Generators

Advanced Python Programming

—

Valerio Velardo - The Sound of AI

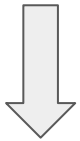
What’s a Generator?

What's a Generator?

A generator is a function that returns a generator object.

What's a Generator?

A generator is a function that returns a generator object.



Generator object is like a function that produces a sequence of values

How does a generator work?

How does a generator work?

- Sequence of values generated while iterating over it

How does a generator work?

- Sequence of values generated while iterating over it
- Use *yield* statement to generate each value in the sequence

How does a generator work?

- Sequence of values generated while iterating over it
- Use *yield* statement to generate each value in the sequence
- Execution of the generator is interrupted after *yield*

How does a generator work?

- Sequence of values generated while iterating over it
- Use *yield* statement to generate each value in the sequence
- Execution of the generator is interrupted after *yield*
- When *next()* is called on the generator -> resume execution after *yield*

Generator vs iterator

Generator vs iterator

- All you can do with a generator, you can do with a class-based iterator

Generator vs iterator

- All you can do with a generator, you can do with a class-based iterator
- However, __iter__() and __next__() come for free!

When should you use generators?

- Produce a ton of data
- Take advantage of lazy evaluation
- Don’t need flexibility of a custom iterator