

目录

Python教程

Python简介

安装Python

第一个Python程序

Python基础

函数

高级特性

函数式编程

模块

面向对象编程

面向对象高级编程

使用__slots__

使用@property

多重继承

定制类

使用枚举类

使用元类

错误、调试和测试

IO编程

进程和线程

正则表达式

常用内建模块

常用第三方模块

virtualenv

图形界面

网络编程

电子邮件

访问数据库

Web开发

异步IO

实战

FAQ

期末总结

关于作者

廖雪峰

自己的Python课程

Python商业爬虫全解码

让天下没有爬不到的数据!

Python爬虫

+

数据分析

Python机器学习

+

深度学习

找廖雪峰老师

廖雪峰老师

自己的Java课程

Java高级架构师

更专业 更权威

源码分析专题

+

微服务架构专题

高并发分布式专题

+

性能优化专题

找廖雪峰老师

python免费公开课

编程学习网

授课模式：在线直播+课后视频，从零基础到中高级开发工程师

查看详情

python免费公开课

编程学习网

授课模式：在线直播+课后视频，从零基础到中高级开发工程师

查看详情

多重继承

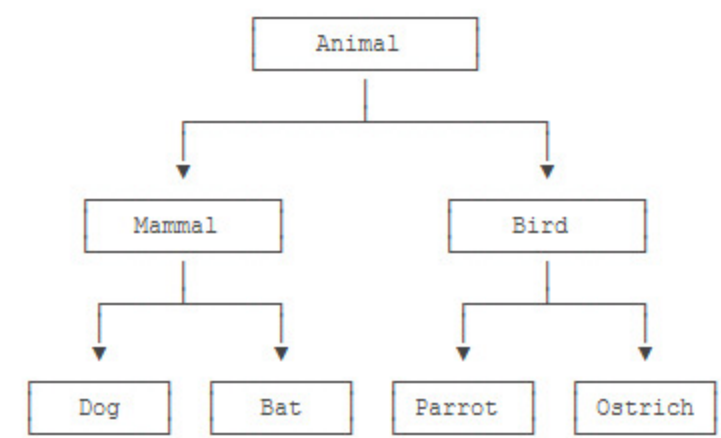
阅读: 332535

继承是面向对象编程的一个重要的方式，因为通过继承，子类就可以扩展父类的功能。

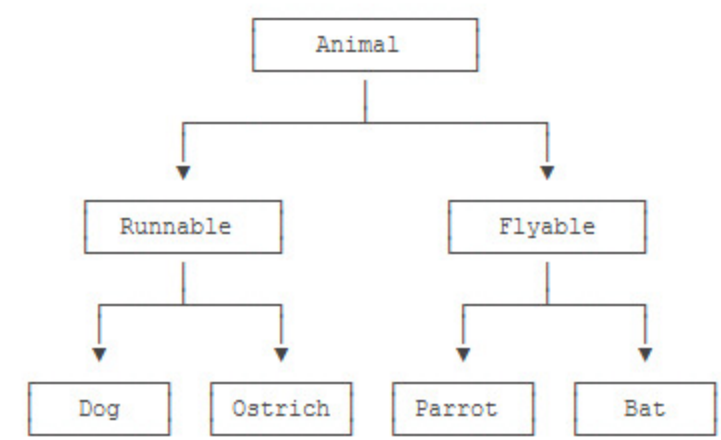
回忆一下 `Animal` 类层次的设计，假设我们要实现以下4种动物：

- Dog - 狗狗；
- Bat - 蝙蝠；
- Parrot - 鹦鹉；
- Ostrich - 鸵鸟。

如果按照哺乳动物和鸟类归类，我们可以设计出这样的类的层次：



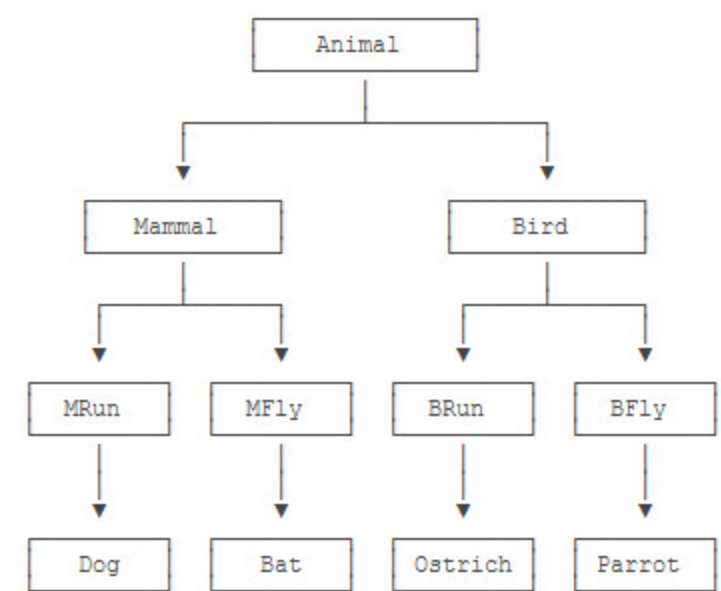
但是如果按照“能跑”和“能飞”来归类，我们就应该设计出这样的类的层次：



如果要把上面的两种分类都包含进来，我们就得设计更多的层次：

- 哺乳类：能跑的哺乳类，能飞的哺乳类；
- 鸟类：能跑的鸟类，能飞的鸟类。

这么一来，类的层次就复杂了：



如果要再增加“宠物类”和“非宠物类”，这么搞下去，类的数量会呈指数增长，很明显这样设计是不行的。

正确的做法是采用多重继承。首先，主要的类层次仍按照哺乳类和鸟类设计：

```
class Animal(object):
    pass

# 大类:
class Mammal(Animal):
    pass

class Bird(Animal):
    pass

# 各种动物:
class Dog(Mammal):
    pass

class Bat(Mammal):
    pass

class Parrot(Bird):
    pass

class Ostrich(Bird):
    pass
```

现在，我们要给动物再加上 `Runnable` 和 `Flyable` 的功能，只需要先定义好 `Runnable` 和 `Flyable` 的类：

```
class Runnable(object):
    def run(self):
        print('Running...')

class Flyable(object):
    def fly(self):
        print('Flying...')
```

对于需要 `Runnable` 功能的动物，就多继承一个 `Runnable`，例如 `Dog`：

```
class Dog(Mammal, Runnable):
    pass
```

对于需要 `Flyable` 功能的动物，就多继承一个 `Flyable`，例如 `Bat`：

```
class Bat(Mammal, Flyable):
    pass
```

通过多重继承，一个子类就可以同时获得多个父类的所有功能。

Mixin

在设计类的继承关系时，通常，主线都是单一继承下来的，例如，`Ostrich` 继承自 `Bird`。但是，如果需要“混入”额外的功能，通过多重继承就可以实现，比如，让 `Ostrich` 除了继承自 `Bird` 外，再同时继承 `Runnable`。这种设计通常称之为Mixin。

为了更好地看出继承关系，我们把 `Runnable` 和 `Flyable` 改为 `RunnableMixin` 和 `FlyableMixin`。类似的，你还可以定义出肉食动物 `CarnivorousMixin` 和植食动物 `HerbivoresMixin`，让某个动物同时拥有好几个Mixin：

```
class Dog(Mammal, RunnableMixin, CarnivorousMixin):
    pass
```

Mixin的目的就是给一个类增加多个功能，这样，在设计类的时候，我们优先考虑通过多重继承来组合多个Mixin的功能，而不是设计多层次复杂的继承关系。

Python自带的很多库也使用了Mixin。举个例子，Python自带了 `TCPServer` 和 `UDPServer` 这两类网络服务，而要同时服务多个用户就必须使用多进程或多线程模型，这两种模型由 `ForkingMixin` 和 `ThreadingMixin` 提供。通过组合，我们就可以创造出合适的服务来。

比如，编写一个多进程模式的TCP服务，定义如下：

```
class MyTCPServer(TCPServer, ForkingMixin):
    pass
```

编写一个多线程模式的UDP服务，定义如下：

```
class MyUDPServer(UDPServer, ThreadingMixin):
    pass
```

如果你打算搞一个更先进的协程模型，可以编写一个 `CoroutineMixin`：

```
class MyTCPServer(TCPServer, CoroutineMixin):
    pass
```

这样一来，我们不需要复杂而庞大的继承链，只要选择组合不同的类的功能，就可以快速构造出所需的子类。

小结

由于Python允许使用多重继承，因此，Mixin就是一种常见的设计。

只允许单一继承的语言（如Java）不能使用Mixin的设计。

读后有收获可以请作者喝咖啡，读后有疑问请加群讨论：



还可以分享给朋友：

分享到微博

廖雪峰

官方 | 独家

Python商业爬虫全解码

找廖雪峰老师

ACM金牌得主

全球顶尖名企一线数据科学家倾力指导

人工智能与自然语言/计算机视觉课程培训

Artificial Intelligence For NLP/CV Courses

无offer退全款

廖雪峰推荐

JAVA进阶教程

原价1599元

0元领取

阿里云

高性能云服务器首台5折

企业上云享半折扣，最大20Gbps内网带宽，450万PPS

立即选购

5折

E-LEARNING

Python全栈实战课程限时免费领取

120天腾讯课堂老师带你从零基础到项目实战，全系统学习爬虫、数据分析、Web开发等Python开发技术

腾讯云

学生服务器体验套餐10元/月

1核2G·1M带宽

50GB存储

立即抢购

- 评论
- 小白的十万个为什么

JasmineABcevvvvv created at a day ago, Last updated at a day ago

class Runnable(object): def run(self): print('Running..')

为什么这里要写成继承自超类object，而不是继承自Animal呢？

全部评论

回复
- python的多重继承，涉及C3算法，看下这个网址连接，说的很详细

abreze161 created at a day ago, Last updated at a day ago