Doku --> http://docs.python.org/3/glossary.html#term-object
http://docs.python.org/3/glossary.html#term-new-style-class
http://docs.python.org/3/library/functions.html?highlight=object#object
http://docs.python.org/3/reference/datamodel.html?highlight=object

Python-OBJEKTE haben grundsätzlich folgende Eigenschaften und Fähigkeiten:

```
  type(OBJ)                     # Klasse von Objekt
  isinstance(OBJ, TYPE)         # Objekt ist aus Klasse oder Oberklasse
  isinstance(OBJ, (TYPE, ...)) # Objekt ist aus Klassen oder Oberklassen
  id(OBJ)                       # Identität von Objekt (Speicheradresse)
  str(OBJ)                      # Textform von Objekt (lesbare Form)
  repr(OBJ)                     # Textform von Objekt (Konstruktor-Aufruf)
  print(OBJ)                    # Textform von Objekt ausgeben
  F"{OBJ}"                      # Textform von Objekt ausgeben
  bool(OBJ)                     # Boolescher Wert von Objekt (STD: False)
  sys.getsizeof(OBJ)            # Größe von Objekt in Byte
  sys.getrefcount(OBJ)          # Anzahl Referenzen auf Objekt
                                #
  OBJ == OBJ2                   # WERT-mäßig gleich ("is" ohne __eq__)
  OBJ != OBJ2                   # WERT-mäßig verschieden ("is not" ohne __eq__))
  OBJ is OBJ2                   # IDENTITÄT gleich (id(OBJ) == id(OBJ2))
  OBJ is not OBJ2               # IDENTITÄT verschieden (id(OBJ) != id(OBJ2))
                                #
  del OBJ                       # Name + (EINE) Referenz auf Objekt löschen
                                #
  l = [OBJ, ...]                # Objekt ist Element in Liste
  t = (OBJ, ...)                # Objekt ist Element in Tupel
  d = {OBJ: VAL, ...}           # Objekt ist Key in Dictionary (falls "hashable")
  d = [KEY: OBJ, ...}           # Objekt ist Value in Dictionary
  s = {OBJ, ...}                # Objekt ist Element in Set (falls "hashable")
  OBJ [not] in l                # Objekt (nicht) in Liste enthalten?
  OBJ [not] in t                # Objekt (nicht) in Tupel enthalten?
  OBJ [not] in d                # Objekt (nicht) in Dictionary enthalten? (als Key)
  OBJ [not] in s                # Objekt (nicht) in Set enthalten?
                                #
  for ELEM (OBJ, ...):          # Schleife über Objekte
      ...                       #
                                #
  def FUNC(OBJ):                # Funktions-Parameter + Rückgabewert
      ...                       #
      return OBJ2               #
  OBJ2 = FUNC(OBJ)              # Aufruf
```

Alle Klassen - insbesondere eigene Klassen - (außer den Standard-Datentypen
NonType, bool, int, float, complex, str, tuple, list, dict, set, frozenset,
bytes, bytearray, memoryview) ERBEN von der obersten Basis-Klasse "object".

Leere Klasse anlegen (erbt automatisch von oberster Basis-Klasse "object"):

```
  class Klasse:       # eigentlich: class Klasse(object):
      pass            #                 pass
```

Objekte einer Klasse anlegen:

```
  OBJ = Klasse()
```

Objekte dieser Klasse (und auch die Objekte vom Typ Funktion, Modul, Klasse)
haben dann bereits folgende Eigenschaften und Fähigkeiten:

```
  OBJ.ATTR = WERT     # Attribute zuweisen
  del OBJ.ATTR        # Attribut löschen
  OBJ.ATTR            # Attribut lesen
  ATTR in OBJ         # Attribut vorhanden?
                      #
  OBJ == OBJ2         # WERT gleich   (fällt zurück auf "is"     ohne __eq__)
  OBJ != OBJ2         # WERT ungleich (fällt zurück auf "is not" ohne __eq__)
```