

Eingebaute Datentypen von Python

(C) 2016-2021 T.Birnthaler OSTC GmbH

Python stellt eingebaute Standard-Datentypen bereit, die von vornherein nach dem Start des Python-Interpreters verfügbar sind und typische Einsatzfälle abdecken. Zusätzlich stellt es weitere Datentypen über die Standard-Bibliothek bereit, dazu ist aber vorher ein passendes Modul zu laden (z.B. "array", "time", "datetime", "calendar", "decimal", "fractions", "collections", "enum").

Doku --> <http://docs.python.org/3/library/stdtypes.html>

Typ	Eigenschaften	Syntax und Werte (bereich)
NoneType	i 1	None (unbekannter/undefinierter Wert)
bool	i 1 Boolean	True/False (Wert 1/0 bei Rechnung)
int	i 1 Integer	1234 -5 0 (Ganze Zahl, beliebig groß)
float	i 1 Float	1.0 1e2 1E-2 (64-Bit Fließkommazahl, IEEE 754)
complex	i 1 2xFloat	(1+2j) (0+0j) (Real+Imaginärteil: 2x float)
str	i Seq UTF8	"ZEICHENKETTE" 'abc' "" "def"" ""
tuple	i Seq Obj	(OBJ1, OBJ2, ...) (OBJ1,)
list	m Seq Obj	[OBJ1, OBJ2, ...] []
dict	m Dict Imu:Obj	{ KEY1:OBJ1, KEY2:OBJ2, ... } {}
set	m Set Imu	{ KEY1, KEY2, ... } set({})
bytearray	m Seq Byte	bytearray(b"AF0310FF3404")
bytes	i Seq Byte	b"AF0310FF3404"
frozenset	i Set Imu	frozenset({ KEY1, KEY2, ... })
int	i 1 PY2	-2147483648 .. +2147483647 (32-Bit Ganzzahl)
long	i 1 PY2	-9223372036854775808..+9223372036854775807 (64)
str	i Seq Byte PY2	"ZEICHENKETTE" 'abc' "" ''
unicode	i Seq UTF8 PY2	u"ZEICHENKETTE" u'abc' u"" u''
Datentyp	- -	NoneType bool int float complex str tuple list dict set bytearray bytes frozenset
Klasse	m Attr Scope	class CLASSNAME: ...
Objekt	m Attr	OBJNAME = CLASSNAME(CLASS,...)
Funktion	i Attr Scope	def FUNCNAME(PARAM,...): ...
Funktion	i Attr Scope	lambda PARAM,...: EXPRESSION
Modul	m Attr Scope	import MODULENAME (Bezeichner ohne Ext. ".py!")

Legende:

None = Wert (entspricht "NULL" in SQL)
 1 = EIN Wert (Skalar, IMMER immutable)
 Seq = Sequenz (geordnete Folge von Werten = indiziert von 0..n-1)
 Dict = Dictionary (ungeordnete Key+Value-Paare, Key unique + immutable)
 Set = Set (ungeordnete unique + immutable Keys, "degeneriertes" Dict)
 i = immutable ("read-only", frozen, static, const, Wert NICHT Änderbar)
 m = mutable ("read-write", Wert Änderbar)
 Obj = Object (beliebiges Python-Objekt als Element)
 Imu = Immutable (beliebiges immutable Python-Objekt als Key/Element)
 Byte = Byte (Byte im Bereich 0..255 als Element)
 UTF8 = Character (Zeichen in UTF-8 Codierung als Element)
 Attr = Attribut (OBJ.ATTR = WERT für beliebige Attribute ATTR möglich)
 Scope = Name Scope (Lokaler Namensraum)
 PY2 = Python 2 (in Python 3 anders oder nicht vorhanden)

Begriffe:

Zahl = bool + int + float + complex (in Rechenausdruck verwendbar)
 Collection = Seq + Dict + Set + ... (Sammlung beliebig vieler Werte, auch leer)
 Container = Analog
 Float = 64 Bit Fließkommazahl (IEEE 754 Standard)
 Bezeichner = 1. Zeichen aus "A-Za-z_", folgende Zeichen aus "A-Za-z_0-9"
 Objekt = Speicher mit Wert + Datentyp + Referenzzähler + Id (Adresse)
 Referenz = Zeiger auf Objekt (nur Referenzen zugewiesen/weitergereicht)
 Referenzzähler erreicht 0 --> Objekt wird freigegeben (Speicher)
 Variable = Enthält Referenz auf Objekt (Name per Referenz an Obj "gebunden")
 del VAR --> Name VAR weg + Referenzzähler dekrementiert
 VAR = OBJ1 --> Objekt OBJ1 erzeugt, Referenzzähler auf 1,
 Referenz darauf in VAR
 VAR2 = VAR --> Referenz auf OBJ1 nach VAR2 kopiert,
 Referenzzähler auf 2
 VAR = OBJ2 --> Objekt OBJ2 erzeugt, Referenzzähler auf 1,
 Referenz darauf in VAR,

COW = Copy on Write (bei Zuweisung) Referenzzähler von OBJ1 dekrementiert

Datentypen:

str	im-mutable	Zeichenkette
dict	mutable	Map/Assoziatives Array/Hash/Zuordnung/Verzeichnis
list	mutable	Array
tuple	im-mutable	Array
set	mutable	Menge
frozenset	im-mutable	Menge
bytearray	mutable	Byte-Strom
bytes	im-mutable	Byte-Strom