The Most Diabolical Python Anti-Pattern

© MigrateUp, LLC - PowerfulPython.com

TMDPAP

Here's the most self-destructive code a Python developer can write:

```
try:
    do_something()
except:
    pass
```

This creates the worst kind of bug.

© MigrateUp, LLC - PowerfulPython.com

After a FULL WEEK

After a full WEEK of engineer time, I was able to isolate the bug to a single block of code:

```
try:
    extract_address(location_data)
except:
    pass
```

© MigrateUp, LLC - PowerfulPython.com

Why???

Why do people do this?

1) Because they expect an exception to occur that can be safely ignored.

That's fine; the problem is being overbroad. Just target narrowly instead:

```
try:
    extract_address(location_data)
except ValueError:
    pass

# Variation: Insert logging.
try:
    extract_address(location_data)
except ValueError:
    logging.info(
        "Invalid location for user %s", username)
```

© MigrateUp, LLC - PowerfulPython.com

Why?

2) Because a code path must continue running regardless of what exceptions are raised.

In that case, this is better:

```
while True:
    try:
        main_loop()
    except Exception:
        logging.exception('Error in main loop')
```

© MigrateUp, LLC - PowerfulPython.com

logging.exception()

Example stack trace:

```
ERROR:root:Error in main loop
Traceback (most recent call last):
  File "example-logging-exception.py", line 14, in <module>
    main_loop()
  File "example-logging-exception.py", line 8, in main_loop
ValueError: Incomplete data
```