Итераторы в Python

42 pythoners-PyHub Kyiv

Абудрахимов Родион

Duck Typing

- **Утиная типизация** - принцип динамической типизации, который определяет границы использования объекта набором его *текущих* методов и свойств, а не его родительским классом.

- **Правило**: все, что ходит как утка и крякает как утка - есть уткой

- Принцип очень активно используется в языке Python и также применяется в итераторах

Что такое итератор

Итератор - это вспомогательный объект, который помогает перечислять элементы контейнера.

Интерфейс итератора

Контейнер поддерживает итерирование, если в нем определены 2 метода:

- `iterator.__iter__()`
- `iterator.__next__()`

Любой объект, который реализует эти 2 метода по принципу утиной типизации - есть итератором

Что делают эти методы

- ***__iter__()*** - Возвращает объект итератора
- ***__next__()*** - Возвращает следующее значение контейнера либо выбрасывает исключение ***StopIteration***

- Для удобства использования есть встроенные функции ***iter(iterator)*** и ***next(iterator)***

Простой пример

```
numbers = [1, 2, 3, 4, 5]

iterator = iter(numbers)

print(next(iterator))  # -> 1
print(next(iterator))  # -> 2
print(next(iterator))  # -> 3
print(next(iterator))  # -> 4
print(next(iterator))  # -> 5
print(next(iterator))  # -> StopIteration exception
```

Собственный объект-итератор

```
class Fibonacci:
    def __init__(self, max_value):
        self.max = max_value

    def __iter__(self):
        self.a = 0
        self.b = 1
        return self

    def __next__(self):
        fib = self.a
        if fib > self.max:
            raise StopIteration
        self.a, self.b = self.b, self.a + self.b
        return fib

obj = Fibonacci(100)

for num in obj:
    print(num)

# -> 0 1 1 2 3 5 8 13 21 34 55 89
```

Изменения итераторов в версии Python 3+

- Изменения интерфейса итераторов касательно метода ***next()*** - в версии 3+ этот метод должен называться ***__next__()***. *(Python Enhancement Proposals (PEP) 3114) - Python v3.0*

- Изменение работы генераторов. При выбрасывании исключения StopIteration из генератора превращать его в исключение RuntimeError, что вызовет прерывание итератора с ошибкой, а не “по-тихому”. Упрощает отладку инетераторов. (PEP 479) - Python v3.5

Спасибо за внимание