Moduly

Modul

Používání modulů má programátorovi umožnit rozšiřování zabudovaných schopností jazyka. Moduly podobně jako funkce „balí“ kód a data pro opakované použití. Moduly jsou soubory jazyka Python. Existuje celá řada modulů standardních, uživatel si však též může vytvářet moduly vlastní.

dir(m)

Užitečnou zabudovanou funkcí jazyka Python je funkce dir(). Když jí předáme jméno modulu, vrátí nám seznam platných jmen — často jsou to jména funkcí —, které modul obsahuje. Nasledují příkaz vypíše seznam:

```
print dir(__builtins__)
```

Poznámka: Pokud budete chtít učinit totéž pro libovolný jiný modul, musíte jej nejdříve zpřístupnit příkazem **import.**

Používání modulů

Python je extrémně rozšiřitelný jazyk v tom smyslu, že můžete přidávat nové možnosti tím, že provedete import potřebných modulů.

Standardní moduly:

sys

```
import sys # zpřístupní vnitřní funkce
sys.exit() # uvádíme předponu 'sys'
```

Pokud víme, že budeme funkce modulu používat velmi často a pokud nemají stejná jména jako funkce, které jsme již dříve importovali nebo vytvořili, pak můžeme psát:

```
from sys import *   # import všech jmen z modulu sys
exit()              # nyní nemusíme uvádět přeponu 'sys'
```

Další moduly a jejich obsah

Uvedeným způsobem můžeme importovat libovolné pythonské moduly, tedy i moduly, které si uživatel sám vytvořil.

V distribuci systému Python se nachází celá řada modulů a mnoho dalších lze stáhnout z Internetu. Než uživatel začne psát nějakou funkci, vždycky by si měl v dokumentaci ověřit, zda již náhodou někde neexistuje.

Moduly plní následující **úlohy:**

- **Znovupoužití kódu** – moduly lze načítat a jména v nich obsažená používat i v dalších modulech
- **Dělení jmenných prostorů** – moduly tvoří nejvyšší úroveň struktury kódu.
- I**mplementace sdílených služeb a dat** – potřebujeme-li k základním datovým strukturám přistupovat z více míst, implementujeme je jako modul a pak je importujeme všude, kde jsou potřeba.

Moduly jako jmenné prostory

Moduly jsou místa, kde jsou uložena jména. Jména v modulech se označují jako **atributy.**
Platí následující pravidla:

- příkazy modulu jsou při prvním importu spuštěny
- v rámci modulu není rozdíl mezi globálním a lokálním oborem platnosti.

Shrnutí

- Bližší určení jmen potřebujeme jen u příkazu *import,* protože *from* jména přímo kopíruje do našeho jmenného prostoru.
- Moduly jsou načteny po prvním importu (příkazy *import* nebo *from*).
- provedením kódu získáme atributy modulu.
- Pozdější importy (*import i from*) již nic načítat nemusí, atributy jsou už načteny.

Vytváření vlastních modulů

V jazyce Python není modul ničím zvláštním. Je to jednoduše prostý textový soubor s příkazy programu v jazyce Python. Obvykle jsou to definice funkcí. Takže, když napíšeme:

from sys import *

tak tím jakoby okopírujeme obsah sys.py do našeho programu. Je to téměř jako kdybychom text modulu přenesli přes schránku (clipboard) operacemi Kopírovat a Vložit (Ovšem, ve skutečnosti to takhle není, ale koncepčně to můžeme tak chápat.) Překladače některých jazyků (z význačných jmenujme C++) kopírují na základě požadavků obsah souborů s moduly do aktuálního programu doslova.

Modul vznikne vytvořením pythonského souboru (.py), který obsahuje funkce, které chceme používat v jiných programech. Potom jednoduše provedeme import našeho modulu přesně stejným způsobem, jako to děláme se standardními moduly.

Okopírujte si níže uvedenou funkci a uložte ji do svého souboru se jménem nasobky.py.

```
def tisk_tabulky(nasobitel):
    print "--- Tisk tabulky násobků číslem %d ---" % nasobitel
    for n in range(1, 13):
        print "%d x %d = %d" % (n, nasobitel, n * nasobitel)
```

Nyní na příkazový řádek systému Python napište:

```
>>> import nasobky
>>> nasobky.tisk_tabulky(12)
```

Právě jste vytvořili modul a použili jste ho.

Důležitá poznámka: Pokud jste Python nespustili ze stejného adresáře, ve kterém je uložen váš soubor nasobky.py, pak jej Python možná nenašel a ohlásil chybu. Pokud tomu tak skutečně je, můžete vytvořit proměnnou prostředí nazvanou PYTHONPATH, která obsahuje seznam adresářů, ve kterých se budou hledat moduly (tedy ty, které nejsou dodávány jako standardní spolu se systémem Python).

Literatura:

[1] Rubeš, J.: Nebojte se programovat, Computer Media, Bedihošť 2001
[2] Lutz, M., Ascher, D.: Naučte se Python, Grada, Praha 2003
[3] Beazley, D. M.: Python, Neocortex, Praha 2002
[4] Python Reference Manual
[5] Švec, J.: Létající circus, Python tutoriál, 2003