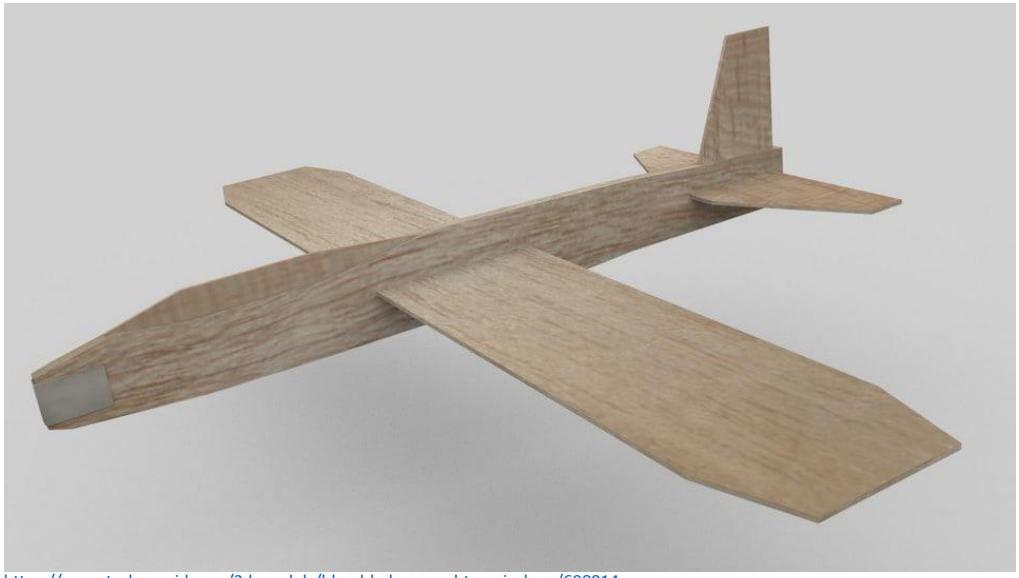
# Balsa

## Lightweight Python Logging



https://www.turbosquid.com/3d-models/blend-balsa-wood-toy-airplane/608014

James Abel

Aug 19, 2018

j@abel.co

www.abel.co @jamesabel www.github.com/jamesabel/balsa

# The `logging` module

- The `logging` module is awesome!
  - Handlers – stream (console), files, sockets, HTTP, custom, … many more!
  - Filters
  - Formatters
  - Hierarchal
  - Log Levels – debug, info, warning, error, critical
- `logging.getLogger(name)` provides the logger associated with `name`
  - Can directly access a logger from anywhere in your program with just the `name` string
- https://docs.python.org/library/logging.html

# `logging` levels

| Level | When it's used |
|---|---|
| DEBUG | Detailed information, typically of interest only when diagnosing problems. |
| INFO | Confirmation that things are working as expected. |
| WARNING | An indication that something unexpected happened, or indicative of some problem in the near future (e.g. 'disk space low'). The software is still working as expected. |
| ERROR | Due to a more serious problem, the software has not been able to perform some function. |
| CRITICAL | A serious error, indicating that the program itself may be unable to continue running. |

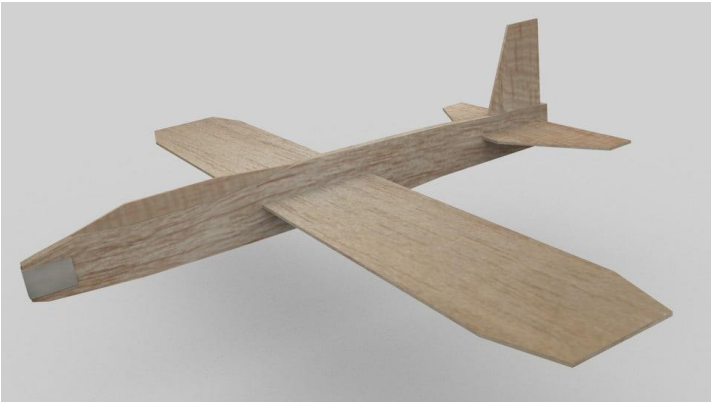
https://docs.python.org/howto/logging.html

# However, logging options and configurations can get rather involved for relatively simple apps

- Log message format
- Handers
- Where to write log files?
- Log levels for each handler
- CLI vs. GUI
- Options
- Tracebacks
- Services

**Setting up `logging` Can Be A Significant Amount of Code**

# Balsa – Lightweight Logging

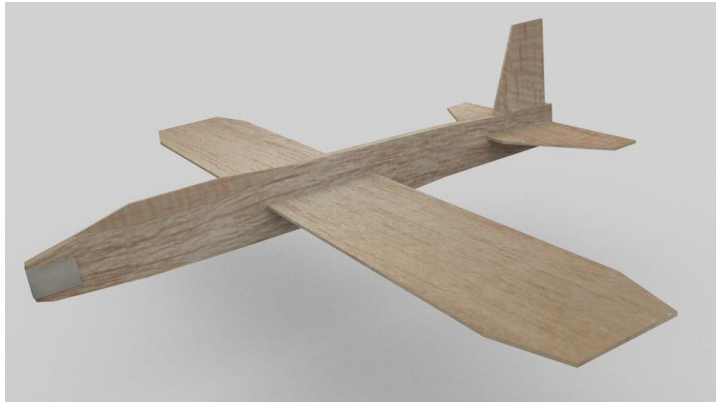


- Provide useful logging with just a few lines of code
- Consistent formatting and interface
  - `appdirs` for log file directory
- Console, GUI, files, exception services built-in
  - tkinter dialog box
  - Sentry (raven)
- Verbosity expressed by intent rather than level (e.g. `verbose` for development)
- `Error` level callback
- Available on PyPI

**`pip install balsa`**

www.abel.co @jamesabel www.github.com/jamesabel/balsa

# Simple Example



```
from balsa import get_logger, Balsa

application_name = 'example'
log = get_logger(application_name)

def main():
    balsa = Balsa(name=application_name , author='james abel')
    balsa.init_logger()

    log.error('my error example')
```

app name

function name

message

```
2018-08-18 20:43:33,756 - example - balsa_simple_example.py - 12 - main - ERROR - my error example
```

timestamp

source file name

line number

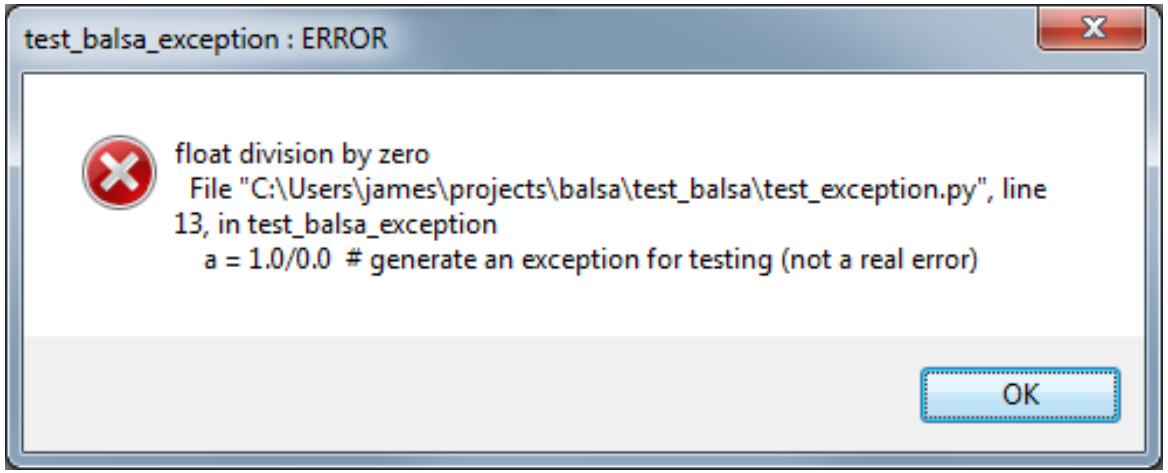
level

Also writes out a file (e.g. Windows):

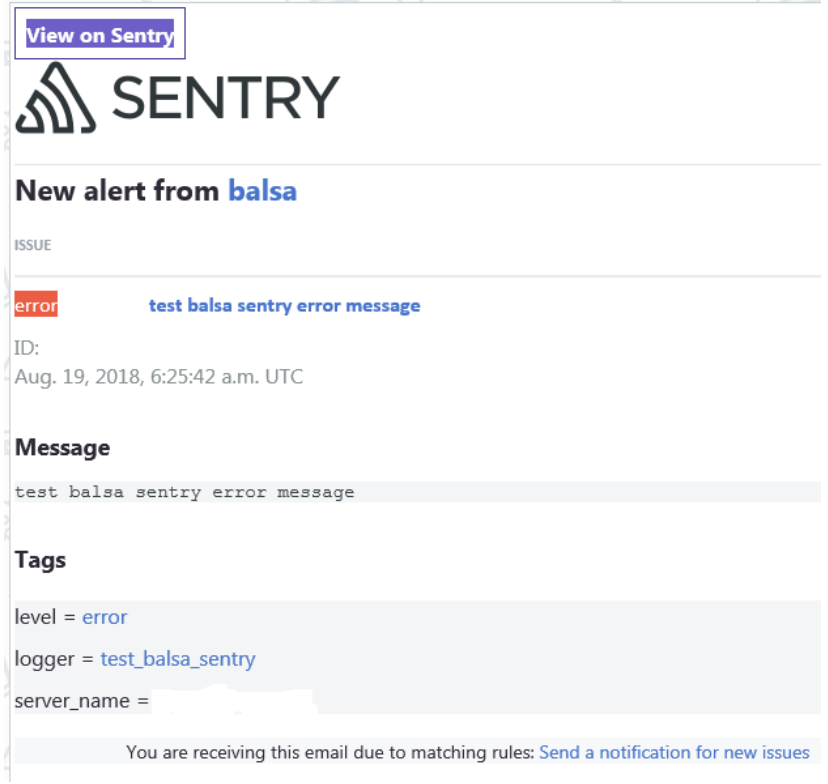
**C:\Users\<user>\AppData\Local\james abel\example\Logs\example.log**

www.abel.co @jamesabel www.github.com/jamesabel/balsa

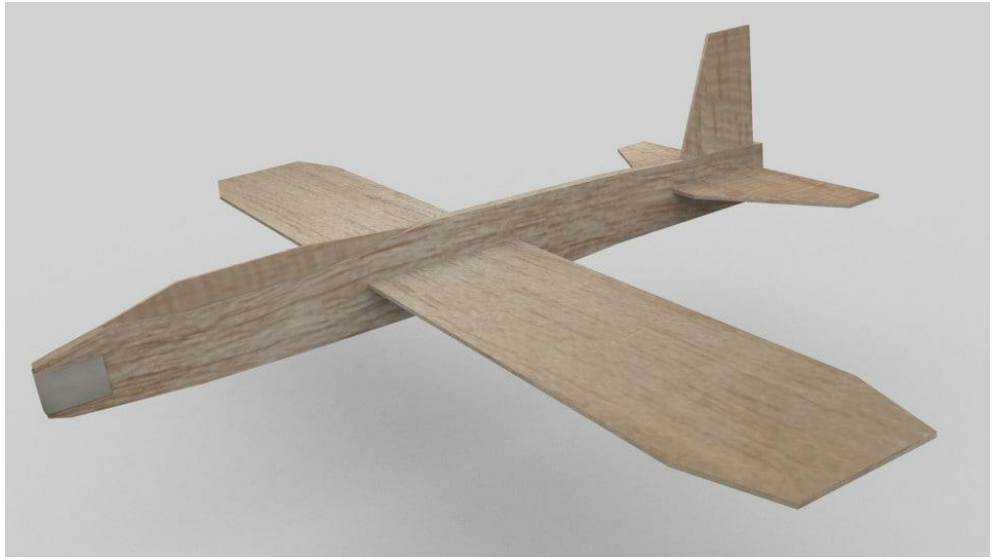
- GUI messages




- Sentry support (exception service)



# Summary and Thank You



- Balsa is lightweight logging!
- Try balsa!

  **`pip install balsa`**

  https://github.com/jamesabel/balsa

  http://balsa.readthedocs.io/
- Please provide feedback, issues, PRs, …
- Thanks to Mark Rice (@MRice88) for testing and feedback