Engineer

Python

språket det meste av nettstedets verktøy er skrevet i

Hva det er

Python er et programmeringsspråk til generell bruk der kildekoden kjøres av en tolk i stedet for å bli kompilert på forhånd til en egen binærfil. Dokumentasjonen kaller det tolket, interaktivt og objektorientert, med moduler, unntak, dynamisk typing og datatyper på høyt nivå innebygd. Det de fleste legger merke til først, er at innrykk er syntaks: en blokk er det som står innrykket under linjen som åpner den, så oppsettet en leser ser, er den strukturen tolken kjører. Guido van Rossum begynte på språket i juleferien 1989, med erfaringen fra arbeidet med språket ABC ved CWI i Amsterdam, la det ut i februar 1991 og oppkalte det etter manusene til Monty Python's Flying Circus, som han leste på den tiden.

Typingen er dynamisk. Et navn bindes til et objekt mens programmet kjører, og typen hører til objektet, ikke til navnet, så den samme variabelen kan holde et tall på én linje og en streng på den neste. Valgfrie annotasjoner, standardisert i PEP 484, lar en funksjon oppgi hva den forventer og returnerer, men den PEP-en sier klart at ingen typesjekking skjer ved kjøretid, og at Python skal forbli et dynamisk typet språk. Annotasjonene leses av egne verktøy, typesjekkere og editorer, mens tolken lagrer dem uten å håndheve dem.

Språket leveres med et stort standardbibliotek, som veiledningen beskriver som «batteries included»: filer, JSON, SQLite, regulære uttrykk, underprosesser, argumenttolking og mer, alt uten å installere noe. Alt annet kommer fra tredjeparter, og pakkene installeres som regel med pip fra Python Package Index. Fordi én tolk bare kan holde én versjon av hver pakke, har standardbiblioteket også `venv`, som lager et selvstendig miljø per prosjekt, slik at ett program kan bruke versjon 1.0 av et bibliotek mens et annet bruker 2.0.

Nye versjoner kommer etter en kalender. Siden Python 3.9 har PEP 602 sendt ut en funksjonsutgivelse hver oktober etter ca. sytten måneders utvikling, og hver av dem vedlikeholdes i fem år. Fra 3.13 betyr det to år med vanlige feilrettingsutgivelser, omtrent annenhver måned, og deretter tre år med bare sikkerhetsrettinger. Python 3.15.0 kom ut 9. oktober 2026. Python 3.10 nådde slutten av levetiden sin 1. oktober 2026, og Python 2.7, den siste i den andre serien, sluttet å få støtte 1. januar 2020. Referansetolken, CPython, har lenge hatt en global tolkelås (global interpreter lock) som lar bare én tråd kjøre Python-kode om gangen. PEP 703 la til et bygg uten den, og PEP 779 pekte ut 3.14 som tidsrammen for fasen der dette bygget med fri tråding er offisielt støttet, men fortsatt valgfritt.

Bruk begrepet i en beslutning

Bruk «Hva er Python?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Delen brukerne aldri ser: hvor dataene ligger, hvilke regler som gjelder, og hvordan andre systemer får lov til å snakke med den.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Designet og driftet 6 ETL/ELT-pipelines — 6 ETL-pipelines på tvers av tre databasemotorer

Designet og driftet 6 ETL/ELT-pipelines Ta kontakt →

Hvorfor det er viktig

Python er ofte den korteste veien fra et spørsmål til et program som virker. Et skript som leser en fil, spør en database og skriver en rapport, kan skrives med standardbiblioteket alene, kjøres med en gang og endres uten et byggesteg. Det er derfor språket så ofte ender som limet i datapipelines, byggeverktøy og testrigger.

Lesbarhet er et bevisst designmål og ikke en bivirkning. The Zen of Python, PEP 20, sier at lesbarhet teller, og PEP 8, stilguiden for standardbiblioteket, avgjør spørsmålene et team ellers ville kranglet om, fra navngiving til linjen på 79 tegn.

Fleksibiliteten har en pris, og den kommer sent. Et feilstavet attributt, en `None` der det var ventet en streng, eller en argumenttype som er feil, er en feil først når den linjen kjører, og det kan være i produksjon. Team som lener seg på Python for noe som skal vare, legger derfor til de sjekkene språket lar være: en linter, annotasjoner med en typesjekker i kvalitetsporten, og tester som faktisk kjører kodeveiene.

Kalenderen betyr like mye som koden. Hver tolkeversjon har en fast sluttdato, så en installasjon som ingen rører, glir ut av sikkerhetsstøtten etter en publisert plan. Et prosjekt som oppgir minsteversjonen sin og flytter seg framover innenfor de to årene med feilrettinger, bytter en krise mot én planlagt oppgradering i året. For dere som eier et slikt prosjekt, er det en avveining som er lett å ta: én kjent jobb hvert år mot en ukjent jobb på et dårlig tidspunkt.

Slik ser det ut i praksis

I porteføljen drev Python på et dokumentert stadium i utviklingen en databasedrevet generator for CV og portefølje: 41 moduler og 10 580 linjer som gjenga seks utdataformater, fra HTML og PDF til Markdown og JSON, fra én SQLite-database med 23 tabeller. En pipeline i 19 trinn bygde databasen, og hvert trinn var trygt å gjenta. Avhengighetene under kjøring ble holdt på nøyaktig to, en malmotor og en PDF-renderer, ut fra argumentet om at en dokumentgenerator som ikke kan installeres om fem år, ikke har bevart noe. Til gjengjeld var hvert dokument da avhengig av den Python-koden, og det fantes ingen annen måte å lage et på. Disse tallene beskriver implementasjonen på det dokumenterte stadiet.

Python vant også en direkte sammenligning i en pipeline for strømforbruk per time, som ble bygd på tre måter, i Python, i JQ og i SQL på PostgreSQL, for å gjøre uregelmessige målinger om til én rad per time og sone, med netto forbruk som produksjon pluss import minus eksport. Varianten skrevet som rene Python-funksjoner var raskest: den behandlet én dags data på 34 ms og et datasett på 30 dager på 180 ms.

Dette nettstedet bygges av Hugo. I øyeblikksbildet av verktøyene som ble dokumentert i oktober 2026, var de 22 hjelpeskriptene, til sammen 8 408 linjer, skrevet i Python. Hvert av dem brukte bare standardbiblioteket, unntatt skriptet som importerte tegninger, som trengte Pillow for å beskjære dem. Filen `.tool-versions` oppga Python 3.14.7, med regelen at tolken kunne være nyere, men aldri eldre, og `task doctor` sjekket det mot den installerte binærfilen i stedet for å stole på filen. Ruff var linteren, og basedpyright sjekket typer på nivået `standard`, fordi det forhåndsvalgte nivået rapporterte 358 funn som nesten alle gjaldt manglende annotasjoner i kode som var enkel med vilje. Første gang typesjekkeren kjørte, fant den en verdi som kunne være `None`, brukt som nøkkel i en ordbok, en feil Ruff hadde sluppet gjennom.

Hvor det går galt

Det meste av trøbbelet med Python kommer fra kode som tolken godtar uten å protestere, og som oppfører seg annerledes enn den leses.

En standardverdi husker tidligere kall. En funksjon returnerer en liste som fortsatt holder elementer fra tidligere kall, så resultatet avhenger av rekkefølgen kallene kommer i. En standardverdi evalueres én gang, når `def` -setningen kjører, så en muterbar standardverdi som en liste eller en ordbok er ett delt objekt for hvert kall som utelater argumentet. Bruk `None` som standardverdi og lag listen inne i funksjonskroppen; Ruff-regelen B006 melder fra om mønsteret.

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

Prosjektene ødelegger avhengighetene for hverandre. En oppgradering av et bibliotek for ett prosjekt knekker et annet, eller `pip install` stopper med en feil om et miljø som styres eksternt. Begge deler kommer av pakker som er installert rett i en delt tolk: prosjektene må dele én kopi av hvert bibliotek, og etter PEP 668 kan den som distribuerer et operativsystem, merke sin egen Python som forbudt område for pip. Gi hvert prosjekt sitt eget miljø med `python -m venv .venv` , installer i det, og skriv avhengighetene ned i en fil, slik at miljøet kan bygges på nytt.

Typehint blir stolt på som om de var sjekker. En funksjon som er annotert til å ta en `int` , får en streng fra en JSON-fil, og feilen dukker opp tre kall senere. Annotasjoner beskriver hensikt, og tolken håndhever dem ikke. Kjør en typesjekker som en del av kvalitetsporten, så avvik mellom annotert kode får bygget til å feile, og valider data der de kommer inn i programmet, fordi en annotasjon aldri kan gå god for inndata.

En tolk eldes ut av støtten. En server beholder versjonen den ble installert med, helt til sikkerhetsutgivelsene for den stopper og pakkene fjerner den fra området de støtter. Hver funksjonsversjon får fem år, og de fem årene for 3.10 var over 1. oktober 2026. Oppgi minsteversjonen i en fil som en sjekk leser, flytt opp innenfor de to årene med feilrettinger for hver utgivelse, og kjør testene på den nye versjonen før dere bytter.

Les mer

- Python documentation — veiledningen, bibliotekreferansen og språkreferansen for den gjeldende utgivelsen.
- The Python Tutorial — språket fra grunnen av, med advarselen om muterbare standardverdier for argumenter.
- PEP 8 – Style Guide for Python Code — konvensjonene for oppsett og navngiving som det meste av Python-kode følger.
- PEP 602 – Annual Release Cycle for Python — hvorfor en funksjonsutgivelse kommer hver oktober, og hvor lenge hver av dem støttes.
- PEP 484 – Type Hints — syntaksen for annotasjoner, og hvorfor tolken overlater sjekkingen til andre verktøy.
- Virtual Environments and Packages — ett miljø per prosjekt med `venv`, og installasjon i det med pip.

Ordliste

Velg ut fra deres faktiske rammer

Beskriv arbeidsflyten, eksisterende grensesnitt og brukerne som trenger endringen. Vi kan definere en testbar leveranse og en overlevering som gjør teamet i stand til å vedlikeholde løsningen.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er Python?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/python/