Engineer

Python

мова, якою написано більшість інструментів

Що це таке

Python — це мова програмування загального призначення, вихідний код якої виконує інтерпретатор, а не компілятор, що заздалегідь перетворює його на окремий двійковий файл. Власна документація мови називає її інтерпретованою, інтерактивною та об'єктно-орієнтованою, з вбудованими модулями, винятками, динамічною типізацією і високорівневими типами даних. Першою зазвичай помічають ту рису, що відступи тут — частина синтаксису: блок — це все, що зсунуто вправо під рядком, який його відкриває, тож розкладка, яку бачить читач, і є структурою, яку виконує інтерпретатор. Гвідо ван Россум почав працювати над нею на різдвяних вихідних 1989 року, спираючись на досвід мови ABC у CWI в Амстердамі, оприлюднив її в лютому 1991 року і назвав на честь сценаріїв «Літаючого цирку Монті Пайтона», які тоді читав.

Типізація динамічна. Ім'я прив'язується до об'єкта під час виконання програми, і тип належить об'єктові, а не імені, тож та сама змінна може зберігати число в одному рядку й рядок тексту в наступному. Необов'язкові анотації, стандартизовані в PEP 484, дають функції змогу оголосити, що вона очікує і що повертає, але цей PEP прямо каже, що під час виконання жодної перевірки типів немає і що Python лишиться мовою з динамічною типізацією. Анотації читають окремі інструменти, перевірячі типів і редактори, а інтерпретатор зберігає їх, не застосовуючи.

Мова постачається з великою стандартною бібліотекою, яку підручник описує словами «batteries included»: файли, JSON, SQLite, регулярні вирази, підпроцеси, розбір аргументів і ще багато чого, і все це без жодного встановлення. Решту дають сторонні пакети, які зазвичай встановлюють через pip із Python Package Index. Оскільки одне встановлення Python не

може задовольнити вимоги всіх програм одночасно, стандартна бібліотека містить і `venv`, що створює окреме середовище для кожного проєкту, тож одна програма може вживати версію 1.0 бібліотеки, а інша — 2.0.

Нові версії виходять за календарем. Починаючи з Python 3.9, PEP 602 випускає нову версію з новими можливостями щожовтня після приблизно сімнадцяти місяців розробки, і кожну з них підтримують п’ять років; від 3.13 це означає два роки звичайних випусків із виправленнями помилок, приблизно раз на два місяці, а потім три роки лише виправлень безпеки. Python 3.15.0 вийшов 9 жовтня 2026 року. Python 3.10 дійшов кінця підтримки 1 жовтня 2026 року, а Python 2.7, остання версія другої серії, перестав отримувати підтримку 1 січня 2020 року. Еталонний інтерпретатор, CPython, давно має глобальне блокування інтерпретатора, яке дає виконувати код Python лише одному потокові водночас; PEP 703 додав збірку без нього, а PEP 779 назвав 3.14 часовими рамками етапу, на якому ця вільнопотокова збірка офіційно підтримується, але лишається необов’язковою.

Застосуйте поняття до рішення

Застосуйте «Що таке Python?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Та частина, якої користувач не бачить: де зберігаються дані, які діють правила і як інші системи можуть із нею взаємодіяти.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Спроєктувала й підтримувала 6 пайплайнів ETL/ELT — 6 пайплайнів ETL на трьох рушіях баз даних

Спроєктувала й підтримувала 6 пайплайнів ETL/ELT Зв'язатися з нами →

Чому це важливо

Python часто дає найкоротший шлях від питання до робочої програми. Скрипт, що читає файл, робить запит до бази даних і пише звіт, можна написати самою лише стандартною бібліотекою, одразу запустити й змінювати без кроку збирання. Саме тому Python так часто стає сполучною ланкою в конвеєрах даних, інструментах збирання і тестових обв’язках.

Читабельність тут — свідома мета проєктування, а не побічний ефект. «Дзен Python», PEP 20, каже, що читабельність має значення, а PEP 8, посібник зі стилю для стандартної бібліотеки, вирішує ті питання, про які команда інакше сперечалася б, від назв до рядка в 79 символів.

Гнучкість має ціну, і ця ціна приходить пізно. Описка в назві атрибута, `None` там, де очікували рядок, або хибний тип аргументу стають помилкою лише тоді, коли виконується саме цей рядок, а це може статися вже в продакшні. Тому команди, що покладаються на Python у чомусь довговічному, додають перевірки, яких мова не робить сама: лінтер, анотації з перевірячем типів у воротах якості і тести, що справді проходять шляхами коду.

Календар важить не менше за код. Кожна версія інтерпретатора має фіксовану дату завершення, тож встановлення, якого ніхто не торкається, випадає з підтримки безпеки за оприлюдненим розкладом. Проєкт, що називає свою мінімальну версію і рухається вперед у межах двох років виправлень помилок, міняє аврал на одне заплановане оновлення на рік.

Як це виглядає на практиці

У портфоліо на одному задокументованому етапі Python забезпечував роботу генератора резюме й портфоліо на основі бази даних: 41 модуль і 10 580 рядків створювали шість вихідних форматів, від HTML і PDF до Markdown і JSON, з однієї бази SQLite на 23 таблиці. Цю базу будував конвеєр із 19 кроків, і кожен крок можна було безпечно повторити. Залежностей часу виконання лишили рівно дві, рушій шаблонів і засіб для відтворення PDF, з тим аргументом, що генератор документів, який неможливо встановити через п’ять років, нічого не зберіг. Натомість кожен документ тоді залежав від цього коду Python, й іншого способу створити його не лишилося. Ці числа описують реалізацію на тому задокументованому етапі.

Python виграв і пряме порівняння в конвеєрі погодинного агрегування споживання електроенергії, який збудували трьома способами, на Python, на JQ і на SQL у PostgreSQL, щоб перетворити нерегулярні показники на один рядок на годину й зону, де чисте споживання — це виробництво плюс імпорт мінус експорт. Варіант, написаний як чисті функції Python, виявився найшвидшим: дані за день він обробляв за 34 мс, а набір за 30 днів — за 180 мс.

Цей сайт збирає Hugo. На зафіксованому в жовтні 2026 року етапі розвитку інструментів 22 його допоміжні скрипти, загалом 8 408 рядків, були написані на Python. Кожен із них обходився самою стандартною бібліотекою, крім скрипта, що імпортував малюнки: йому був потрібен Pillow, щоб їх обрізати. Файл `.tool-versions` задавав Python 3.14.7 із правилом, що інтерпретатор міг бути новішим, але ніколи не старішим, і `task doctor` перевіряв це на

встановленому двійковому файлі, а не вірив файлові на слово. Ruff був лінтером, а basedpyright перевіряв типи на рівні `standard`, бо його типовий рівень видав 358 знахідок, майже всі про відсутні анотації в коді, який свідомо був простим. Під час першого запуску перевіряч типів знайшов значення, що могло бути `None`, використане як ключ словника, — дефект, який Ruff пропустив без зауважень.

Де виникають проблеми

Здебільшого клопіт із Python походить від коду, який інтерпретатор приймає без заперечень і який поводиться інакше, ніж читається.

Типове значення аргументу пам'ятає попередні виклики. Функція повертає список, у якому ще лежать елементи з попередніх викликів, тож результат залежить від порядку викликів. Типове значення обчислюється один раз, коли виконується інструкція `def`, тож змінюване типове значення, як-от список чи словник, — це один спільний об'єкт для всіх викликів, що пропускають цей аргумент. Задайте типовим значенням `None` і створюйте список у тілі функції; правило Ruff B006 повідомляє про цей шаблон.

```
def add_tag(tag, tags=None):
    if tags is None:
        tags = []  # новий список за кожного виклику
    tags.append(tag)
    return tags
```

Проєкти ламають залежності одне одного. Оновлення бібліотеки для одного проєкту ламає інший, або `pip install` зупиняється з помилкою про зовнішньо кероване середовище. В обох випадках пакети встановлено просто в спільний інтерпретатор: проєкти змагаються за одну копію кожної бібліотеки, а за PEP 668 той, хто розповсюджує операційну систему, може позначити її власний Python як закритий для pip. Дайте кожному проєктові окреме середовище через `python -m venv .venv`, встановлюйте пакети в нього й записуйте залежності у файл, щоб середовище можна було відтворити.

Анотаціям типів довіряють як перевіркам. Функція з анотацією `int` отримує рядок із файлу JSON, і збій виринає на три виклики пізніше. Анотації описують намір, а інтерпретатор їх не застосовує. Запускайте перевіряч типів

як частину воріт якості, щоб розбіжності в анотованому коді зупиняли збірку, і перевіряйте дані там, де вони потрапляють у програму, бо жодна анотація не ручається за вхідні дані.

Інтерпретатор випадає з підтримки. Сервер тримає ту версію, з якою його встановили, аж доки для неї припиняються випуски безпеки, а пакети прибирають її зі свого діапазону підтримки. Кожна версія з новими можливостями отримує п’ять років, і п’ять років 3.10 скінчилися 1 жовтня 2026 року. Указуйте мінімальну версію у файлі, який читає перевірка, переходьте вгору в межах двох років виправлень помилок кожного випуску й проганяйте тести на новій версії, перш ніж перемикатися.

Що почитати далі

- Python documentation — підручник, довідник бібліотеки й довідник мови для поточного випуску.
- The Python Tutorial — мова з самих основ, разом із застереженням про змінювані типові значення аргументів.
- PEP 8 – Style Guide for Python Code — правила розкладки й найменування, яких дотримується більшість коду на Python.
- PEP 602 – Annual Release Cycle for Python — чому нова версія з новими можливостями виходить щожовтня і скільки часу кожну підтримують.
- PEP 484 – Type Hints — синтаксис анотацій і те, чому інтерпретатор лишає перевірку іншим інструментам.
- Virtual Environments and Packages — окреме середовище для кожного проєкту через `venv` і встановлення пакетів у нього через pip.

Словник

Обирайте за фактичними умовами

Опишіть робочий процес, наявні інтерфейси та користувачів, яким потрібна зміна. Ми можемо визначити перевірюваний результат і спосіб передачі, який дасть вашій команді змогу підтримувати його.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке Python?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/python/