Metaohjelmointi Python-kielellä

Mikko Koho

Helsingin Yliopisto

26. marraskuuta 2014

Metaohjelmointi Python-kielellä

Metaohjelmointi

- Sellaisen ohjelman tekeminen, joka kirjoittaa uusia ohjelmia
- Ohjelma, joka manipuloi toisia ohjelmia ajon aikana
- Reflektio

Python

- Dynaaminen olio-ohjelmointikieli
- Dynaamisesti tyypitetty
- Ensimmäinen versio 1991
- Nykyään käytössä versiot 2 ja 3
- Kääntäjiä CPython, Jython, IronPython, PyPy

Syntaksi

```
lista = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
joukko = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
monikko = (0, 1, 2, 3, 4, 5, 6, 7, 8, 9)

print(lista == joukko)
# False

print(set(lista) == joukko)
# True

print(lista == list(joukko) == list(monikko))
# True
```

Syntaksi

```
parents, babies = (1, 1)
while babies < 100:
    print 'This generation has {0} babies'.format(babies)
    parents, babies = (babies, parents + babies)

# This generation has 1 babies
# This generation has 2 babies
# This generation has 3 babies
# This generation has 5 babies
# This generation has 8 babies
# This generation has 13 babies
# This generation has 21 babies
# This generation has 34 babies
# This generation has 55 babies
# This generation has 89 babies
```

---

https://wiki.python.org/moin/SimplePrograms

Luokat

```
class MyClassA(object):
  def a(self):
    print('foo')

class MyClassB(object):
  def b(self):
    print('bar')

class MyClassC(MyClassA, MyClassB):
  """Moniperija"""
  def c(self):
    self.a()
    self.b()

olio = MyClassC()
olio.c()
# foo    # bar
```

Introspektio

```
for attr in dir(olio):
  print(attr)

# __class__
# __delattr__
# __dict__
# __doc__
# __format__
# __getattribute__
# __hash__
# ...
# a
# b
# c
```

inspect-moduuli

```
import inspect

print(len(dir(inspect)))  # 87

print(inspect.getdoc(inspect.getdoc))
# Get the documentation string for an object.
#
# All tabs are expanded to spaces.  To clean up docstrings that are
# indented to line up with blocks of code, any whitespace than can be
# uniformly removed from the second line onwards is removed.

print(inspect.getdoc(inspect))
# Get useful information from live Python objects.   ...

print(inspect.ismodule(inspect))
# True
```

inspect-moduuli

```
for attr in dir(olio):
  docstr = str(inspect.getdoc(getattr(olio, attr)))
  docstr_head = docstr.splitlines()[0]
  print("olio.%s: %s" % (attr, docstr_head))

# olio.__class__: Moniperija
# olio.__delattr__: x.__delattr__('name') <==> del x.name
# olio.__dict__: dict() -> new empty dictionary
# olio.__doc__: str(object) -> string
# olio.__format__: default object formatter
# olio.__getattribute__: x.__getattribute__('name') <==> x.name
# olio.__hash__: x.__hash__() <==> hash(x)
# olio.__init__: x.__init__(...) initializes x; see help(type(x)) for s
# ...
# olio.a: None
# olio.b: None
# olio.c: None
```

inspect-moduuli

```
print(inspect.getsource(olio.c))
#       def c(self):
#             self.a()
#             self.b()

print(inspect.isfunction(olio.c))
# False

print(inspect.ismethod(olio.c))
# True

print(inspect.isroutine(olio.c))
# True

print(inspect.getmro(MyClassC))
# (<class '__main__.MyClassC'>,
#  <class '__main__.MyClassA'>,
```

Ohjelman ajonaikainen muokkaus

Monkey patching

Esimerkiksi jonkin kirjaston toiminnallisuuden muuttaminen ajon aikana.

Koodin kääntäminen ajonaikana

AST-puun muokkaus