Python "Magic Methods" (C) 2018-2021 T.Birnthaler OSTC GmbH

Python kennt 94 magische Methoden ("dunder methods" = double underscore), die Python automatisch aufruft, wenn die entsprechende Operation mit einem Objekt (oder einer Klasse) durchgefÃ¼hrt wird (P2/P3 = nur in Python 2/3).

Doku --> http://docs.python.org/3/reference/datamodel.html#object.__new__

Magische Methode		Aufruf durch ...	Bedeutung
__new__		CLS()	Subclass immutable type
__init__		CLS()	Konstruktor
__del__		del OBJ garbage collector	Destruktor
__slots__		CLS attributes	Attribute
__enter__		with ... as ...	Kontext-Manager
__exit__		with ... as ...	"
__iter__		for ... in ..., enumerate()	Iterator-Protokoll
__next__		for ... in ..., enumerate()	"
__getinitargs__		pickle.dump()/.restore()	Pickling-Protokoll
__getnewargs__		pickle.dump()/.restore()	"
__getstate__		pickle.dump()/.restore()	"
__setstate__		pickle.dump()/.restore()	"
__reduce__		pickle.dump()/.restore()	"
__reduce_ex__		pickle.dump()/.restore()	"
__copy__		copy.copy()	Flache Kopie (shallow)
__deepcopy__		copy.deepcopy()	Tiefe Kopie (deep)
__repr__		repr()	Maschinen-ReprÃ¤sentation
__str__		str()	Menschliche Darstellung
__format__		F"..." "...".format()	Formatierung
__unicode__	P2	unicode()	in P3 __str__ + __byte__
__hash__		hash()	
__nonzero__	P2	bool() on instances	in P3 renamed __bool__
__bool__		bool()	Typ-Konvertierung
__bytes__		bytes()	"
__complex__		complex()	"
__int__		int()	"
__long__	P2	long()	"
__float__		float()	"
__oct__		oct()	Zahlen-Konvertierung
__hex__		hex()	"
__bin__		bin()	"
__index__		OBJ[N:M:S]	Slice-Operator
__dir__		dir()	Objekt-Attributliste
__call__		OBJ() called as function	x() --> x.__call__()
__coerce__	P2		Mixed mode arithmetic
__getattr__		OBJ.ATTR getattr()	Objekt-Attribute
__getattribute__		OBJ.ATTR getattr()	"
__setattr__		OBJ.ATTR = VAL setattr()	"
__delattr__		del OBJ.ATTR delattr()	"
__get__			Deskriptor-Klasse TODO
__set__			"
__delete__			"
__set_name__			"
__init_subclass__			TODO
__instancecheck__		isinstance()	
__subclasscheck__		issubclass()	
__class_getitem__			TODO
__len__		len()	Anzahl Container-Elemente
__length_hint__			TODO
__reversed__		reversed()	Reihenfolge umdrehen
__contains__		in not in	EnthÃ¤lt (nicht)
__getitem__		OBJ[KEY]	Indexzugriff auf Element
__missing__		OBJ[KEY]	dict-Subklasse
__setitem__		OBJ[KEY] = VALUE	"
__delitem__		del OBJ[KEY]	"
__eq__		x == y	Werte-Vergleich

__ne__		x != y	"
__lt__		x < y	"
__le__		x <= y	"
__gt__		x > y	"
__ge__		x >= y	"
__cmp__	P2	x </==/> y --> -1/0/1	"
__add__		+	Arithmetische Operatoren
__sub__		-	"
__mul__		*	"
__matmul__		@	"
__div__	P2	/	"
__truediv__	P3	/	"
__floordiv__	P3	//	"
__mod__		%	"
__divmod__		divmod()	"
__pow__		** pow()	"
__lshift__		<<	"
__rshift__		>>	"
__and__		&	"
__or__		|	"
__xor__		^	"
__radd__		+	Reflektierte arith. Op.
__rsub__		-	"
__rmul__		*	"
__rmatmul__		@	"
__rdiv__	P2	/	"
__rtruediv__	P3	/	"
__rfloordiv__	P3	//	"
__rmod__		%	"
__rdivmod__		divmod()	"
__rpow__		**	"
__rlshift__		<<	"
__rrshift__		>>	"
__rand__		&	"
__ror__		|	"
__rxor__		^	"
__iadd__		+=	Erweiterte Zuweisung
__isub__		-=	"
__imul__		*=	"
__imatmul__		@=	"
__idiv__	P2	/=	"
__itruediv__	P3	/=	"
__ifloordiv__	P3	//=	"
__imod__		%=	"
__ipow__		**=	"
__ilshift__		<<=	"
__irshift__		>>=	"
__iand__		&=	"
__ior__		|=	"
__ixor__		^=	"
__neg__		-	UnÃ¤re arithmetische Op.
__pos__		+	"
__abs__		abs()	"
__invert__		~	"
__round__		round()	FlieÃM-^_komma-Konvertierung
__floor__		floor()	"
__ceil__		ceil()	"
__trunc__		trunc()	"

HINWEISE:

* Die Zuweisung "=" ist KEIN Operator und kann daher nicht Ã¼berladen werden
  (gibt immer ID zu einem Objekt zurÃ¼ck).

* "is" und "is not" vergleichen die Objekt-IdentitÃ¤t per ID und kann auch
  nicht Ã¼berladen werden. AbkÃ¼rzung fÃ¼r:
    o1 is o2   # id(o1) == id(o2)
    o1 is not o2   # id(o1) != id(o2)

* __iOP__ = Intrinsic Operator, aufgerufen bei verkÃ¼rzter Zuweisung a OP= b

* __rOP__ = Reflected Operator, aufgerufen falls die von o1 OP o2 aufgerufene
  Funktion type(o1).__OP__(o2) den Wert "NotImplemented" zurueckgibt oder falls
  die Funktion type(o1).__OP__() nicht vorhanden ist.