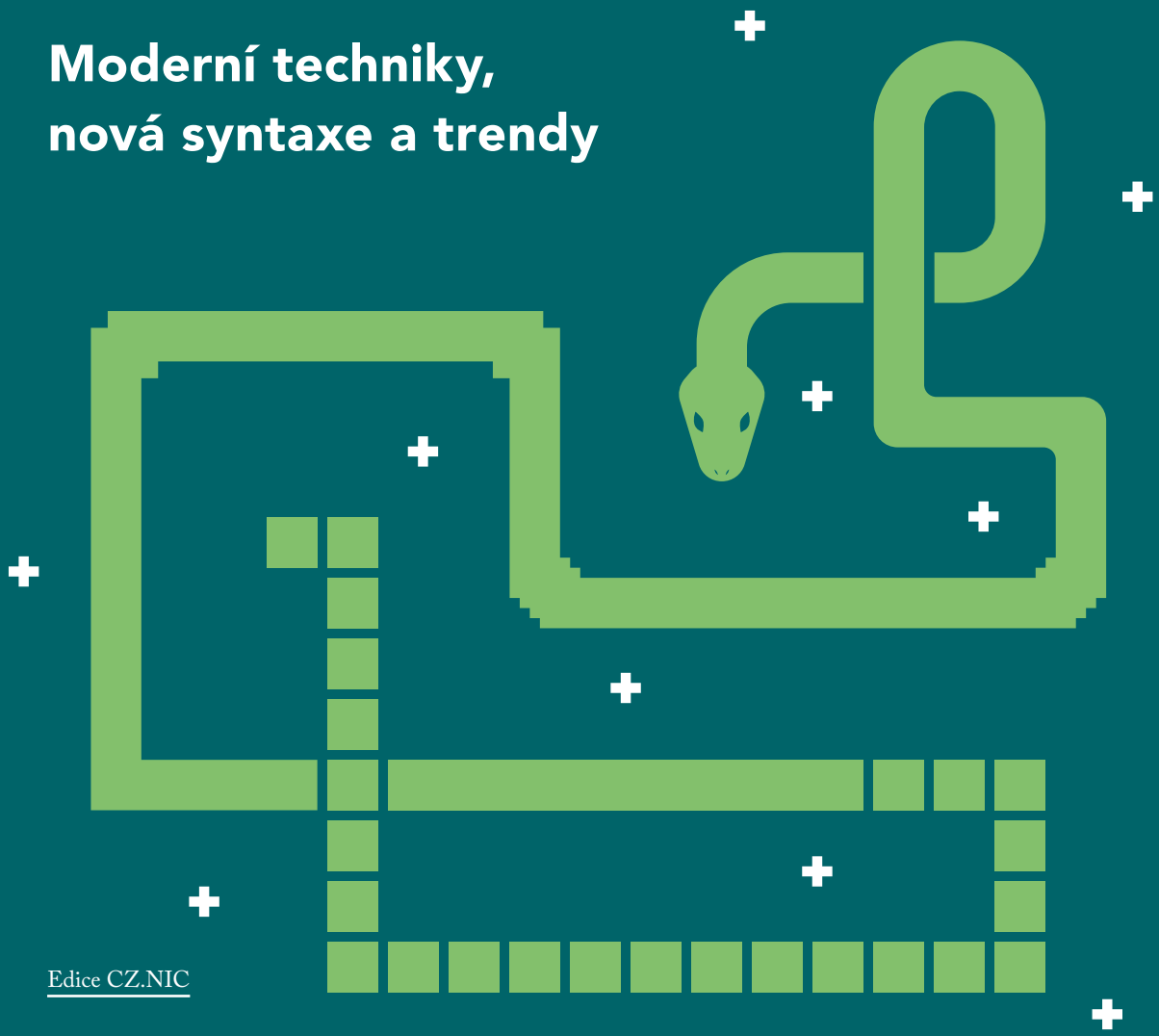
Pavel Tišnovský

Evoluce Pythonu

Moderní techniky, nová syntaxe a trendy



Edice CZ.NIC

EVOLUCE PYTHONU
Moderní techniky, nová syntaxe a trendy

Pavel Tišnovský

Vydavatel:
CZ.NIC, z. s. p. o.
Milešovská 5, 130 00 Praha 3
Edice CZ.NIC
www.nic.cz

1. vydání, Praha 2024
Kniha vyšla jako 31. publikace v Edici CZ.NIC.
ISBN 978-80-88168-77-5

ISBN 978-80-88168-77-5

— Pavel Tišnovský

Evoluce Pythonu

Moderní techniky, nová syntaxe a trendy

— Edice CZ.NIC

Předmluva vydavatele

Předmluva vydavatele

Python je programovací jazyk s dlouhou historií a širokým využitím od jednoduchých skriptů až po webové služby. V průběhu své existence se (stejně jako další jazyky) dále vyvíjí a mění, a je tedy třeba sledovat nové možnosti a trendy. Tato kniha nás provede historií jazyka a ukáže nejdůležitější novinky a změny, které nastaly za posledních 9 let. Jednotlivé nové vlastnosti jsou vždy popsány a doprovázeny příklady. Pochopení jejich dosahu a možného využití je tedy velmi jednoduché. Poměrně velká část knihy je věnována funkcionálnímu programování a jeho použití v Pythonu, a to jak v rámci standardní knihovny, tak i knihovny funcy. Popisuje přístupy a nástroje, které jsou potřeba k použití Pythonu jako funkcionálního jazyka. Zmíněna je též podpora pro běh aplikací ve více vláknech a/nebo procesech. Jsou probrány jednotlivé přístupy k paralelnímu zpracování včetně ukázek a benchmarků. Také je zde popsána možnost řízení a komunikace uvnitř tohoto typu aplikací. Za perličku považuji zmínění použití Pythonu pro vývoj frontendu webových aplikací, a to buď pomocí překladu Python kódu do JavaScriptu/WebAssembly, nebo použitím Python interpretru přímo ve webovém prohlížeči. Poslední část knihy je věnována správci projektů PDM a linteru Ruff. Nástroje tohoto typu jsou nedílnou součástí ekosystému a také zde dochází k vývoji a posunům standardů. Tyto relativně nové nástroje jsou představeny jako zajímavé alternativy k těm v současnosti rozšířeným. Kniha bude zajímavým doplněním knihovničky každého programátora (nejen) v Pythonu, své uplatnění ale nalezne určitě i u ostatních.

So, happy Pythoning and… always look on the bright side of life…

Tomáš Pazderka, CZ.NIC
Praha, 26. února 2024

Obsah

1 Úvod

1 Úvod

Programovací jazyk Python v současnosti patří mezi nejpopulárnější a nejčastěji nasazované programovací jazyky. Přitom se jedná o univerzální jazyk, který je možné použít v mnoha aplikačních oblastech. Python tak nalezne své využití jak při psaní krátkých a jednoduchých skriptů umožňujících a zjednodušujících administraci systému, tak i pro vývoj složitějších utilit, ale i například pro tvorbu desktopových aplikací (zde se využívají knihovny PyQt, PySide, wxPython, Tkinter, PyGObject, Wax či PySimpleGUI), webových služeb, pro analýzu a zpracování dat, strojové učení (*ML – machine learning*) a umělou inteligenci (*AI – artificial intelligence*). Python je dokonce možné v případě potřeby použít i na front endu, tedy pro ty části webových aplikací, které jsou spouštěny přímo ve webových prohlížečích u klientů. A díky existenci projektu MicroPython se jazyk Python používá i pro programování mikrořadičů.

Samotný programovací jazyk Python se přitom po celou dobu své existence (to je více než třicet let!) neustále vyvíjí, protože jsou do něj postupně přidávány nové vlastnosti. Ovšem ve skutečnosti se nevylepšuje pouze samotný jazyk Python, ale i celý jeho ekosystém, do něhož patří knihovny, analytické nástroje, ladicí nástroje, správci projektů, správci virtuálních prostředí atd. V této knize se budeme zabývat jak novinkami, které byly postupně do Pythonu přidány, tak i nástroji, které okolo tohoto programovacího jazyka vznikly a jenž například umožňují výše zmíněné použití Pythonu ve webovém prohlížeči. Zmíníme se i o některých trendech, jimiž dnes „žije“ komunita vývojářů používajících tento programovací jazyk. Jedná se v první řadě o takzvané typové anotace, které postupně začalo využívat velké množství nástrojů a překladačů Pythonu.

2 Stručná historie vývoje Pythonu

2 Stručná historie vývoje Pythonu

V této kapitole se ve stručnosti seznámíme s historií vývoje programovacího jazyka Python, a to od samotných prvopočátků (vše začalo programovacím jazykem nazvaným ABC) až k nejnovější verzi Pythonu, což je v době psaní této knihy Python 3.12. Celá historie je, minimálně z pohledu IT, poměrně dlouhá, protože výše zmíněný programovací jazyk ABC byl poprvé vydán již v roce 1987, tedy vlastně v počítačovém středověku (počítačovým pravěkem můžeme označovat dobu první generace takzvaných *mainframů* v padesátých letech minulého století).

2.1 Předchůdce Pythonu: programovací jazyk ABC

Autorem programovacího jazyka Python je Guido van Rossum. Před samotným popisem vývoje jazyka Python se však pro úplnost zmiňme o ještě starším programovacím jazyku, který se jmenuje *ABC*. Práce na vývoji jazyka ABC začala již v polovině osmdesátých let minulého století v CWI (Centrum Wiskunde & Informatica), což je instituce sídlící v Nizozemsku. A právě na vývoji tohoto programovacího jazyka má svůj podíl i Guido, který zde pracoval společně s Lambertem Meertensem, Leo Geurtsem a Stevenem Pembertonem (mimochodem: zajímavé je, že Guido byl v té době na juniorské pozici). Cílem tohoto vývojového týmu bylo vytvořit nový programovací jazyk, který by byl dobře použitelný i lidmi, kteří sice nejsou profesionálními programátory, ale aplikace potřebují vytvářet a nějakým způsobem je i udržovat (jednou z cílových skupin byli pochopitelně vědci; ostatně nejedná se ani o první a ani o poslední programovací jazyk s podobnými ambicemi).

Na tomto místě je možná dobré si ve stručnosti připomenout, v jakém stavu se vlastně nacházela informatika v polovině osmdesátých let minulého století. Právě v této době totiž došlo k obrovskému (a z pohledu dalšího vývoje i revolučnímu) rozvoji v oblasti osmibitových domácích mikropočítačů i šestnáctibitových osobních mikropočítačů. A především domácím osmibitovým mikropočítačům kraloval jiný programovací jazyk určený pro širokou veřejnost a nikoli (pouze) pro profesionály.

Jednalo se o tehdy slavný *BASIC (Beginners' All-purpose Symbolic Instruction Code)*, což byl programovací jazyk, který existoval v mnoha navzájem nekompatibilních dialektech a taktéž většina těchto dialektů nepodporovala strukturované programování, pracovalo se v něm pouze s globálními proměnnými a i typový systém byl pro větší projekty neuspokojivý (a to vůbec nemluvíme o nepodpoře objektově orientovaného programování). Tyto vlastnosti do určité míry vycházely z toho, aby byl jazyk jednoduše pochopitelný a použitelný, ovšem navíc se vycházelo z požadavků, aby se celý interpret BASICu mohl uložit do 2kB, 8kB či v některých případech do 16kB ROM.

Pro zajímavost se podívejme na to, jak vypadá jednoduchý program zapsaný v nestrukturovaném BASICu:

```
1 REM
2 REM ****************************
3 REM
4 REM Algoritmus bublinoveho razeni
5 REM
6 REM Uprava pro Atari BASIC
7 REM
8 REM ****************************
9 REM
10 DIM A(20)
11 FOR I=0 TO 20
12 A(I)=INT(100*RND(0))
13 NEXT I
14 GOSUB 100:REM TISK OBSAHU POLE
20 FOR I=19 TO 0 STEP -1
21 PRINT ".";
25 FOR J=0 TO I
30 IF A(J)<A(J+1) THEN X=A(J):A(J)=A(J+1):A(J+1)=X
35 NEXT J
40 NEXT I
49 PRINT ""
50 PRINT "SORTED:"
60 GOSUB 100:REM TISK OBSAHU POLE
99 STOP
100 REM TISK OBSAHU POLE
101 FOR I=0 TO 20
102 PRINT I,A(I)
103 NEXT I
104 RETURN
998 REM finito
999 STOP
```

Zdrojový kód tohoto příkladu můžete najít na adrese
https://github.com/tisnik/Evoluce_Pythonu_priklady/blob/master/historie/bubble-sort.bas

Na rozdíl od (většinou) nestrukturovaného BASICu, který navíc programátorům nabízel pouze základní datové typy (typicky se jednalo jen o čísla, textové řetězce, jednorozměrná pole a matice) byl programovací jazyk ABC navržen odlišným, dnes bychom mohli říci, že mnohem modernějším způsobem (zdá se, že Meertens dokonce BASIC přímo nesnášel, jeho mottem totiž bylo „Stamp out Basic!“). Ostatně podívejme se na jednoduchý příklad programu napsaného v jazyce ABC, který získá všechna slova ze vstupního dokumentu. Můžeme zde vidět poměrně velkou po-

dobnost s pozdějším Pythonem. Zejména je patrné, že odsazování řádků s programovým kódem je součástí syntaxe jazyka, dále zde nalezneme používání dvojteček, použití programové smyčky typu `for-each`, využití operátoru `not.in` apod.:

```
HOW TO RETURN words document:
   PUT {} IN collection
   FOR line IN document:
      FOR word IN split line:
         IF word not.in collection:
            INSERT word IN collection
   RETURN collection
```

Zdrojový kód tohoto příkladu můžete najít na adrese
https://github.com/tisnik/Evoluce_Pythonu_priklady/blob/master/historie/jazyk.abc

Malá poznámka k syntaxi jazyka ABC: slova `HOW TO RETURN` definují novou funkci, což vlastně znamená, že se jedná o obdobu k dnešnímu klíčovému slovu `def`. Referenční příručka celého jazyka ABC je dostupná na stránce https://homepages.cwi.nl/~steven/abc/qr.html.

2.2 Slibný začátek, ale pouze začátek

Ovšem nakonec se ukázalo, že svět IT nebyl na tento programovací jazyk připraven (někdo by dokonce mohl říci, že se jednalo o špatné načasování vstupu na trh, ovšem ABC nebyl vyvíjen na čistě komerční bázi). Na mikropočítačích s relativně malými systémovými zdroji nebylo možné potenciálu programovacího jazyka ABC využít a navíc mu v této oblasti velmi zdárně konkuroval jazyk BASIC, o kterém vycházely desítky, možná spíše i stovky knížek a který byl díky uložení přímo v paměti ROM logicky prvním jazykem většiny začátečníků (a popravdě řečeno: většina vlastníků domácích mikropočítačů o existenci ABC ani nevěděla).

Na druhé straně výkonnostního spektra, tedy typicky na počítačích s operačním systémem UNIX, se již používaly odlišné skriptovací jazyky, například programovací jazyk *Tcl* (1988) a *Perl* (1987). Na strojích vyráběných společností IBM se naproti tomu používal programovací jazyk *Rexx* (ten je dokonce ještě starší, protože pochází z roku 1979), ze kterého se později na slavné Amize vyvinul jazyk nazvaný *ARexx*. Nicméně i přes relativní neúspěch samotného jazyka ABC nebyly základní myšlenky v něm obsažené ztraceny, protože je později Guido van Rossum použil právě při implementaci první verze Pythonu, který z jazyka ABC v mnoha ohledech vycházel.

2.3 Odsazování řádků zdrojového kódu v jazycích ABC a Python

Zkušenější programátoři, kteří se poprvé seznamují s programovacím jazykem Python, bývají mnohdy překvapeni tím, že součástí syntaxe tohoto programovacího jazyka je i odsazení (tedy vlastně „prázdné“ znaky). Navíc se na začátku jednotlivých bloků kódu používá dvojtečka (naopak začátečníci, kteří jiný programovací jazyk neznají, tento fakt většinou zcela přirozeně přijmou). Na tomto místě si možná někdo položí otázku, kde se vlastně tento neobvyklý zápis programů objevil?

Dokonce ještě před samotným programovacím jazykem ABC bylo vytvořeno několik jeho předchůdců označovaných písmenem B a číslovkou (*B0*, *B1*, …). A již v jazyku se jménem *B0* se objevilo *povinné* odsazování, které zde mělo význam sdružení jednotlivých operací (příkazů a dalších programových konstrukcí) do programových bloků. Ovšem samotné bloky tehdy byly uvozeny klíčovými slovy `BEGIN` a `END`. Plánovalo se, že díky použití klíčových slov bude odsazování prováděno automaticky ve specializovaném programátorském textovém editoru (což byla v době vzniku B0 relativně přelomová myšlenka, která byla prakticky realizována až o mnoho let později).

Poznámka: výše uvedená klíčová slova `BEGIN` *a* `END` *ovšem známe například i z Pascalu nebo z programovacího jazyka Lua. Ovšem v těchto jazycích není odsazování striktně vyžadováno, na rozdíl od jazyka B.*

Ovšem poté si tvůrci jazyka B0 uvědomili, že vyžadovat odsazení a současně navíc i použití klíčových slov `BEGIN` a `END` je vlastně nadbytečné a proto v jazyce B1 (tedy ve druhé variantě programovacího jazyka B) již chybělo klíčové slovo `BEGIN` (což znamená, že jsme již na půli cesty k Pythonu). Namísto klíčového slova `END` se používalo spojení dvou klíčových slov, například `END IF`, `END FOR` atd. To tedy znamená, že podmínky a programové smyčky automaticky vytvářely bloky, což je podle mého názoru dobrý způsob strukturování programů. A nakonec se v jazyce B2 (tedy v pořadí již v jeho třetí variantě) programátoři „museli“ zcela obejít bez zápisu začátků a konců bloků pomocí klíčových slov – vše již bylo vyřešeno pouhým odsazením.

Poznámka: na tomto místě je vhodné doplnit, že se odsazení pro specifikaci bloků se používalo i v jiných programovacích jazycích (Miranda atd.), ovšem jazyk B byl pravděpodobně prvním jazykem, kde k této implementaci došlo.

2.4 Dvojtečka zapisovaná na začátku bloků kódu

Dalším typickým prvkem programovacího jazyka B (a posléze i jazyků ABC a Python) je dvojtečka zapisovaná na začátku bloků kódu, což může v moderním Pythonu vypadat následovně (pro zajímavost je ukázán algoritmus bublinkového řazení, který jsme viděli i v BASICové variantě):

```
import random

size = 100

a = [random.randrange(0, 10000) for i in range(size)]

for i in range(size - 1, 0, -1):
    for j in range(0, i):
        if a[j] > a[j + 1]:
            a[j], a[j + 1] = a[j + 1], a[j]

print("Sorted:")
print(a)
```

Zdrojový kód tohoto příkladu můžete najít na adrese
https://github.com/tisnik/Evoluce_Pythonu_priklady/blob/master/historie/bubble-sort.py

Myšlenka na použití dvojtečky v tomto kontextu vznikla v roce 1978, kdy Robert Dewar, Peter King, Jack Schwartz a náš starý známý Lambert Meertens navrhovali syntaxi jazyka B a porovnávali různě zápisy bubble sortu (opět!). Nakonec zavolali manželku Roberta Dewara a zeptali se jí, zda se jí navržená varianta líbí. Ta odpověděla, že má pocit, že se zápis `FOR i ...` vztahuje pouze k jednomu řádku a nikoli k celému bloku pod tímto řádkem. A právě na základě tohoto *alfa testingu* návrhu nového programovacího jazyka bylo rozhodnuto před začátkem bloku používat dvojtečku, což Pythonu vydrželo až do současnosti a s velkou pravděpodobností se již měnit nebude.

Poznámka: to vlastně znamená, že dvojtečka před začátkem bloku (namísto závorek či klíčových slov) není v žádném případě novým a neobvyklým vynálezem; je s námi v IT již více než 40 let.

2.5 Další skriptovací jazyk na již obsazené nice?

Jak jsme se již zmínili v předchozím textu, vznikl programovací jazyk Python na samotném začátku devadesátých let minulého století. Z mnoha pohledů se jednalo o důležitý mezník v rozvoji informatiky, protože právě tehdy se začala stále více rozšiřovat myšlenka, že programovací jazyky určené *pro vývoj plnohodnotných aplikací* lze zhruba rozdělit do dvou kategorií – překládané systémové jazyky (*compiled languages*) a jazyky skriptovací (*interpreted languages*, což je však hrubě nepřesné).

Samozřejmě, že se skriptovací jazyky používaly i před tímto obdobím, ale většinou se jednalo o relativně primitivní formy předpisů pro dávkové úlohy (výjimkou je například již zmíněný jazyk Rexx, jehož vyjadřovací prostředky již byly na vysoké úrovni). A navíc převažoval názor, že plnohodnotné aplikace musí být psány v překládaných jazycích, tedy typicky v jazycích ALGOLské větve se statickým typováním (schválně nepíšu se silným typováním, to je sice související, ovšem odlišná vlastnost, ostatně k velmi důležité problematice datových typů se ještě vrátíme).

V průběhu devadesátých let se tedy zpočátku mírně opovrhované skriptovací jazyky staly mnohdy nedílnou součástí mnoha profesionálních aplikací. Celý vývoj v této oblasti a s ním související myšlenkový posun byl nakonec shrnut ve slavném článku Johna Ousterhouta „Scripting: Higher Level Programming for the 21st Century", v němž se opakovala myšlenka na souběžné a kooperativní použití dvou jazyků – systémového a skriptovacího. Mimochodem: John Ousterhout věděl, o čem píše, protože je autorem programovacího jazyka TCL.

Poznámka: do skupiny klasických skriptovacích jazyků z této doby lze zařadit právě Python, dále Perl, Tcl, Rexx, shell (resp. jeho různé varianty) a taktéž poněkud později vytvořený jazyk Ruby. Ovšem ze širšího pohledu můžeme skupinu rozšířit například i o programovací jazyk Lua nebo o JavaScript a jazyky nad ním postavené (což je například CoffeeScript atd.).

2.6 Od Pythonu 0.9.0 k Pythonu 3.12

Jazyk Python se vyvíjí již čtyři desetiletí. V této podkapitole se ve stručnosti zmíníme o některých důležitých milnících.

2.6.1 První verze programovacího jazyka Python

Prvotní verze Pythonu, tedy verze, které vyšly ještě před oficiálním vydáním 1.0, jsou vypsány v následující tabulce:

Verze	Datum vydání
0.9.0	20. února 1991
0.9.1	konec února 1991
0.9.2	podzim 1991
0.9.4	24. prosince 1991
0.9.5	2. ledna 1992
0.9.6	6. dubna 1992
0.9.8	9. ledna 1993
0.9.9	29. července 1993

Jak již čísla verzí napovídají, nejednalo se o stabilní produkt.

2.6.2 Python 1

První stabilní (zejména ve smyslu již zmíněné *sémantiky*) verze programovacího jazyka Python byla vydána v roce 1994 a verze 1.x byly postupně vydávány až do roku 2001 (!), v němž vyšla poslední jedničková verze 1.6.1, která celou řadu Python 1 uzavřela. V Pythonu 1 ještě pochopitelně nenalezneme všechny vlastnosti, které známe ze stále ještě částečně rozšířeného Pythonu 2 nebo Pythonu 3 (který dnes Python 2 postupně nahrazuje).

Například řetězce byly v Pythonu 1 čistě osmibitové (ASCII), zatímco už v Pythonu 2 bylo možné použít Unicode řetězce. Dále v Pythonu 1 nebyla podporována takzvaná *generátorová notace* a od ní odvozené konstrukce pro inicializaci seznamů, množin a slovníků. Taktéž ale neexistovaly ani operátory spojené s operací přiřazení, tedy například operátor `+=`, `|=` apod. Tyto varianty operátorů byly přidány až do Pythonu 2; inspirací byly v této oblasti pochopitelně především céčkové jazyky. A konečně, v Pythonu 1 neměly řetězce žádné metody, například `string.startswith` atd. – i tato funkcionalita byla přidána až v Pythonu 2.

V následující tabulce jsou vypsány všechny oficiálně vydané verze Pythonu 1:

Verze	Datum vydání
1.0.0	26. ledna 1994
1.0.2	15. února 1994
1.0.3	4. května 1994
1.0.4	14. července, 1994
1.1	11. října 1994
1.1.1	10. listopadu 1994

1.2	13. dubna 1995
1.3	13. října 1995
1.4	25. října 1996
1.5	3. ledna 1998
1.5.1	31. října 1998
1.5.2	13. dubna 1999
1.6	5. září 2000
1.6.1	25. února 2001

Poznámka: v současnosti, kdy verze nějakého frameworku nebo knihovny určené pro ekosystém jazyka JavaScript vydaná před dvěma týdny je mnohdy považována za zoufale zastaralou, se to může zdát zvláštní, ale stále se můžeme ve specifických případech setkat s použitím Pythonu verze 1. Konkrétně jsem viděl použití Pythonu 1.5.2 a 1.6.1. Ostatně není bez zajímavosti, že tyto verze jsou dostupné ve formě zdrojových kódů a jsou stále přeložitelné i na současných verzích Linuxu či systému Microsoft Windows (i když pro nové projekty vřele doporučuji používat Python 3).

2.6.3 Python 2

Vydání Pythonu 2.0 datujeme na 16. říjen 2000. Python 2 byl velmi úspěšným projektem, který do značné míry zajistil Pythonu místo na samotném vrcholu popularity programovacích jazyků. Ovšem již v roce 2008 vyšel Python 3.0 (založený na PEP 3000, což bylo někdy spojováno do označení „Python 3000“), který je s verzí 2 částečně nekompatibilní. A právě popularita a velké rozšíření Pythonu 2 a jeho balíčků na jedné straně a nekompatibilita Pythonu 3 na straně druhé zapříčinila více než desetileté schizma světa Pythonu, které je teprve postupně napravováno s tím, jak se stále více balíčků a aplikací portuje na Python 3 (a v některých oblastech ani zdaleka není ukončeno, nicméně všechny nejdůležitější knihovny jsou již nabízeny buď výhradně pro Python 3 nebo jak pro Python 2, tak i pro Python 3).

Poznámka: na druhou stranu je nutné poznamenat, že se přechod na Python 3 v mnoha oblastech podařil a to v mnoha případech relativně snadno. Nenastala tedy taková patová situace, jako v případě Perlu 5 vs. Perl 6 (v současnosti Raku).

Opět si pochopitelně vypíšeme data vydání různých verzí Pythonu 2 (bez „setinkových“ vydání:

Verze	Datum vydání
2.0	16. října 2000
2.1	17. dubna 2001
2.2	21. prosince 2001
2.3	29. července 2003
2.4	30. listopadu 2004
2.5	16. září 2006
2.6	1. října 2008
2.7	3. července 2010

Žádná z těchto verzí Pythonu již není oficiálně podporována a doporučuje se, aby všechny nové projekty vznikaly již pro Python 3.

2.6.4 Python 3

Python 3.0, tedy první verze Pythonu řady 3, byla vydána v prosinci 2008. Za zmínku stojí „překryv" v současném vydávání Pythonu 2.x a 3.x. Většina informací z této knihy se vztahuje právě k novějším verzím Pythonu 3.x:

Verze	Datum vydání
3.0	3. prosince 2008
3.1	27. června 2009
3.2	20. února 2011
3.3	29. září 2012
3.4	16. března 2014
3.5	13. září 2015
3.6	23. prosince 2016
3.7	27. června 2018
3.8	14. října 2019
3.9	5. října 2020
3.10	4. října 2021
3.11	24. října 2022
3.12	2. října 2023

Povšimněte si velké pravidelnosti ve vydávání verzí, ke které došlo od verze 3.8. Příští plánovaná verze bude 3.13 a vyjít by měla letos, a to opět v říjnu.

2.7 Popularita Pythonu

V současnosti patří programovací jazyk Python mezi nejpopulárnější programovací jazyky vůbec. Nachází se na předních místech v mnoha statistikách, z nichž jmenujme například:

1. Tiobe index dostupný na adrese https://www.tiobe.com/tiobe-index/
2. PYPL PopularitY of Programming Language dostupný na adrese https://pypl.github.io/PYPL.html
3. Top 10 programming languages na GitHubu dostupný na adrese https://github.blog/2023-11-08-the-state-of-open-source-and-ai/