



Jython: Združený zoznam

V tejto časti nášho seriálu sa ešte v krátkosti pozrieme na cykly v Jythone, vstupno-výstupné operácie a na spustenie jythonovských modulov z príkazového riadka.

Cyklus for

Cyklus `for` sa v Jythone používa častejšie než `while` a správa sa tu úplne inak než v Jave. Používa sa na iteráciu cez sekvencie. Syntax je nasledujúca:

```
for názov_premennej in sekvencia_alebo_výraz:
    blok
else:
    blok
```

Klauzula `else` sa správa rovnako ako v prípade cyklu `while`. Nasledujúce fragmenty kódu sú úplne rovnocenné:

```
>>> for x in [1,2,3]:
...     print x
...
1
2
3
```

```
>>> for x in range(1,4):
...     print x
...
1
2
3
```

```
>>> zoznam = [1,2,3]
>>> index = 0
>>> while index < len(zoznam):
...     print zoznam[index]
...     index += 1
...
1
2
3
```

Združený zoznam alebo List comprehension

Jedna z veľmi často vykonávaných operácií je vytváranie nových zoznamov zo zoznamov už existujúcich. Takto vytvorený nový zoznam možno dosiahnuť napríklad nasledujúcou konštrukciou:

```
>>> zoznam = ['prvy', 'druhy', 'treti']
>>> vysledok = []
>>> for x in zoznam:
...     vysledok.append(x + 'A')
...
>>> vysledok
['prvyA', 'druhyA', 'tretiA']
```

Tento príklad je priamočiary, ale možno ho zapísať pomocou konštrukcie nazvanej po anglicky *list comprehension*:

```
[vyraz for premenna in sekvencia]
```

Uvedený fragment sa dá prepísať takto:

```
>>> zoznam = ['prvy', 'druhy', 'treti']
>>> [x + 'A' for x in zoznam]
['prvyA', 'druhyA', 'tretiA']
```

Ďalej budeme namiesto anglického termínu *list comprehension* používať pojem *združený zoznam*. Združený zoznam môžeme využívať všade tam, kde možno použiť obyčajný zoznam. Často sa používa na pravej strane priradovacieho príkazu a v cykloch `for`. Nasledujúce konštrukcie zotriedia zoznam reťazcov podľa ich dĺžky:

```
>>> zoznam = ['a', 'ab', 'abc', 'abcd']
>>> dvojice = [(-len(x), x) for x in zoznam]
```

```
>>> dvojice
[(-4, 'abcd'), (-3, 'abc'), (-2, 'ab'), (-1, 'a')]
>>> zotriedZoznam = [x[l] for x in dvojice]
>>> zotriedZoznam
['abcd', 'abc', 'ab', 'a']
```

Konštrukciu združeného zoznamu možno rozšíriť pridaním filtrovacej podmienky:

```
[vyraz for premenna in sekvencia if testovacia_podmienka]
```

Pokiaľ chceme zotriediť zoznam reťazcov tak, aby sme získali iba prvky spĺňajúce určité podmienky, postupujeme takto:

```
>>> zoznam = ['a', 'ab', 'abc', 'abcd']
>>> dvojice = [(-len(x), x) for x in zoznam
               if len(x) >= 3]
>>>
>>> dvojice
[(-3, 'abc'), (-4, 'abcd')]
>>>
>>> zotriedZoznam = [x[l] for x in dvojice]
>>> zotriedZoznam
['abc', 'abcd']
```

Teraz si ukážeme najjednoduchšie nástroje, ktoré nám umožnia predkladať našim programom dáta na spracovanie s následným zobrazením výsledkov.

Vstup

Na interaktívne zadávanie údajov nám poslúži vstavaná funkcia `raw_input`, ktorá má nasledujúcu syntax:

```
raw_input([prompt])
```

Táto funkcia prečíta zo štandardného vstupu riadok, prevedie ho na reťazec (odsekne znaky konca riadka `\n`) a poskytne tento reťazec ako návratovú hodnotu. Pokiaľ bude pri volaní funkcie vyplnený nepovinný argument `prompt`, vypíše sa na štandardný výstup.

```
>>> s = raw_input("Zadajte vstupne udaje: ")
Zadajte vstupne udaje: Ahoj svet!!!
>>>
>>> s
'Ahoj svet!!!'
```

Výstup

Jython umožňuje tlačíť výstupné hodnoty na štandardný výstup prostredníctvom príkazu `print`, ktorého najjednoduchšia forma je:

```
print vyraz
```

Pri vykonávaní tohto príkazu sa najskôr vyhodnotí výraz a potom možno tlačíť na štandardný výstup nasledovaný znakom pre nový riadok:

```
>>> print 3
3
>>> print 1 + 2
3
>>> print ['a', 'b']
['a', 'b']
```

Tlačíť možno súčasne i niekoľko výrazov, pokiaľ použijete ako oddeľovač *čiarku*:

```
>>> print 'Vysledok je: ', 123, ' a dalsi je: ', [1,2,3]
Vysledok je: 123 a dalsi je: [1, 2, 3]
```

Ak je posledný výraz v príkaze `print` zakončený *čiarkou*, nevolí sa za výstup znak nového riadka:

```
>>> #print ukonceny novym riadkom
>>> print 1; print 2
1
2
>>> #teraz sa vytlací vsetko na jeden riadok
>>> print 1, ; print 2
1 2
```

Zápis programu

Až do tohto okamihu sme si pri predvážaní vlastností jazyka Jython vystačili s interaktívnym módom, ktorý nám umožňoval vkladať jednotlivé konštrukcie jazyka priamo a zároveň nám hneď zobrazil výsledky nášho úsilia. Táto vlastnosť je výhodná vtedy, keď vytvárame nejaký prototyp nášho programu a potrebujeme interaktívne testovať jeho správanie.

Na opätovné využitie nášho kódu však táto vlastnosť nie je veľmi vhodná. Aby sme mohli naše programy spúšťať kedykoľvek, keď to potrebujeme, je vhodnejšie ich ukladať do textových súborov. Na rozdiel od Javy názvy textových súborov nie sú nijako zviazané s ich obsahom. Slúžia len ako úložisko našich zdrojových kódov. Neskôr, až sa oboznámime s ďalšími konceptmi jazyka Jython, ukážeme si, ako tieto súbory organizovať do zložitejších štruktúr, ako napríklad knižníc a balíčkov.

Súbor, ktorý obsahuje zdrojový kód, sa nazýva *modul*. Moduly jazyka Jython majú extenziu `.py`. Pokiaľ chceme náš modul spustiť, stačí na príkazový riadok zadať príkaz:

```
jython mujmodul.py
```

```
D:\Jython\Source codes>jython mujmodul.py
Vytlačí 10 čísel po sebe:
```

```
0
1
2
3
4
5
6
7
8
9
```

```
D:\Jython\Source codes>
```

Čo sa týka obsahu modulu, moduly môžu v najjednoduchšom prípade obsahovať spustiteľný kód, ktorý je okamžite vykonaný. Moduly väčšinou okrem tohto kódu obsahujú odkazy na iné moduly, definície funkcií a tried, ktoré sa potom ďalej využívajú. Nabudúce vám ponúkneme malú ukážku jednoduchého modulu, ktorý vypočíta najmenšieho spoločného deliteľa dvoch čísel.



■ ŠTEFAN HAVLÍČEK,
Sales engineer, InterSystems B. V.