Engineer

Python

sproget, det meste af dette sites værktøj er i

Hvad det er

Python er et alsidigt programmeringssprog, hvis kildekode køres af en fortolker i stedet for at blive oversat på forhånd til en selvstændig binær fil. Dets egen dokumentation kalder det fortolket, interaktivt og objektorienteret, med moduler, undtagelser, dynamisk typning og højniveau-datatyper indbygget. Det træk, de fleste lægger mærke til først, er, at indrykning er syntaks: en blok er det, der er rykket ind under den linje, der åbner den, så det layout, læseren ser, er den struktur, fortolkeren kører. Guido van Rossum begyndte på det i juleferien 1989 med erfaringerne fra ABC-sproget på CWI i Amsterdam i bagagen, lagde det ud i februar 1991 og opkaldte det efter manuskripterne til Monty Python's Flying Circus, som han læste på det tidspunkt.

Typningen er dynamisk. Et navn bindes til et objekt, mens programmet kører, og typen hører til objektet, ikke til navnet, så den samme variabel kan rumme et tal på én linje og en streng på den næste. Valgfrie annotationer, standardiseret i PEP 484, lader en funktion erklære, hvad den forventer og returnerer, men den PEP siger udtrykkeligt, at der ikke sker nogen typekontrol ved kørsel, og at Python forbliver et dynamisk typet sprog. Annotationerne læses af separate værktøjer, typecheckere og editorer, mens fortolkeren gemmer dem uden at håndhæve dem.

Sproget leveres med et stort standardbibliotek, som tutorialen beskriver som »batteries included«: filer, JSON, SQLite, regulære udtryk, underprocesser, argumentparsing og meget mere, alt sammen uden at installere noget. Alt andet kommer som tredjepartspakker, som regel installeret med pip fra Python Package Index. Fordi én Python-installation ikke kan opfylde alle programmers krav på én gang, indeholder standardbiblioteket også `venv`, der laver et selvstændigt miljø pr. projekt, så ét program kan bruge version 1.0 af et bibliotek, mens et andet bruger 2.0.

Nye versioner kommer efter en kalender. Siden Python 3.9 har PEP 602 sendt en featureudgivelse ud hver oktober efter omkring sytten måneders udvikling, og hver af dem vedligeholdes i fem år; fra 3.13 betyder det to år med almindelige fejlrettelser, cirka hver anden måned, og derefter tre år med kun sikkerhedsrettelser. Python 3.15.0 udkom den 9. oktober 2026. Python 3.10 nåede sin end of life den 1. oktober 2026, og Python 2.7, den sidste i den anden serie, holdt op med at få support den 1. januar 2020. Referencefortolkeren, CPython, har længe haft en global fortolkerlås, der kun lader én tråd udføre Python-kode ad gangen; PEP 703 tilføjede et build uden den, og PEP 779 udpegede 3.14 som tidsrammen for den fase, hvor det frit trådede build er officielt understøttet, men stadig valgfrit.

Brug begrebet i en beslutning

Brug »Hvad er Python?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Den del, brugerne aldrig ser: hvor data ligger, hvilke regler der gælder, og hvordan andre systemer må tale med den.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har designet og drevet 6 ETL/ELT-pipelines — 6 ETL-pipelines på tre databasemotorer

Har designet og drevet 6 ETL/ELT-pipelines Kontakt os →

Hvorfor det betyder noget

Python er ofte den korteste vej fra et spørgsmål til et program, der virker. Et script, der læser en fil, spørger en database og skriver en rapport, kan skrives med intet andet end standardbiblioteket, køres med det samme og ændres uden et buildtrin. Derfor ender det så tit som limen i datapipelines, buildværktøjer og testopsætninger.

Læsbarhed er et bevidst designmål og ikke en bivirkning. The Zen of Python, PEP 20, siger, at læsbarhed tæller, og PEP 8, stilguiden for standardbiblioteket, afgør de spørgsmål, et team ellers ville diskutere, fra navngivning til linjen på 79 tegn.

Fleksibiliteten har en pris, og den kommer sent. En stavefejl i et attributnavn, en `None`, hvor der var ventet en streng, eller en forkert argumenttype er først en fejl, når netop den linje kører, og det kan være i produktion. Teams, der bruger Python til noget varigt, lægger derfor de tjek til, som sproget udelader: en linter, annotationer med en typechecker i kvalitetsporten og tests, der faktisk kører kodestierne.

Kalenderen betyder lige så meget som koden. Hver fortolkerversion har en fast slutdato, så en installation, ingen rører, glider ud af sikkerhedssupporten efter en offentlig tidsplan. Et projekt, der angiver sin mindste version og rykker videre inden for de to år med fejlrettelser, bytter en nødsituation ud med én planlagt opgradering om året.

Sådan ser det ud i praksis

I porteføljen drev Python på et dokumenteret trin i udviklingen en databasedrevet generator til CV og portefølje: 41 moduler og 10.580 linjer, der gengav seks outputformater, fra HTML og PDF til Markdown og JSON, ud fra én SQLite-database med 23 tabeller. En pipeline i 19 trin byggede databasen, og hvert trin var sikkert at gentage. Kørselsafhængighederne blev holdt nede på præcis to, en skabelonmotor og en PDF-gengiver, ud fra det argument, at en dokumentgenerator, der ikke kan installeres om fem år, ikke har bevaret noget. Til gengæld afhang hvert dokument dengang af den Python-kode, og der var ingen anden måde at frembringe et på. Disse tal beskriver implementeringen på det dokumenterede trin i udviklingen.

Python vandt også en direkte sammenligning i en pipeline, der aggregerer elforbrug time for time, som blev bygget på tre måder, i Python, i JQ og i SQL på PostgreSQL, for at gøre uregelmæssige målinger til én række pr. time og zone, med nettoforbruget som produktion plus import minus eksport. Udgaven skrevet som rene Python-funktioner var den hurtigste og klarede en dags data på 34 ms og et datasæt for 30 dage på 180 ms.

Dette site bygges af Hugo. I det dokumenterede øjebliksbillede af værktøjerne fra oktober 2026 var dets 22 hjælpescripts, i alt 8.408 linjer, skrevet i Python. Hvert af dem klarede sig med standardbiblioteket alene, undtagen det script, der importerede tegningerne, som skulle bruge Pillow til at beskære dem. Filen `.tool-versions` angav Python 3.14.7 med den politik, at fortolkeren gerne måtte være nyere, men aldrig ældre, og `task doctor` kontrollerede det mod den installerede binære fil i stedet for at stole på filen. Ruff var linteren, og basedpyright tjekkede typer på sit `standard` -niveau, fordi dets forvalgte niveau meldte 358 fund, der næsten alle handlede om manglende annotationer i kode, der med vilje var enkel. Ved sin første kørsel fandt typecheckeren en værdi, der kunne være `None` , brugt som nøgle i en ordbog, en fejl, som Ruff havde godkendt.

Hvor det går galt

Det meste besvær med Python kommer af kode, som fortolkeren accepterer uden indvendinger, og som opfører sig anderledes, end den læses.

En standardværdi husker tidligere kald. En funktion returnerer en liste, der stadig rummer elementer fra tidligere kald, så resultatet afhænger af rækkefølgen. En standardværdi evalueres én gang, når `def` -sætningen kører, så en foranderlig standardværdi som en liste eller en ordbog er ét fælles objekt for alle kald, der udelader argumentet. Brug `None` som standardværdi og opret listen inde i funktionen; Ruffs regel B006 melder mønstret.

```
def add_tag(tag, tags=None):
    if tags is None:
        tags = []  # a new list on every call
    tags.append(tag)
    return tags
```

Projekter ødelægger hinandens afhængigheder. En opgradering af et bibliotek til ét projekt får et andet til at fejle, eller `pip install` stopper med en fejl om et eksternt administreret miljø. Begge dele skyldes pakker, der installeres direkte i en fælles fortolker: projekterne kæmper om én kopi af hvert bibliotek, og efter PEP 668 kan den, der distribuerer et styresystem, markere systemets egen Python som lukket for pip. Giv hvert projekt sit eget miljø med `python -m venv .venv` , installér i det, og skriv afhængighederne ned i en fil, så miljøet kan bygges igen.

Typehints bliver betragtet som tjek. En funktion, der er annoteret til at tage en `int` , får en streng fra en JSON-fil, og fejlen dukker op tre kald senere. Annotationer beskriver en hensigt; fortolkeren håndhæver dem ikke. Kør en typechecker som en del af kvalitetsporten, så uoverensstemmelser i annoteret kode stopper buildet, og validér data dér, hvor de kommer ind i programmet, for ingen annotation kan stå inde for input.

En fortolker glider ud af supporten. En server beholder den version, den blev installeret med, indtil sikkerhedsudgivelserne for den stopper, og pakkerne fjerner den fra deres understøttede interval. Hver featureversion får fem år, og de fem år for 3.10 sluttede den 1. oktober 2026. Angiv den mindste version i en fil, som et tjek læser, ryk op inden for de to år med fejlrettelser for hver udgivelse, og kør testsuiten på den nye version, før der skiftes.

Læs mere

- Python documentation — tutorialen, biblioteksreferencen og sprogreferencen for den aktuelle udgivelse.
- The Python Tutorial — sproget fra bunden, med advarslen om foranderlige standardværdier.
- PEP 8 – Style Guide for Python Code — de konventioner for layout og navngivning, det meste Python-kode følger.
- PEP 602 – Annual Release Cycle for Python — hvorfor en featureudgivelse lander hver oktober, og hvor længe hver af dem understøttes.
- PEP 484 – Type Hints — syntaksen for annotationer, og hvorfor fortolkeren overlader kontrollen til andre værktøjer.
- Virtual Environments and Packages — ét miljø pr. projekt med `venv` og installation i det med pip.

Ordliste

Vælg ud fra jeres faktiske vilkår

Beskriv arbejdsgangen, de eksisterende grænseflader og de brugere, der har brug for ændringen. Vi kan definere en testbar leverance og en overdragelse, der gør teamet i stand til at vedligeholde løsningen.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er Python?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/python/