

MASARYKOVA UNIVERZITA
FAKULTA INFORMATIKY



Nástroje na analýzu kódu v jazyku Python a vizualizácia ich výstupu

BAKALÁRSKA PRÁCA

Ján Vorčák

Brno, 2012

Prehlásenie

Prehlasujem, že táto bakalárska práca je mojím pôvodným autorským dielom, ktoré som vypracoval samostatne. Všetky zdroje, pramene a literatúru, ktoré som pri vypracovaní používal alebo z nich čerpal, v práci riadne citujem s uvedením úplného odkazu na príslušný zdroj.

Ján Vorčák

Vedúci práce: Mgr. Marek Grác

Pod'akovanie

Zhrnutie

Klíčové slova

Obsah

1	Úvod	2
2	O jazyku Python	3
2.1	Čo je to Python?	3
2.2	Introspekcia v jazyku Python	3
2.2.1	Funkcia dir()	3
2.2.2	Atribút __doc__	4
2.2.3	Atribút __name__	4
	Funkcia type()	5
2.2.4	Funkcia id()	5
2.2.5	Funkcie hasattr() a getattr()	5
2.2.6	Funkcia callable()	5
2.2.7	Funkcie isinstance() a subclass()	5
3	Nástroje na analýzu kódu pre jazyk Python	7
3.1	Nástroje analýzy kódu projektu	7
3.2	Nástroje na vizualizáciu projektu	7
4	Analýza nástroja Gaphas	8
4.1	Základná charakteristika nástroja Gaphas	8
4.2	Popis Gaphas API	8
4.2.1	Model	8
4.2.2	View	8
4.2.3	Controller	8
4.3	Zhrnutie	8
5	Projekt gpylint	9
5.1	Aplikácia jednotlivých nástrojov	9
5.2	Výhodnotenie	9
6	Záver	10

1 Úvod

2 O jazyku Python

2.1 Čo je to Python?

Python je vysoko úrovňový, objektovo orientovaný dynamický jazyk, ktorý vytvoril holandský programátor Guido van Rossum ako následníka jazyka ABC. Vyznačuje sa prehľadnou syntaxou, jednoduchosťou a modulárnosťou. Python podporuje viacero programátorských paradigmat, najmä objektovo orientované, imperatívne a čiastočne aj funkcionálne paradigma.

Existuje viacero implementácií jazyka Python. Medzi najznámejšie patria:

- CPython
- Jython
- Python pre .NET
- IronPython
- PyPy

Najpožívanejšou a najpodporovanejšou je ale CPython. Každá z implementácií sa môže líšiť špecifickými informáciami mimo štandardnej Python dokumentácie. Zdrojové kódy sú preložené do byte kódu a zvyčajne uložené v .pyc alebo .pyo súboroch, ktoré sú neskôr spúšťané virtuálnou

V štandardnej knižnici je dostupné množstvo datových typov ako napríklad reálne a komplexné čísla, celé čísla s neobmedzenou dĺžkou, znakové reťazce, zoznamy a slovníky. Datové typy sú silno a dynamicky typované. Použitie nekompatibilného typu spôsobí vyvolanie výnimky. Python podporuje objektovo orientované programovanie vrátane viacnásobnej dedičnosti. Kód je sústredený do modulov a balíkov s možnosťou importovať špecifický modul, triedu, funkciu alebo iný objekt. Za účelom ošetrenia chýb Python podporuje vyvolávanie a odchytávanie výnimiek. Automatická správa pamäti nahrádza nutnosť manuálne alokovať a uvoľňovať pamäť v kóde.

2.2 Introspekcia v jazyku Python

Introspekcia je schopnosť preskúmať daný objekt a rozhodnúť o jeho identite, vlastnostiach a schopnostiach. Jazyk Python podporuje rozsiahlu introspekciu objektov. Medzi hlavné informácie, ktoré potrebujeme o objektoch v jazyku Python zistiť patrí ich meno, typ, identita, vlastnosti, schopnosti a ich pôvod. Jazyk Python ponúka množstvo nástrojov, ktoré nám tieto vlastnosti umožňujú zistiť. Môžeme ich rozdeliť do dvoch hlavných skupín. V prvej skupine sú funkcie či už zo štandardnej knižnice alebo z pomocných modulov akým je napríklad modul inspect, do druhej skupiny radíme atribúty objektov, ktoré priamo v objekte uchovávajú užitočné informácie.

Funkcia dir() Funkcia dir() je jedným z hlavných nástrojov introspekcie v jazyku Python a vracia zotriedený zoznam mien atribútov objektu, ktorý bol uvedený ako jej argument. Funkcia je súčasťou štandardnej knižnice, takže nemusíme importovať žiaden modul pre jej

použitie. Na výpis funkcií zo štandardnej knižnice môžeme teda využiť samotnú funkciu `dir()`.

```
In [1]: print dir(__builtin__)[-10:]
['str', 'sum', 'super', 'tuple', 'type', 'unichr', 'unicode', 'vars',
  , 'xrange', 'zip']
```

V prípade, že je funkcia `dir()` použitá bez argumentov vracia zoznam mien, ktoré sú momentálne definované.

```
In [2]: print dir()
['In', 'Out', '_', '__', '___', '__builtin__', '__builtins__', '
  __name__', '_dh', '_i', '_il', '_i2', '_ih', '_ii', '_iii', '_oh',
  , '_sh', 'exit', 'get_ipython', 'help', 'quit']
```

Atribút `__doc__` Atribút `__doc__` obsahuje komentáre, ktoré popisujú objekt. V prípade, že prvý v module, triede alebo v metóde je znakový reťazec, je automaticky považovaný za `__doc__` atribút. V opačnom prípade sa jeho hodnota nastaví na `None`. Hodnoty `__doc__` sa z dôvodu kompaktnosti nevkladajú do byte kódu.

```
In [3]: print __doc__.__doc__
str(object) -> string
```

Return a nice string representation of the object.

If the argument **is** a string, the **return** value **is** the same object.

Atribút `__name__` Atribút `__name__` obsahuje názov objektu odvodený z jeho typu. Niektoré objekty, ako napríklad objekty typu `string` tento atribút neobsahujú. Tento atribút obsahujú napríklad moduly. V prípade, že spúšťame script priamo pomocou Python interpretu je atribút `__name__` nastavený na `'__main__'` nakoľko Python interpreter je považovaný za hlavný modul. Tak isto v prípade, že Python skript spúšťame z príkazového riadku. Je teda často používaný na rozpoznanie či daný modul len importujeme alebo priamo spúšťame.

```
In [4]: def my_function():
      ....:     pass
      ....:
```

```
In [5]: my_function.__name__
Out[5]: 'my_function'
```

```
In [6]: __name__
Out[6]: '__main__'
```

Funkcia `type()` Funkcia `type()` zo štandardnej knižnice vracia typ jej argumentu. Ten vracia v podobe typového objektu, ktorý môže byť porovnávaný s typmi definovanými v module `types`.

```
In [7]: type(my_function)
```

```
Out[7]: function
```

```
In [8]: type(1)
```

```
Out[8]: int
```

Funkcia id() Funkcia id() vracia unikátnu identitu objektu. Táto funkcia je užitočná nakoľko viacero premenných môže odkazovať na rovnaký objekt. Funkcia id() konkrétne vracia pamäťovú adresu objektu.

```
In [9]: id('string')
```

```
Out[9]: 3078023712L
```

```
In [10]: id(id)
```

```
Out[10]: 3078187660L
```

Funkcie hasattr() a getattr() V prípade, že je potrebné zistiť prítomnosť alebo hodnotu atribútu štandardná knižnica ponúka funkcie hasattr() a getattr().

```
In [11]: hasattr(id, '__name__')
```

```
Out[11]: True
```

```
In [12]: getattr(id, '__name__')
```

```
Out[12]: 'id'
```

Funkcia callable() V niektorých prípadoch môžu objekty slúžiť na vyvolanie určitého druhu udalosti. Pomocou funkcie callable() sa dá overiť, či je daný objekt spustiteľný.

```
In [13]: callable(id)
```

```
Out[13]: True
```

```
In [14]: callable(1)
```

```
Out[14]: False
```

Funkcie isinstance() a isinstance() Funkcia isinstance() vracia hodnotu v závislosti na tom, či je objekt inštanciou danej triedy. Funkcia vracia True aj v prípade, že je objekt inštanciou jej predka.

Funkcia isinstance() vracia True v prípade, že objekt reprezentujúci triedu je podtriedou druhého argumentu.

```
class Person(object):
```

```
    pass
```

```
class Student(Person):
```

```
    pass
```

```
p=Person()
```

```
s=Student()
```

```
In [15]: isinstance(s, Student)
```

```
Out[15]: True
```

```
In [16]: isinstance(s, Person)
```

```
Out[16]: True
```

```
In [17]: issubclass(Student, Person)
```

```
Out[17]: True
```

3 Nástroje na analýzu kódu pre jazyk Python

3.1 Nástroje analýzu kódu projektu

3.2 Nástroje na vizualizáciu projektu

4 Analýza nástroja Gaphas

4.1 Základná charakteristika nástroja Gaphas

Gaphas predstavuje zoskupenie knižníc a nástrojov na vykresľovanie grafických objektov na určené elementy grafického rozhrania GTK. Je naprogramovaný v jazyku Python a vydaný pod ??? licenciou.

4.2 Popis Gaphas API

Gaphas API využíva MVC návrhový vzor a môžeme ho teda rozdeliť na 3 hlavné časti.

- Model - canvas, items
- View - view
- Controller - tools

4.2.1 Model

4.2.2 View

View obsahuje všetko súvisiace so zobrazovaním a vykresľovaním jednotlivých elementov.

4.2.3 Controller

- api/view
- api/painters
- api/gtkview

4.3 Zhrnutie

5 Projekt gpylint

5.1 Aplikácia jednotlivých nástrojov

5.2 Vyhodnotenie

6 Závěr