Generators: The Final Frontier

Copyright (C) 2014
David M. Beazley
http://www.dabeaz.com

Presented at PyCon'14, April 10, 2014, Montreal.

Introduction

This tutorial discusses advanced uses of using generators to alter program control flow, explode brains, and exponentially increase your job security. Topics include context managers, inlined futures, concurrency, asyncio, actors, compilers, and more.

- Presentation Slides (PDF)
- Screencast from PyCon'2014 (Youtube)
- Official PyCon'2014 recording (pyvideo.org)

Support Data Files

The following file contains some supporting data files that are used by the various code samples. Download this to your machine to work the examples that follow.

- finalgenerator.zip

This download includes the presentation slides and all of the code sample below.

Background Material

This tutorial is part 3 of a trilogy of tutorials involving generators and coroutines. Although it stands on its own, you may want to review the first two parts to get the bigger picture:

- Generator Tricks for Systems Programmers. Presented at PyCon'08.
- A Curious Course on Coroutines and Concurrency. Presented at PyCon'09.

Code Samples

Here are various code samples that are used in the course. You can cut and paste these to your own machine to try them out. The order in which these are listed follow the course outline. You'll need Python 3.4.

Part 2 : And Now For Something Completely Different

- cm.py. Sample implementation of context managers using generators. Similar to `contextlib` standard library.

Part 3 : Call me Maybe

- simplefuture.py. Simple example of concurrent futures.
- inline1.py. First implementation of an inlined future generator.
- inline2.py. Inlined future with exception handling support.
- inline_recursive.py. Inlined future with odd recursion.

Part 4: Yield From Yield From Yield From Future

- inline_puzzle.py. Various attempts to make library functions.
- inline_iter.py. Near complete solution involving iterable Futures.

- inline_future.py. Final solution with tasks, inlined futures, proper result handling.
- async1.py. Simple example from asyncio library.
- async2.py. Echo server example from asyncio

Part 5: GIL

- fib1.py. Fibonacci numbers in a process pool and inlined futures.
- fib2.py. Performance test, side-stepping the GIL (it works!)
- fib3.py. Performance test with a different thread execution model.

Part 6 : Fake it Until You Make It (Actors)

- actor1.py. Simple actor example.
- actor2.py. Recursive ping-pong (broken).
- tactor.py. Recursive ping-pong using threads.
- actor3.py. Recursive ping-pong with simple message queue and runner.

Part 7: A Terrifying Visitor

- compiler1.py. A simple compiler and tree traversal using the visitor pattern.
- compiler2.py. Non-recursive tree traversal using generators (insane).