Class trong Python

# Tên và Object

Tương tự trong Java, trong Python cũng cho phép aliasing, tức nhiều reference cho cùng một dữ liệu, hay nói cách khác, nhiều tên trong các scope khác nhau cho cùng một object.

# Các scope trong Python và namespace

* Một namespace là một ánh xạ từ tên vào object. Các ánh xạ hầu hết được lập trình như một dict trong Python.
* Các loại namespace :
  + built-in (các hàm như len, abs, …)

Ví dụ:

print dir(type)

['\_\_abstractmethods\_\_', '\_\_base\_\_', '\_\_bases\_\_', '\_\_basicsize\_\_', '\_\_call\_\_', '\_\_class\_\_', '\_\_delattr\_\_', '\_\_dict\_\_', '\_\_dictoffset\_\_', '\_\_doc\_\_', '\_\_eq\_\_', '\_\_flags\_\_', '\_\_format\_\_', '\_\_ge\_\_', '\_\_getattribute\_\_', '\_\_gt\_\_', '\_\_hash\_\_', '\_\_init\_\_', '\_\_instancecheck\_\_', '\_\_itemsize\_\_', '\_\_le\_\_', '\_\_lt\_\_', '\_\_module\_\_', '\_\_mro\_\_', '\_\_name\_\_', '\_\_ne\_\_', '\_\_new\_\_', '\_\_reduce\_\_', '\_\_reduce\_ex\_\_', '\_\_repr\_\_', '\_\_setattr\_\_', '\_\_sizeof\_\_', '\_\_str\_\_', '\_\_subclasscheck\_\_', '\_\_subclasses\_\_', '\_\_subclasshook\_\_', '\_\_weakrefoffset\_\_', 'mro']

* + global names trong module
  + local names trong các hàm trong module
* Một attribute là mỗi tên đi sau một dấu chấm. Tồn tại song ánh từ attribute vào global namespace. Nó có thể bị xoá đi
  + del HuongHouse.area

print HuongHouse.getArea()

AttributeError: House instance has no attribute 'area'

(xoá ở object này không ảnh hưởng đến object khác, VinhHouse.getArea() OK. Nếu muốn xoá hết, dùng House.area)

* Các namespaces tạo ra ở các thời điểm khác nhau. Các built-in được tạo ra khi interpreter được khởi động, và không bao giờ bị xoá. Global namespaces khi module được đọc, cho đến khi thoát interpreter. Local namespace được tạo khi hàn được gọi, và xoá khi return hoặc khi raise exception. Mỗi bước recurrence lại có namespaces riêng của nó.
* Scope là vùng trong chương trình Python mà namespace được truy cập trực tiếp. Scope được xác định tĩnh (từ chương trình).
* Các scope:
  + Scope trong cùng: local names, được tìm đầu tiên
  + Nếu không có, tìm các scope bên ngoài (do recurrence hoặc do hàm bao nó) dần dần
  + Next-to-last scope: global name trong module hiện tại
  + Outermost scope (tìm sau cùng): built-in names
* Nếu một tên được định nghĩa global, thì reference và assignment nhảy vào scope giữa ngay thay vào scope trong cùng. Nếu không, các biến ngoài scope trong cùng nhưng không được declare global chỉ là read-only.
* Class tạo ra một namespace trong local scope.