

Python Essential

Генератори

Python Essential

Після уроку обов'язково



Повторіть цей урок у відео форматі на [ITVDN.com](http://itvdn.com)



Перевірте, як Ви засвоїли даний матеріал
на [TestProvider.com](http://testprovider.com)

Python Essential

Тема

Генератори

Ітератори та генератори

Генератори

Функція-генератор (generator function) – це функція, яка повертає спеціальний **ітератор генератора (generator iterator)** (також **об'єкт-генератор - generator object**). Вона характеризується наявністю ключового слова **yield** всередині функції.

- Термін **генератор (generator)**, залежно від контексту, може означати або функцію-генератор, або ітератор генератора (найчастіше, останнє).
- Методи `__iter__` та `__next__` у генераторів створюються автоматично.
- `yield` заморожує стан функції-генератора та повертає поточне значення. Після наступного виклику `__next__()` функція-генератор продовжує своє виконання з того місця, де вона була призупинена.
- Коли виконання функції-генератора завершується (за допомогою ключового слова `return` або досягнення кінця функції), виникає виняток **StopIteration**.



Ітератори та генератори

Вирази-генератори

Деякі прості генератори можуть бути записані у вигляді виразу. Вони виглядають як вираз, що містить деякі змінні, після якого одне або кілька ключових слів **for**, котрі задають, які значення повинні приймати дані змінні (синтаксис відповідає заголовку циклу **for**), і нуль або кілька умов, що фільтрують значення, які генеруються (синтаксис відповідає заголовку оператора **if**). Такі вирази називаються **виразами-генераторами (generator expressions)**.

```
function(x, y) for x in range(10) for y in range(5) if x != y
```



Ітератори та генератори

Підгенератори

У Python 3 є так звані **підгенератори (subgenerators)**. Якщо у функції-генераторі зустрічається пара ключових слів `yield from`, після яких слідує об'єкт-генератор, то даний генератор делегує доступ до підгенератора, поки він не завершиться (не закінчаться його значення), після чого продовжує своє виконання.

```
def generator():  
    ...  
    yield from subgenerator()  
    ...
```



Ітератори та генератори

Yield-вирази

- Насправді `yield` є виразом. Воно може набувати значень, які відправляються в генератор. Якщо в генератор не відправляються значення, результат цього виразу дорівнює `None`.
- `yield from` також є виразом. Його результатом є те значення, яке підгенератор повертає у виключенні `StopIteration` (для цього значення повертається за допомогою ключового слова `return`).

```
def generator():  
    ...  
    data = yield  
    ...
```



Ітератори та генератори

Методи генераторів

| Метод | Опис |
|--|---|
| <code>__next__()</code> | Починає або продовжує виконання функції-генератора. Результат поточного <code>yield</code> -виразу дорівнюватиме <code>None</code> . Виконання потім продовжується до наступного <code>yield</code> -виразу, який видає значення туди, де був викликаний <code>__next__</code> . Якщо генератор завершується без видачі значення за допомогою <code>yield</code> , виникає виняток <code>StopIteration</code> . Метод зазвичай викликається неявно, тобто циклом <code>for</code> або вбудованою функцією <code>next()</code> . |
| <code>send(value)</code> | Продовжує виконання та відправляє значення до функції-генератора. Аргумент <code>value</code> стає значенням поточного <code>yield</code> -виразу. Повертає видане значення. Якщо <code>send()</code> використовується для запуску генератора, то єдиним допустимим значенням є <code>None</code> , так як ще не було виконано жодного <code>yield</code> -виразу, якому можна привласнити це значення. |
| <code>throw(type[, value[, traceback]])</code> | Викидає виняток типу <code>type</code> у місці, де було призупинено генератор, і повертає наступне значення генератора (або викидає <code>StopIteration</code>). |
| <code>close()</code> | Викидає виняток <code>GeneratorExit</code> у місці, де було призупинено генератор. Якщо генератор повертає чергове значення, викидається виняток <code>RuntimeError</code> . |

Ітератори та генератори

Співпрограми

- **Співпрограма** (англ. **coroutine**) – компонент програми, що узагальнює поняття підпрограми, який додатково підтримує безліч вхідних точок (а не одну, як підпрограма) та зупинку і продовження виконання із збереженням певного положення.
- Розширені можливості генераторів у Python (вирази `yield` та `yield from`, відправка значень у генератори) використовуються для реалізації співпрограм.
- Співпрограми корисні для реалізації асинхронних неблокуючих операцій і кооперативної багатозадачності в одному потоці без використання функцій зворотного виклику (callback-функцій) і написання асинхронного коду в синхронному стилі.
- Python 3.5 включає підтримку співпрограм на рівні мови. Для цього використовуються ключові слова `async` та `await`.



Дивіться наші уроки у відео форматі

ITVDN.com



Перегляньте цей урок у відео форматі на освітньому порталі [ITVDN.com](http://itvdn.com) для закріплення пройденого матеріалу.

Усі курси записані сертифікованими тренерами, які працюють у навчальному центрі CyberBionic Systematics



Перевірка знань

TestProvider.com

TestProvider

Мы помогаем людям оценить себя

Регистрация Войти

Поиск сертификата

Главная Услуги и цены Центр Тестирования Поддержка О нас

Мы в социальных сетях

Тестирование

Языки программирования и информационные технологии

Microsoft

C# ASP.NET MVC JavaScript Patterns OF Design SQL Architecture Guide WCF HTML&CSS XML SEO WPF HTML5&CSS3 JQuery XNA SharePoint GUI for Android Windows Azure Platform Microsoft Patterns&Practices TFS SCRUM ReSharper TDD WWF LINQ Entity Framework Windows Forms Refactoring Microsoft Expression Blend 4 Windows Phone 8 Windows 8 AppStore Visual Studio Tips&Tricks MSF MEF SilverLight AJAX MEF Service Oriented Architecture

Добро пожаловать на TestProvider.com!

Сайт перенесен на новую облачную платформу с использованием системы единой авторизации Single Sign On. Если вы хотите восстановить статистику по предыдущим экзаменам обратитесь в [службу поддержки](#). Для восстановления информации с предыдущей версии сайта, просба написать в службу поддержки Ваш старый и новый логины.

ITVDN PROMETRIC TEST CENTER CyberBionic Microsoft Partner Windows Azure Cloud Partner EBA

TestProvider – це online сервіс перевірки знань з інформаційних технологій. З його допомогою Ви можете оцінити Ваш рівень та виявити слабкі місця. Він буде корисним як у процесі вивчення технології, так і для загальної оцінки знань IT-спеціаліста.

Після кожного уроку проходите тестування для перевірки знань на TestProvider.com

Успішне проходження фінального тестування дозволить Вам отримати відповідний Сертифікат.



Python Essential

Q&A

Інформаційний відеосервіс для розробників програмного забезпечення

