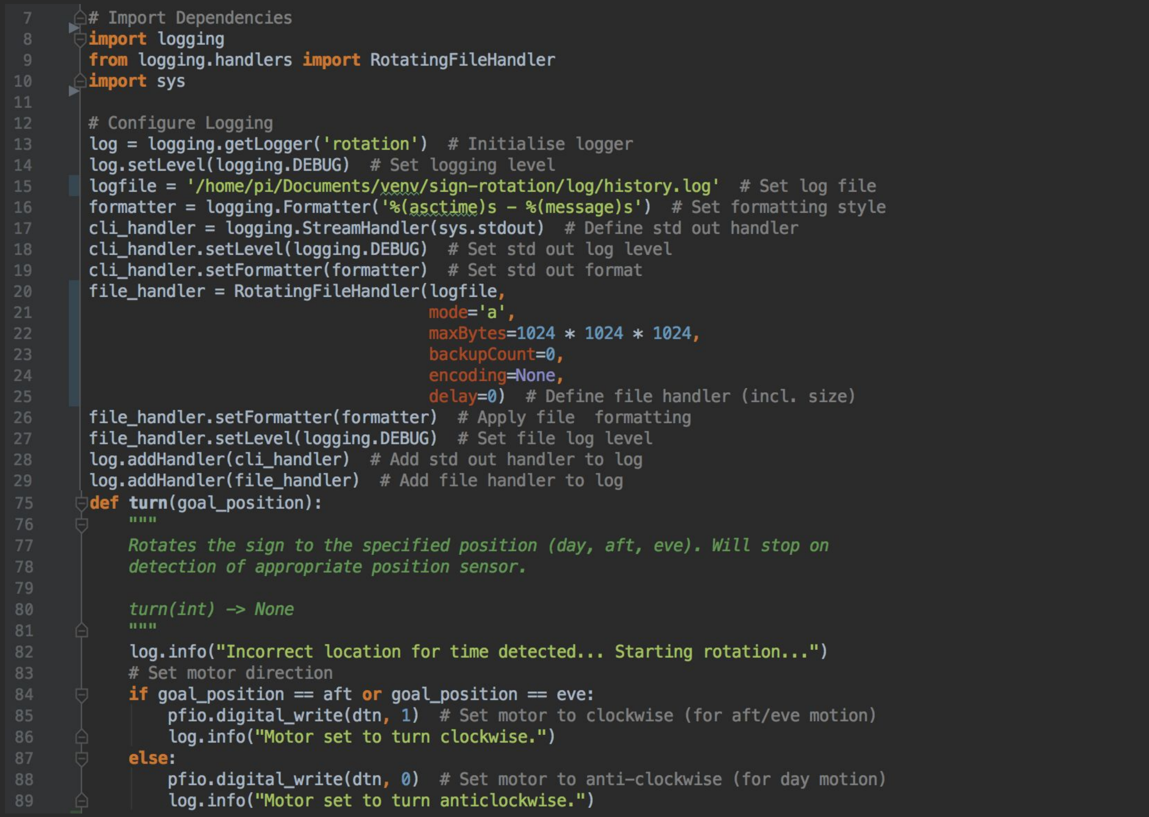
```
import logging

logger = logging.basicConfig(filename='basic.log', level=logging.DEBUG)
logging.debug("This message won't show up in your log unless you set a level.")
logging.info("Neither will this one.")
logging.warning("Warnings are an indicator of something unexpected.")
# Warnings are also the default logging level.
logging.error("Something has gone wrong with your program.")
logging.critical("The program has probably failed by this point.")
```



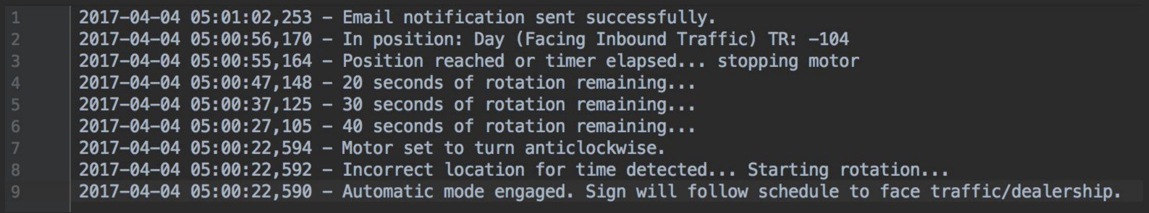
```
DEBUG:root:This message won't show up in your log unless you set a level.
INFO:root:This message is of higher priority than a debug message.
WARNING:root:Warnings are an indicator of something unexpected.
ERROR:root:Something has gone wrong with your program.
CRITICAL:root:The program has probably failed by this point.
```

More Advanced Logging

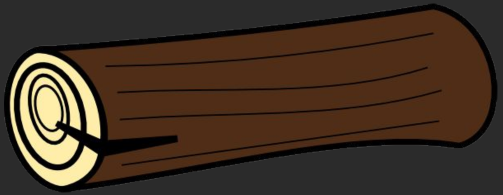
- Rotating File Handlers
- Multiple outputs (e.g. command line and log file).
- Control size of log files
- Custom output formats
- Automatic Backups



```python
# Import Dependencies
import logging
from logging.handlers import RotatingFileHandler
import sys

# Configure Logging
log = logging.getLogger('rotation')  # Initialise logger
log.setLevel(logging.DEBUG)  # Set logging level
logfile = '/home/pi/Documents/venv/sign-rotation/log/history.log'  # Set log file
formatter = logging.Formatter('%(asctime)s - %(message)s')  # Set formatting style
cli_handler = logging.StreamHandler(sys.stdout)  # Define std out handler
cli_handler.setLevel(logging.DEBUG)  # Set std out log level
cli_handler.setFormatter(formatter)  # Set std out format
file_handler = RotatingFileHandler(logfile,
                                   mode='a',
                                   maxBytes=1024 * 1024 * 1024,
                                   backupCount=0,
                                   encoding=None,
                                   delay=0)  # Define file handler (incl. size)
file_handler.setFormatter(formatter)  # Apply file  formatting
file_handler.setLevel(logging.DEBUG)  # Set file log level
log.addHandler(cli_handler)  # Add std out handler to log
log.addHandler(file_handler)  # Add file handler to log
def turn(goal_position):
    """
    Rotates the sign to the specified position (day, aft, eve). Will stop on
    detection of appropriate position sensor.

    turn(int) -> None
    """
    log.info("Incorrect location for time detected... Starting rotation...")
    # Set motor direction
    if goal_position == aft or goal_position == eve:
        pfio.digital_write(dtn, 1)  # Set motor to clockwise (for aft/eve motion)
        log.info("Motor set to turn clockwise.")
    else:
        pfio.digital_write(dtn, 0)  # Set motor to anti-clockwise (for day motion)
        log.info("Motor set to turn anticlockwise.")
```



```
2017-04-04 05:01:02,253 - Email notification sent successfully.
2017-04-04 05:00:56,170 - In position: Day (Facing Inbound Traffic) TR: -104
2017-04-04 05:00:55,164 - Position reached or timer elapsed... stopping motor
2017-04-04 05:00:47,148 - 20 seconds of rotation remaining...
2017-04-04 05:00:37,125 - 30 seconds of rotation remaining...
2017-04-04 05:00:27,105 - 40 seconds of rotation remaining...
2017-04-04 05:00:22,594 - Motor set to turn anticlockwise.
2017-04-04 05:00:22,592 - Incorrect location for time detected... Starting rotation...
2017-04-04 05:00:22,590 - Automatic mode engaged. Sign will follow schedule to face traffic/dealership.
```



Logging

More Resources & Tutorials

Logging Docs:
https://docs.python.org/3/library/logging.html

Useful logging tutorials:
Basic Logging Tutorial (Python Documentation)
https://docs.python.org/3/howto/logging.html#logging-basic-tutorial
In-Depth Logging Tutorial (Patrick's Software Blog)
http://www.patricksoftwareblog.com/python-logging-tutorial/
The Argument for Logging vs. Print (Hitchhiker's Guide to Python)
http://python-guide-pt-br.readthedocs.io/en/latest/writing/logging/

Other libraries to check out:
Slumber (Accessing REST APIs, built on Requests)
http://slumber.readthedocs.io/en/v0.6.0/
Pillow (Image Manipulation)
https://pillow.readthedocs.io/en/4.0.x/
Django (Web Framework, like Flask, but with everything built in)
https://docs.djangoproject.com/en/1.10/intro/tutorial01/

The best Python Editor:
PyCharm (Free while you're a student - does web dev, too)
https://www.jetbrains.com/student/

Jupyter Notebook

Like IDLE but better

Jupyter Notebook

What it does and why you should care

- Write Python like a notebook
- Keep input & output
- Sweet visualisations
- All in your browser

Jupyter Notebook

Code Example

Jupyter Notebook

Find out more

Docs:
http://jupyter-notebook.readthedocs.io

What next?

Join UQCS
Ask Questions

UQ Computing Society

What's that thing called?

Where do I get it?

What can I do with it?

How's it work?

I broke it, help me!

slack.uqcs.org.au