PEP 484: 类型提示

原 文 PEP 484 -- Type Hints

作 者
Guido van Rossum (guido@python.org)
Jukka Lehtosalo (jukka.lehtosalo@iki.fi)
Lukasz Langa (lukasz@python.org)

翻 译 Wills Hua (wills.hua96@gmail.com)

状 态 暂定提案

类 型 标准类

发 布 历 史
2015年1月16日
2015年3月20日
2015年4月17日
2015年5月20日
2015年5月22日

目录

一、摘要

- PEP 3107已经引入了函数注解的语法, 但有意将语义保留为未定义. 目前第三方静态类型分析应用工具已经足够多了, 社区人员采用标准用语和标准库中的基线工具就将获益良多.

- 为了提供标准定义和工具, 本PEP引入了一个临时模块, 且列出一些不适用于注解情形的约定.
- 需要注意的是, 即使注解符合本规范, 本PEP依然明确不会妨碍注解的其他用法, 也不要求(或禁止)对注解的任意特殊处理. 正如PEP 333对Web框架的约定, 这只是为了能够更好地合作.
- 例如这个简单函数, 其参数和返回值都在注解中给出了声明:

  ```
  def greeting(name: str) -> str:
      return "Hello " + name
  ```

  虽然在运行时通过常规的 `__annotations__` 属性可以访问到上述注解, 但运行时并不会进行类型检查. 本提案假定存在一个独立的脱机类型检查程序, 用户可以自愿对源代码运行此检查程序. 这种类型检查程序实质上就是一种非常强大的查错工具. (当然某些用户是可以在运行时采用类似的检查程序实现"契约式设计"或JIT优化, 但这些工具尚未完全成熟.)
- 本提案受到mypy的强烈启发. 如, "整数序列"类型可以写为 `Sequence[int]`. 方括号表示无需向语言添加新的语法. 上述示例用到了自定义类型 `Sequence`, 是从纯Python模块 `typing` 中导入的.

二、理由与目标

三、注解的含义

四、类型定义的语法

五、与函数注释其他用法的兼容性

六、类型注释

七、指定类型

八、NewType工具函数

九、存根文件

十、typing模块

十一、Python 2.7和跨版本代码的建议语法

十二、未被接受的替代方案

十三、PEP开发过程

十四、参考文献